Шай Шалев-Шварц и Шай Бен-Давид

Идеи машинного обучения

От теории к алгоритмам

Shai Shalev-Shwartz and Shai Ben-David

ЛАНЬ®

UNDERSTANDING MACHINE LEARNING

From Theory to Algorithms

ЛАНЬ®

CAMBRIDGE
UNIVERSITY PRESS

Шай Шалев-Шварц и Шай Бен-Давид

ИДЕИ МАШИННОГО ОБУЧЕНИЯ

От теории к алгоритмам

ДМК ПРЕСС
ИЗДАТЕЛЬСТВО
Москва, 2019

УДК 004.4
ББК 32.972
Ш18

Шалев-Шварц Ш., Бен-Давид Ш.
Ш18 Идеи машинного обучения: от теории к алгоритмам / пер. с англ. А. А. Слинкина. – М.: ДМК Пресс, 2019. – 436 с.: ил.

ISBN 978-5-97060-673-5

Машинное обучение – один из самых быстро развивающихся разделов информатики с приложениями в самых разных областях. Цель этой книги – познакомить читателя с фундаментальными принципами машинного обучения и характерными для него алгоритмическими парадигмами. Книга содержит обширный свод основополагающих теоретических идей машинного обучения и математические выкладки, благодаря которым эти идеи становятся практическими алгоритмами. Вслед за изложением базовых основ дисциплины рассматривается широкий спектр тем, не нашедших достаточного отражения в предшествующих учебниках: вычислительная сложность обучения, понятия выпуклости и устойчивости, важные алгоритмы, включая стохастический градиентный спуск, нейронные сети и обучение структурированному выводу, а также совсем недавние теоретические концепции, например, PAC-байесовский подход и границы сжатия.

Издание ориентировано на студентов старших курсов, обучающихся информатике, техническим наукам, математике или статистике, а также может быть полезно исследователям, желающим углубить свои теоретические знания. Предполагается, что читатель знаком с основами теории вероятностей, линейной алгебры, математического анализа и теории алгоритмов.

УДК 004.4
ББК 32.972

Original English language edition published by Cambridge University Press is part of the University of Cambridge. Copyright © 2014 by Shai Shalev-Shwartz and Shai Ben-David. Russian-language edition copyright © 2019 by DMK Press. All rights reserved.

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

ISBN 978-1-107-05713-5 (анг.)
ISBN 978-5-97060-673-5 (рус.)

© 2014 Shai Shalev-Shwartz and Shai Ben-David
© Издание, перевод, оформление, ДМК Пресс, 2019

Три Ш посвящают эту книгу трем М.

СОДЕРЖАНИЕ

Предисловие 15
Благодарности 16

Глава 1. Введение 17
1.1. Что такое обучение? 17
1.2. Когда необходимо машинное обучение? 19
1.3. Типы обучения 20
1.4. Связи с другими дисциплинами 22
1.5. Как читать эту книгу 23
1.5.1. Варианты построения курса на основе книги 24
1.6. Обозначения 25

ЧАСТЬ I. ОСНОВАНИЯ 28

Глава 2. Малый вперед 29
2.1. Формальная модель – схема статистического обучения 29
2.2. Минимизация эмпирического риска 31
2.2.1. Не все коту масленица – переобучение 31
2.3. Минимизация эмпирического риска с индуктивным смещением 32
2.3.1. Конечные классы гипотез 33
2.4. Упражнения 37

Глава 3. Формальная модель обучения 39
3.1. Вероятно почти корректное обучение 39
3.2. Более общая модель обучения 40
3.2.1. Отказ от предположения о реализуемости – агностическое PAC-обучение 41
3.2.2. Круг моделируемых проблем обучения 43
3.3. Резюме 45
3.4. Библиографические сведения 46
3.5. Упражнения 46

Глава 4. Обучаемость и равномерная сходимость 50
4.1. Равномерная сходимость – достаточное условие обучаемости 50
4.2. Конечные классы допускают агностическое PAC-обучение 51
4.3. Резюме 54
4.4. Библиографические сведения 54
4.5. Упражнения 54

Глава 5. Компромисс между смещением и сложностью 56
5.1. Теорема об отсутствии бесплатных завтраков 57
5.1.1. Теорема о бесплатных завтраках и априорное знание 59
5.2. Разложение ошибки 60
5.3. Резюме 61
5.4. Библиографические сведения 62
5.5. Упражнения 62

Глава 6. VC-размерность 63
6.1. Бесконечные классы могут быть обучаемыми 63
6.2. VC-размерность 64
6.3. Примеры 66
6.3.1. Ступенчатые функции 66
6.3.2. Интервалы 67
6.3.3. Осепараллельные прямоугольники 67
6.3.4. Конечные классы 68
6.3.5. VC-размерность и количество параметров 68
6.4. Фундаментальная теорема PAC-обучения 68
6.5. Доказательство теоремы 6.7 69
6.5.1. Лемма Зауэра и функция роста 70
6.5.2. Равномерная сходимость для классов небольшого эффективного размера 71
6.6. Резюме 74
6.7. Библиографические сведения 74
6.8. Упражнения 75

Глава 7. Неравномерная обучаемость 79
7.1. Неравномерная обучаемость 79
7.1.1. Характеристика неравномерной обучаемости 80
7.2. Структурная минимизация риска 81
7.3. Минимальная длина описания и бритва Оккама 85
7.3.1. Бритва Оккама 87
7.4. Другие концепции обучаемости – согласованность 88
7.5. Обсуждение различных понятий обучаемости 89
7.5.1. Еще раз о теореме об отсутствии бесплатных завтраков 92
7.6. Резюме 92
7.7. Библиографические сведения 93
7.8. Упражнения 93

Глава 8. Время обучения 96
8.1. Вычислительная сложность обучения 97
8.1.1. Формальное определение* 98
8.2. Реализация правила ERM 99
8.2.1. Конечные классы 100
8.2.2. Осепараллельные прямоугольники 100
8.2.3. Булевы конъюнкции 102
8.2.4. Обучение трехчленных ДНФ 103

8.3. Эффективно обучаемый, но не собственный алгоритм ERM 103
8.4. Трудность обучения* 104
8.5. Резюме 106
8.6. Библиографические сведения 106
8.7. Упражнения 106

ЧАСТЬ II. ОТ ТЕОРИИ К АЛГОРИТМАМ 110

Глава 9. Линейные предикторы 111
9.1. Полупространства 112
9.1.1. Линейное программирование для класса полупространств 113
9.1.2. Перцептрон для полупространств 114
9.1.3. VC-размерность класса полупространств 116
9.2. Линейная регрессия 117
9.2.1. Метод наименьших квадратов 118
9.2.2. Линейная регрессия для задач полиномиальной регрессии 119
9.3. Логистическая регрессия 120
9.4. Резюме 121
9.5. Библиографические сведения 122
9.6. Упражнения 122

Глава 10. Усиление 124
10.1. Слабая обучаемость 125
10.1.1. Эффективная реализация ERM для класса решающих пней 127
10.2. Алгоритм AdaBoost 128
10.3. Линейные комбинации базовых гипотез 131
10.3.1. VC-размерность $L(B, T)$ 133
10.4. Применение AdaBoost для распознавания лиц 134
10.5. Резюме 135
10.6. Библиографические сведения 136
10.7. Упражнения 136

Глава 11. Выбор и контроль модели 138
11.1. Выбор модели с помощью SRM 139
11.2. Контроль 140
11.2.1. Зарезервированный набор 140
11.2.2. Контроль при выборе модели 141
11.2.3. Кривая выбора модели 142
11.2.4. k-групповая перекрестная проверка 143
11.2.5. Обучение–контроль–тестирование 144
11.3. Что делать, если обучить не удается 144
11.4. Резюме 147
11.5. Упражнения 148

Глава 12. Выпуклые проблемы обучения 149
12.1. Выпуклость, липшицевость и гладкость 149
12.1.1. Выпуклость 149
12.1.2. Липшицевость 153

12.1.3. Гладкость ... 154
12.2. Выпуклые проблемы обучения ... 156
12.2.1. Обучаемость выпуклых проблем обучения ... 157
12.2.2. Выпуклые-липшицевы/гладкие-ограниченные проблемы обучения ... 158
12.3. Суррогатные функции потерь ... 159
12.4. Резюме ... 161
12.5. Библиографические сведения ... 161
12.6. Упражнения ... 161

Глава 13. Регуляризация и устойчивость ... 163
13.1. Минимизация регуляризированной потери ... 163
13.1.1. Гребневая регрессия ... 164
13.2. Устойчивые правила не подвержены переобучению ... 165
13.3. Регуляризация Тихонова как стабилизатор ... 166
13.3.1. Липшицева потеря ... 168
13.3.2. Гладкая неотрицательная потеря ... 169
13.4. Управление компромиссом между аппроксимацией и устойчивостью ... 170
13.5. Резюме ... 172
13.6. Библиографические сведения ... 172
13.7. Упражнения ... 173

Глава 14. Стохастический градиентный спуск ... 176
14.1. Градиентный спуск ... 177
14.1.1. Анализ метода ГС для выпуклых липшицевых функций ... 178
14.2. Субградиенты ... 180
14.2.1. Вычисление субградиентов ... 181
14.2.2. Субградиенты липшицевых функций ... 182
14.2.3. Субградиентный спуск ... 182
14.3. Стохастический градиентный спуск (СГС) ... 183
14.3.1. Анализ СГС для выпуклых–липшицевых–ограниченных функций ... 183
14.4. Варианты ... 185
14.4.1. Добавление шага проецирования ... 185
14.4.2. Переменный размер шага ... 186
14.4.3. Другие способы усреднения ... 186
14.4.4. Строго выпуклые функции* ... 187
14.5. Обучение с помощью СГС ... 188
14.5.1. Применение СГС для минимизации риска ... 188
14.5.2. Анализ СГС для выпуклых–гладких проблем обучения ... 190
14.5.3. Применение СГС для минимизации регуляризированной потери ... 191
14.6. Резюме ... 192
14.7. Библиографические сведения ... 192
14.8. Упражнения ... 192

Глава 15. Метод опорных векторов ... 194
15.1. Зазор и SVM с жестким зазором ... 194
15.1.1. Однородный случай ... 197

15.1.2. Выборочная сложность правила Hard-SVM 197
15.2. SVM с мягким зазором и регуляризация по норме 198
15.2.1. Выборочная сложность Soft-SVM 200
15.2.2. Сравнение границ, основанных на зазоре и норме, с размерностью 201
15.2.3. Рамповая функция потерь* 201
15.3. Условия оптимальности и «опорные векторы»* 202
15.4. Двойственность* 203
15.5. Реализация Soft-SVM с помощью СГС 204
15.6. Резюме 205
15.7. Библиографические сведения 205
15.8. Упражнения 206

Глава 16. Ядерные методы 207
16.1. Погружение в пространство признаков 207
16.2. Ядерный трюк 209
16.2.1. Ядра как способ выразить априорное знание 213
16.2.2. Характеристика ядерных функций* 214
16.3. Реализация Soft-SVM с ядрами 215
16.4. Резюме 216
16.5. Библиографические сведения 217
16.6. Упражнения 217

Глава 17. Многоклассовая категоризация, ранжирование и сложные проблемы предсказания 219
17.1. Один против всех и все пары 220
17.2. Линейные многоклассовые предикторы 222
17.2.1. Как построить Ψ 222
17.2.2. Стоимостная классификация 224
17.2.3. ERM 224
17.2.4. Обобщенная кусочно-линейная потеря 225
17.2.5. SVM и СГС в многоклассовом случае 226
17.3. Предсказание структурированного выхода 228
17.4. Ранжирование 230
17.4.1. Линейные предикторы для ранжирования 232
17.5. Двудольное ранжирование и многомерные показатели качества 235
17.5.1. Линейные предикторы для двудольного ранжирования 237
17.6. Резюме 239
17.7. Библиографические сведения 239
17.8. Упражнения 240

Глава 18. Решающие деревья 242
18.1. Выборочная сложность 243
18.2. Алгоритмы на решающих деревьях 244
18.2.1. Реализации меры выигрыша 245
18.2.2. Редукция 246
18.2.3. Пороговые правила разбиения для вещественных признаков 247

18.3. Случайные леса 247
18.4. Резюме 248
18.5. Библиографические сведения 248
18.6. Упражнения 248

Глава 19. Ближайшие соседи 250
19.1. Метод ***k*** ближайших соседей 250
19.2. Анализ 251
19.2.1. Граница обобщаемости для правила 1-NN 252
19.2.2. Проклятие размерности 255
19.3. Эффективная реализация* 256
19.4. Резюме 256
19.5. Библиографические сведения 257
19.6. Упражнения 257

Глава 20. Нейронные сети 260
20.1. Нейронные сети прямого распространения 261
20.2. Обучение нейронных сетей 262
20.3. Выразительная способность нейронных сетей 263
20.3.1. Геометрическая интерпретация 265
20.4. Выборочная сложность нейронных сетей 266
20.5. Время обучения нейронных сетей 267
20.6. СГС и обратное распространение 268
20.7. Резюме 272
20.8. Библиографические сведения 272
20.9. Упражнения 273

ЧАСТЬ III. ДОПОЛНИТЕЛЬНЫЕ МОДЕЛИ ОБУЧЕНИЯ 275

Глава 21. Онлайновое обучение 276
21.1. Онлайновая классификация в реализуемом случае 277
21.1.1. Онлайновая обучаемость 279
21.2. Онлайновая классификация в нереализуемом случае 283
21.2.1. Алгоритм взвешенного большинства 284
21.3. Онлайновая выпуклая оптимизация 288
21.4. Алгоритм онлайнового перцептрона 290
21.5. Резюме 293
21.6. Библиографические сведения 293
21.7. Упражнения 294

Глава 22. Кластеризация 296
22.1. Алгоритмы кластеризации на основе связи 299
22.2. Метод k-средних и другие методы кластеризации на основе минимизации стоимости 300
22.2.1. Алгоритм k-средних 302
22.3. Спектральная кластеризация 303
22.3.1. Разрезание графа 304
22.3.2. Лапласиан графа и ослабленные разрезы графа 304

22.3.3. Ненормированная спектральная кластеризация 305
22.4. Метод информационного горлышка* 306
22.5. Общий взгляд на кластеризацию 307
22.6. Резюме 309
22.7. Библиографические сведения 309
22.8. Упражнения 309

Глава 23. Понижение размерности 312
23.1. Метод главных компонент (PCA) 313
23.1.1. Более эффективное решение для случая $d \gg m$ 315
23.1.2. Реализация и демонстрация 315
23.2. Случайные проекции 317
23.3. Сжатое измерение сигнала 319
23.3.1. Доказательства* 321
23.4. PCA или сжатое измерение сигнала? 326
23.5. Резюме 327
23.6. Библиографические сведения 327
23.7. Упражнения 328

Глава 24. Порождающие модели 330
24.1. Оценка максимального правдоподобия 331
24.1.1. Оценка максимального правдоподобия для непрерывных случайных величин 332
24.1.2. Максимальное правдоподобие и минимизация эмпирического риска 333
24.1.3. Анализ обобщаемости 333
24.2. Наивная байесовская классификация 335
24.3. Линейный дискриминантный анализ 335
24.4. Скрытые переменные и EM-алгоритм 336
24.4.1. EM как алгоритм поочередной максимизации 338
24.4.2. EM-алгоритм для смеси нормальных распределений (мягкий алгоритм k-средних) 340
24.5. Байесовское рассуждение 341
24.6. Резюме 343
24.7. Библиографические сведения 343
24.8. Упражнения 343

Глава 25. Отбор и порождение признаков 345
25.1. Отбор признаков 346
25.1.1. Фильтры 347
25.1.2. Подходы на основе жадного отбора 348
25.1.3. Нормы, индуцирующие разреженность 351
25.2. Манипулирование и нормировка признаков 353
25.2.1. Примеры преобразований признаков 355
25.3. Обучение признаков 356
25.3.1. Обучение словаря с помощью автокодировщиков 356
25.4. Резюме 358

25.5. Библиографические сведения 358
25.6. Упражнения 359

ЧАСТЬ IV. ДОПОЛНИТЕЛЬНЫЕ ГЛАВЫ 361

Глава 26. Радемахеровская сложность 362

26.1. Радемахеровская сложность 362
26.1.1. Исчисление Радемахера 367
26.2. Радемахеровская сложность линейных классов 369
26.3. Границы обобщаемости метода SVM 371
26.4. Границы обобщаемости для предикторов с малой нормой ℓ_1 373
26.5. Библиографические сведения 374

Глава 27. Числа покрытия 375

27.1. Покрытие 375
27.1.1. Свойства 375
27.2. От покрытия к радемахеровской сложности через сцепление 376
27.3. Библиографические сведения 378

Глава 28. Доказательство фундаментальной теоремы теории обучения 379

28.1. Верхняя граница для агностического случая 379
28.2. Нижняя граница для агностического случая 380
28.2.1. Доказательство того, что $m(\epsilon, \delta) \geq 0{,}5\log(1/(4\boldsymbol{\delta}))/\epsilon^2$ 381
28.2.2. Доказательство того, что $m(\epsilon, 1/8) \geq 8\mathrm{d}/\epsilon^2$ 382
28.3. Верхняя граница для реализуемого случая 385
28.3.1. От ϵ-сетей к PAC-обучаемости 388

Глава 29. Многоклассовая обучаемость 389

29.1. Размерность Натараджана 389
29.2. Фундаментальная многоклассовая теорема 390
29.2.1. О доказательстве теоремы 29.3 390
29.3. Вычисление размерности Натараджана 391
29.3.1. Метод «один против всех» 391
29.3.2. Сведение многоклассовой категоризации к бинарной классификации в общем случае 392
29.3.3. Линейные многоклассовые предикторы 392
29.4. О хороших и плохих правилах ERM 394
29.5. Библиографические сведения 395
29.6. Упражнения 396

Глава 30. Границы сжатия 397

30.1. Границы сжатия 397
30.2. Примеры 399
30.2.1. Осепараллельные прямоугольники 399
30.2.2. Полупространства 399
30.2.3. Разделение полиномов 401

30.2.4. Разделение с зазором 401
30.3. Библиографические сведения 401

Глава 31. PAC-байесовский подход 402
31.1. PAC-байесовские границы 402
31.2. Библиографические сведения 404
31.3. Упражнения 405

Приложение A. Технические леммы 406

Приложение B. Концентрация меры 409
B.1. Неравенство Маркова 409
B.2. Неравенство Чебышева 410
B.3. Границы Чернова 411
B.4. Неравенство Хёфдинга 412
B.5. Неравенства Беннета и Бернштейна 413
B.5.1. Применение 414
B.6. Неравенство Слада 415
B.7. Концентрация случайных величин χ^2 415

Приложение C. Линейная алгебра 418
C.1. Основные определения 418
C.2. Собственные значения и собственные векторы 419
C.3. Положительно определенные матрицы 419
C.4. Сингулярное разложение 419

Литература 423

Предметный указатель 432

ЛАНЬ®

ПРЕДИСЛОВИЕ

Под *машинным обучением* понимается автоматизированное нахождение осмысленных закономерностей – паттернов – в данных. За последние двадцать лет оно стало обычным инструментом для решения почти всех задач, в которых требуется извлекать информацию из больших наборов данных. Нас со всех сторон окружают технологии, основанные на машинном обучении: поисковые системы учатся показывать наиболее полезные результаты (и при этом подсовывать прибыльную рекламу), антиспамные программы учатся фильтровать нашу почту, а операции с кредитными картами защищены программами, которые учатся распознавать мошенничество. Цифровые камеры учатся распознавать лица, а персональные помощники в смартфонах – понимать голосовые команды. Автомобили оснащаются системами предотвращения аварий, в которые встроены алгоритмы машинного обучения. Машинное обучение активно применяется и в научных дисциплинах, в т. ч. в биоинформатике, медицине и астрономии.

Все эти приложения объединяет общая черта – в отличие от традиционного применения компьютеров распознаваемые паттерны настолько сложны, что программист не способен явно сформулировать детальный алгоритм решения таких задач. Разумные существа приобретают или совершенствуют многие свои навыки путем *обучения* на собственном опыте (а не следования явным инструкциям). Задача инструментов машинного обучения – наделить программы способностью «обучаться» и адаптироваться. Первая цель этой книги – предложить строгое и вместе с тем достаточно простое для чтения введение в основные вопросы машинного обучения: что такое обучение, как обучается машина; как количественно оценить ресурсы, необходимые для обучения данной концепции; всегда ли возможно обучение; как узнать, завершился процесс обучения успешно или неудачно.

Вторая цель книги – изложить некоторые важнейшие алгоритмы машинного обучения. Мы выбрали алгоритмы, которые, с одной стороны, успешно применяются на практике, а с другой – представляют широкий спектр технических приемов обучения. Кроме того, мы уделили особое внимание алгоритмам, пригодным для обучения на больших объемах данных (так называемых «больших данных»), поскольку в последние годы наш мир стремительно «оцифровывается», так что объем данных, доступных для обучения невероятно вырос. В результате многие приложения больше не испытывают недостатка в данных, и узким местом становится время вычислений. Поэтому мы явным образом оцениваем как объем данных, так и время, необходимое для обучения данной концепции.

Книга состоит из четырех частей. Задача первой части – дать первоначальные строгие ответы на фундаментальные вопросы обучения. Мы опишем модель обучения, предложенную Валиантом, – вероятно почти корректное (Probably Approximately Correct – PAC), которая стала первым основательным ответом на вопрос «что такое обучение». Мы сформулируем три правила обучения – минимизация

эмпирического риска (Empirical Risk Minimization – ERM), структурная минимизация риска (Structural Risk Minimization – SRM) и минимальная длина описания (Minimum Description Length – MDL), – которые показывают, «как машина может обучаться». Мы количественно оценим объем данных, необходимый для обучения по правилам ERM, SRM и MDL, и, доказав теорему об «отсутствии бесплатных завтраков», покажем, каким образом обучение может завершиться неудачно. Мы также обсудим, сколько времени необходимо для обучения. Во второй части книги мы опишем различные алгоритмы обучения. В некоторых случаях мы сначала представим некий общий принцип обучения, а затем продемонстрируем, как алгоритм следует этому принципу. Если первые две части книги основаны на модели PAC, то в третьей мы расширим свой кругозор, введя в рассмотрение другие модели обучения. И, наконец, в последней части излагается более сложная теория.

Мы старались по возможности сделать книгу независимой. Однако предполагается, что читатель знаком с основами теории вероятностей, линейной алгебры, математического анализа и теории алгоритмов. Первые три части ориентированы на студентов первого года магистратуры по информатике, техническим наукам, математике или статистике. Они доступны также студентам средних курсов с хорошей подготовкой. Главы четвертой части могут быть полезны исследователям, желающим углубить свои теоретические знания.

Благодарности

В основу книги легли курсы «Введение в машинное обучение», прочитанные Шаем Шалев-Шварцем в Еврейском университете и Шаем Бен-Давидом в университете Ватерлоо. Первая черновая редакция появилась на свет из конспектов к курсу, прочитанному в Еврейском университете Шаем Шалев-Шварцем в 2010–2013 гг. Мы высоко ценим помощь Охада Шамира, который был учебным ассистентом на курсе 2010 г., и Алона Гонена, исполнявшего эту роль на курсах 2011–2013 гг. Охад и Алон подготовили конспекты и значительную часть упражнений. Алон, перед которым мы в долгу за помощь на протяжении всей работы над книгой, также подготовил ключ к решениям.

Мы глубоко признательны Дане Рубинштейн за ее ценнейший труд. Дана осуществила научное и литературное редактирование рукописи, превратив ее из разрозненных глав в связный, легко читаемый текст.

Отдельное спасибо Амиту Даниэли, который помогал в тщательной вычитке более сложных разделов книги и написал главу о многоклассовой обучаемости. Мы также благодарны членам иерусалимского клуба читателей, которые внимательно прочли каждую строчку рукописи и высказали конструктивную критику. Это Майя Элрой (Maya Alroy), Йосси Арджевани (Yossi Arjevani), Аарон Бирнбаум (Aharon Birnbaum), Алон Коэн (Alon Cohen), Алон Гонен (Alon Gonen), Рой Ливни (Roi Livni), Офер Месхи (Ofer Meshi), Дэн Розенбаум (Dan Rosenbaum), Дана Рубинштейн (Dana Rubinstein), Шахар Сомин (Shahar Somin), Алон Винников (Alon Vinnikov) и Йоав Вальд (Yoav Wald). Мы также признательны Гэлу Элидану (Gal Elidan), Амиру Глоберсону (Amir Globerson), Нике Нахталаб (Nika Haghtalab), Ши Маннору (Shie Mannor), Амнону Шашуа (Amnon Shashua), Нати Сребро (Nati Srebro) и Рут Эрнер (Ruth Urner) за полезные дискуссии.

Глава 1

ВВЕДЕНИЕ

Тема этой книги – автоматизированное обучение, или, как его чаще называют, машинное обучение (МО). То есть мы хотим запрограммировать компьютер так, чтобы он мог «обучаться» на доступных ему данных. Грубо говоря, обучение – это процесс преобразования эмпирического опыта в знания и умения. На вход алгоритма обучения подаются обучающие данные, представляющие опыт, а на выходе получаются знания, обычно принимающие вид другой компьютерной программы, способной выполнить некоторое задание. Если мы хотим описать эту идею формально-математически, то должны явно определить, что понимается под каждым вышеупомянутым термином. Что представляют собой обучающие данные, к которым будут обращаться наши программы? Как можно автоматизировать процесс обучения? Как оценить успешность этого процесса (т. е. качество результата обучающей программы)?

1.1. Что такое обучение?

Начнем с рассмотрения двух примеров, естественно возникающих при обучении животных. Некоторые из самых фундаментальных проблем МО видны уже в этом контексте, с которым все мы знакомы.

Недоверие к приманке – крысы учатся избегать отравленных приманок. Когда крыса обнаруживает еду с незнакомым видом или запахом, она сначала откусывает очень маленький кусочек, а ее дальнейшее поведение зависит от вкуса пищи и ее физиологических последствий. Если крыса почувствует себя плохо, то свяжет новую еду с недомоганием и не станет ее есть. Очевидно, здесь присутствует какой-то механизм обучения: животное воспользовалось прошлым опытом питания и приобрело умение определять безопасность пищи. Если с прошлым опытом были связаны негативные воспоминания, то животное предсказывает, что и в будущем последствия будут негативными.

Вдохновившись этим примером успешного обучения, продемонстрируем типичную задачу машинного обучения. Пусть требуется запрограммировать машину, которая будет обучаться фильтрации спама. Наивное решение было бы похоже на то, как крыса учится избегать отравленных приманок. Машина просто *запоминает* все предшествующие сообщения, которые были помечены как спам человеком. Когда приходит новое сообщение, машина ищет его среди уже известных спамных и, если находит, то отбрасывает. В противном случае сообщение отправляется в папку «Входящие».

Хотя такое «обучение путем запоминания» иногда полезно, этому подходу недостает важной черты обучающих систем – способности помечать ранее не встречавшиеся почтовые сообщения. Успешный обучаемый должен уметь совершать переход от отдельных примеров к более широкому *обобщению*. Это называется также *индуктивным рассуждением*, или *индуктивным выводом*. В описанном примере недоверия к приманке крыса, встретившая некоторый образчик пищи, распространяет свое отношение к нему на новые, ранее не виданные образчики пищи с похожим вкусом и запахом. Чтобы перейти к обобщению в задаче фильтрации спама, обучаемый может просканировать ранее предъявленные сообщения и выделить набор слов, наличие которых свидетельствует о спамном характере сообщения. Тогда, получив новое сообщение, машина сможет проверить, встречаются ли в нем подозрительные слова, и, соответственно, предсказать метку. Такая система потенциально могла бы правильно предсказывать метки ранее не предъявлявшихся сообщений.

Однако индуктивное рассуждение может приводить к ложным заключениям. Проиллюстрируем это еще одним примером обучения животных.

Суеверное поведение голубей. В эксперименте психолога Б.Ф. Скиннера группу голодных голубей помещали в клетку. Клетка была оборудована автоматическим механизмом, который через одинаковые интервалы выдавал голубям пищу вне зависимости от их поведения. Голодные голуби бродили по клетке, и в момент первой доставки пищи каждая птица чем-то занималась (долбила клювом пол, поворачивала голову и т. д.). Появление пищи подкрепляло действие каждой птицы, и впоследствии она чаще повторяла это действие, чем другие. Это, в свою очередь, увеличивало вероятность, что при следующем кормлении голуби также будут совершать это действие. В результате формируется цепочка событий, которая подкрепляет у голубей ассоциацию между появлением пищи и теми действиями, которые они совершали при первом кормлении. В дальнейшем они совершают эти действия намеренно[1].

Чем отличаются механизмы, приводящие к суеверию и к полезному обучению? Этот вопрос имеет первостепенное значение для разработки автоматизированных систем обучения. Человек может полагаться на здравый смысл, отбрасывая случайные бессмысленные заключения, но, переходя к обучению машины, мы должны сформулировать четкие, не допускающие двоякого толкования принципы, которые не дадут программе прийти к бессмысленным или бесполезным выводам. Разработка таких принципов и есть главная цель теории машинного обучения.

Так в чем же все-таки причина того, что обучение крыс оказалось успешнее, чем обучение голубей? В качестве первого шага к ответу на этот вопрос рассмотрим более пристально феномен недоверия к приманке.

Еще раз о недоверии к приманке: у крыс не формируется условно-рефлекторная связь между пищей и ударом электрическим током или между звуком и тошнотой. Механизм недоверия к приманке у крыс оказывается сложнее, чем могло бы показаться. В экспериментах Гарсия (Garcia & Koelling, 1996) было продемонстрировано, что если неприятный стимул, сопутствующий поеданию пищи, заменить, например, ударом электрическим током (а не тошнотой), то условного

[1] См. http://psychclassics.yorku.ca/Skinner/Pigeon.

рефлекса не возникает. Даже после нескольких испытаний, в которых поедание определенной пищи сопровождалось неприятным ударом током, крысы не отказывались от этой пищи. Условный рефлекс не формировался и тогда, когда свойство пищи, вызывающее тошноту (например, вкус или запах), заменялось звуковым сигналом. У крыс, похоже, имеется априорное «встроенное» знание о том, что временная связь между пищей и тошнотой может быть причинно-следственной, а наличие причинно-следственной связи между поеданием пищи и ударом током или между звуками и тошнотой маловероятно.

Мы приходим к выводу, что одним из отличий между недоверием к приманке и суеверному поведению голубей является наличие *априорного знания*, которое модифицирует механизм обучения. Его еще называют *индуктивным смещением* (inductive bias). Голуби в эксперименте готовы принять *любое* объяснение появления еды. Но крысы «знают», что еда не может быть причиной удара электрическим током и что сопровождение еды шумом вряд ли может оказать влияние на питательные качества этой еды. Процесс обучения крыс смещен в сторону обнаружения определенных паттернов, тогда как другие временные корреляции между событиями игнорируются.

Выясняется, что включение априорного знания, смещающего процесс обучения, является обязательным условием успешности алгоритмов обучения (это утверждение, строго сформулированное и доказанное в главе 5, получило название «теоремы об отсутствии бесплатных завтраков»). Разработка инструментов для выражения знания о предметной области, трансляции их в смещение алгоритма обучения и количественной оценки влияния этого смещения на успех обучения – центральная тема теории машинного обучения. Грубо говоря, чем сильнее априорные знания (или априорные предположения), с которыми мы начинаем процесс обучения, тем легче идет обучение на последующих примерах. Однако чем сильнее априорные предположения, тем менее гибким будет обучение – оно ограничено необходимостью соблюдать эти предположения. Мы будем явно обсуждать эти вопросы в главе 5.

1.2. Когда необходимо машинное обучение?

Когда возникает необходимость прибегнуть к машинному обучению, вместо того чтобы непосредственно запрограммировать компьютер на решение стоящей задачи? Два свойства задачи наводят на мысль воспользоваться программами, которые могут обучаться и совершенствоваться на основе своего «опыта»: сложность и требование адаптивности.

Задачи, слишком сложные для программирования

- *Задачи, решаемые животными или людьми*. Есть немало задач, которые человек решает естественно и привычно, и тем не менее мы недостаточно хорошо понимаем, как мы это делаем, чтобы оформить это в виде программы. Вот несколько примеров: вождение автомобиля, распознавание речи, понимание того, что изображено на картинке. Во всех этих случаях современные программы машинного обучения, «учащиеся на своем опыте», позволяют

добиться вполне удовлетворительных результатов, если им предъявить достаточно много обучающих примеров.

- *Задачи, выходящие за пределы человеческих возможностей*. Еще один класс задач, для которого машинное обучение дает очевидный выигрыш, связан с анализом очень больших и сложных наборов данных: астрономические данные, преобразование архивов медицинских записей в медицинские знания, прогнозирование погоды, анализ генома, поисковые системы в вебе, электронная торговля. По мере накопления цифровых данных становится очевидно, что в архивах скрыты информационные сокровища, вот только для человеческого разума они слишком велики и сложны. Обучение машин выявлять осмысленные паттерны в больших и сложных наборах данных – это многообещающая область, в которой сочетание обучающихся программ с почти безграничной памятью и постоянно растущим быстродействием компьютеров открывает новые горизонты.

Адаптивность

Одно из ограничений программируемых инструментов – отсутствие гибкости: после того, как программа написана и установлена, она уже не изменяется. С другой стороны, многие задачи изменяются со временем или в зависимости от конкретного пользователя. Средства машинного обучения – программы, поведение которых адаптируется к входным данным, – предлагают решение таких проблем; они по природе своей приспосабливаются к изменениям среды, с которой взаимодействуют. Типичные примеры успешного применения машинного обучения к такого рода задачам – программы распознавания рукописного текста (одна и та же программа способна адаптироваться к почерку разных пользователей), программы обнаружения спама (автоматически адаптируются к изменениям в характере спамных сообщений) и программы распознавания речи.

1.3. Типы обучения

Конечно, обучение – чрезвычайно обширная область. Поэтому в машинном обучении выделяют несколько дисциплин, различающихся типами обучения. Мы дадим приблизительную таксономию парадигм обучения, чтобы читатель понимал, какое место эта книга занимает в обширном массиве исследований по машинному обучению.

Мы будем классифицировать парадигмы обучения по четырем параметрам.

- **С учителем и без учителя.** Поскольку обучение подразумевает взаимодействие между обучаемым и окружением, задачи обучения можно различать по характеру этого взаимодействия. Прежде всего, отметим разницу между обучением с учителем и без учителя. В качестве примера сравним задачи о распознавании спама и об обнаружении аномалий. В случае задачи о распознавании спама будем считать, что обучаемый получает образцы сообщений с метками «спам – не спам». По результатам такого обучения обучаемый должен выработать правило пометки новых сообщений. С другой стороны, в задаче об обнаружении аномалий обучаемый полу-

чает только тело сообщения (без каких бы то ни было меток) и должен выявить «необычные» сообщения.

Если рассматривать обучение более абстрактно – как процесс «использования эмпирического опыта для получения знаний», то обучению с учителем соответствует случай, когда «опыт» – обучающий пример – содержит существенную информацию (например, метки «спам – не спам»), отсутствующую в еще не виденных «тестовых примерах», к которым будут применены обретенные в ходе обучения знания. При таком подходе цель приобретения знаний – предсказать отсутствующую в тестовых данных информацию. Мы можем считать окружение учителем, который «наставляет» обучаемого, предоставляя ему дополнительную информацию (метки). А в случае обучения без учителя между обучающими и тестовыми данными нет никакого различия. Обучаемый обрабатывает входные данные, имея целью составить на их основе некоторый дайджест, или сжатое представление. Типичный пример такой задачи – кластеризация набора данных, т. е. разбиение его на подмножества похожих объектов.

Существует также промежуточный тип обучения, когда обучающие примеры содержат больше информации, чем тестовые, а обучаемый должен предсказать для тестовых примеров еще больше информации. Например, можно попытаться обучить функцию, которая для каждой позиции на шахматной доске оценивает, насколько позиция белых лучше, чем у черных. При этом единственная информация, доступная на этапе обучения на позициях фактически сыгранных партий, – метка, сообщающая, кто в итоге выиграл. Такие подходы обычно изучаются в дисциплине, называемой *обучение с подкреплением*.

- **Активный и пассивный обучаемый.** Парадигмы обучения могут классифицироваться по роли обучаемого. Мы различаем «активного» и «пассивного» обучаемого. Активный обучаемый взаимодействует с окружением на этапе обучения, например, задавая вопросы или ставя эксперименты, тогда как пассивный только наблюдает за информацией, поставляемой окружением (или учителем), не оказывая на нее никакого влияния и не направляя процесс обучения. Отметим, что обучаемый фильтр спама обычно пассивен: он ждет, пока пользователи пометят входящие сообщения. Но можно было бы представить и активную конфигурацию, когда обучаемый сам выбирает или даже составляет сообщения и просит пользователей их пометить, тем самым улучшая свое понимание того, что такое спам.
- **Полезность учителя.** В процессе обучения человека, будь то ребенок дома или ученик в школе, обычно присутствует готовый прийти на помощь учитель, который пытается дать обучаемому информацию, наиболее полезную для достижения цели обучения. Напротив, когда ученый исследует природу, окружающую среду, роль учителя можно в лучшем случае считать пассивной: яблоко падает, солнце светит, дождь льет, не обращая никакого внимания на потребности обучаемого. Для моделирования таких типов обучения мы постулируем, что обучающие данные (опыт обучаемого) порождаются некоторым случайным процессом. Это основная идея «статистического обучения». Наконец, обучение имеет место и тогда, когда входные данные генерирует некий противодействующий «учитель». Так может

быть и при обучении фильтра спама (когда спамер стремится сбить с толку проектировщика фильтра), и при обучении обнаружению мошеннических действий. Модель противодействующего учителя – так называемое состязательное обучение – применяется также в худшем случае, когда нельзя безопасно предполагать более мягких условий. Если система может обучиться в условиях, когда учитель активно противодействует обучению, то она гарантированно добьется успеха при взаимодействии с любым сколь угодно странным учителем.

- **Онлайновое и пакетное обучение.** И напоследок упомянем различие между ситуацией, в которой обучаемый должен отвечать в режиме онлайн на протяжении всего процесса обучения, и ситуацией, когда разрешается применить обретенные знания уже после обработки значительного объема данных. Например, биржевой брокер должен ежедневно принимать решения на основе полученного к этому моменту опыта. Со временем он, возможно, станет экспертом, но в процессе обучения может допускать дорогостоящие ошибки. С другой стороны, во многих задачах добычи данных обучаемый – добытчик – располагает большим объемом обучающих данных, с которыми он может экспериментировать, прежде чем от него потребуют конкретных выводов.

ЛАНЬ®

В этой книге мы будем обсуждать только подмножество возможных парадигм обучения. Основное внимание мы уделим статистическому пакетному обучению с учителем и пассивным обучаемым (например, мы попытаемся выдавать прогноз течения болезни на основе больших архивов медицинских записей, которые были собраны независимо и уже помечены информацией о судьбе пациента). Мы также кратко остановимся на онлайновом обучении и пакетном обучении без учителя (в частности, кластеризации).

1.4. Связи с другими дисциплинами

Располагаясь на стыке различных дисциплин, машинное обучение связано многочисленными нитями с математической статистикой, теорией информации, теорией игр и оптимизацией. Естественно, оно является частью информатики, поскольку наша цель – программирование машин, способных обучаться. В некотором смысле машинное обучение можно рассматривать как отрасль искусственного интеллекта (ИИ), поскольку умение обращать опыт в знания и обнаруживать осмысленные паттерны в сложной сенсорной информации – это краеугольный камень разума человека (и животного). Однако следует отметить, что, в отличие от традиционного ИИ, машинное обучение стремится не столько добиться автоматизированной имитации разумного поведения, сколько использовать сильные стороны компьютеров и присущие только им возможности, чтобы дополнить человеческий разум, и на этом пути зачастую решает задачи, выходящие за рамки возможностей человека. Например, способность просматривать и обрабатывать гигантские базы данных позволяет программам машинного обучения обнаруживать паттерны за пределами человеческого восприятия.

ЛАНЬ®

То, что мы называем в машинном обучении опытом, часто относится к случайно сгенерированным данным. Задача обучаемого – на основе обработки слу-

чайных примеров прийти к выводам, справедливым для окружения, из которого эти примеры выбирались. Такое описание машинного обучения раскрывает его тесную связь со статистикой. И действительно, между этими двумя дисциплинами много общего – в части как целей, так и применяемых методов. Но есть, однако, и существенные различия в постановке задачи: если врач выдвигает гипотезу о наличии корреляции между курением и болезнями сердца, то задача статистика – обработать выборку пациентов и проверить достоверность этой гипотезы (это типичная статистическая задача о проверке гипотез). Напротив, цель машинного обучения – использовать данные, собранные о пациентах, и предложить описание возможных причин болезней сердца. При этом мы надеемся, что автоматизированные методы смогут выявить осмысленные паттерны (или гипотезы), ускользнувшие от внимания человека.

В отличие от традиционной статистики, в машинном обучении вообще и в этой книге в частности важную роль играют алгоритмические соображения. Суть машинного обучения заключается в обучении компьютеров, поэтому от алгоритмов никуда не деться. Мы разрабатываем алгоритмы для решения задач обучения, и их вычислительная эффективность нам отнюдь не безразлична. Еще одно отличие состоит в том, что статистику часто интересует асимптотическое поведение (например, сходимость выборочных статистических оценок, когда размер выборки стремится к бесконечности), тогда как теория машинного обучения акцентирует внимание на конечных выборках. Точнее, теория пытается оценить, какой верности предсказаний можно ожидать от обучаемого при данном размере доступных выборок.

Существуют и другие различия между обеими дисциплинами, но здесь мы упомянем только одно из них. В статистике, вообще говоря, имеется некоторое априорное предположение о модели данных (например, о нормальности порождающих данные распределений или о линейности функциональных зависимостей), тогда как в машинном обучении упор делается на работу в условиях «неизвестного распределения», когда обучаемый старается не делать никаких предположений о природе распределения данных, а обучающему алгоритму предоставляется возможность определить, какая модель наилучшим образом аппроксимирует процесс порождения данных. Для строгого обсуждения этого вопроса необходима математическая подготовка, но мы вернемся к нему позже и, в частности, в главе 5.

1.5. Как читать эту книгу

В первой части книги излагаются базовые теоретические принципы, лежащие в основе машинного обучения (МО). В каком-то смысле это фундамент, на котором возведено здание книги. На базе этой части можно прочитать мини-курс по теоретическим основаниям МО.

Во второй части книги дается введение в наиболее распространенные алгоритмические подходы к машинному обучению с учителем. Подмножество этих глав можно использовать как введение в машинное обучение в рамках общего курса ИИ для студентов, изучающих математику, информатику или технические дисциплины.

В третьей части рамки обсуждения расширяются: мы переходим от статистической классификации к другим моделям обучения. Рассматривается онлайновое обучение, обучение без учителя, понижение размерности, порождающие модели и обучение признаков.

Четвертая часть книги ориентирована на читателей с исследовательской жилкой. Здесь излагаются более сложные математические методы, необходимые для анализа и дальнейшего развития теоретического машинного обучения.

В приложениях описаны некоторые технические средства, используемые в книге. В частности, приводятся основные результаты из теории концентрации меры и линейной алгебры.

Разделы, помеченные звездочкой, адресованы более подготовленным студентам. Каждая глава завершается списком упражнений. Ключ к решениям имеется на сайте курса.

ЛАНЬ®

1.5.1. Варианты построения курса на основе книги

14-недельный вводный курс для студентов магистратуры

1. Главы 2–4.
2. Глава 9 (без вычисления VC).
3. Главы 5–6 (без доказательств).
4. Глава 10.
5. Главы 7, 11 (без доказательств).
6. Главы 12, 13 (с некоторыми простыми доказательствами).
7. Глава 14 (с некоторыми простыми доказательствами).
8. Глава 15.
9. Глава 16.
10. Глава 18.
11. Глава 22.
12. Глава 23 (без доказательств для сжатого измерения сигнала).
13. Глава 24.
14. Глава 25.

ЛАНЬ®

14-недельный повышенный курс для студентов магистратуры

1. Главы 26, 27.
2. (продолжение)
3. Главы 6, 28.
4. Глава 7.
5. Глава 31.
6. Глава 30.
7. Главы 12, 13.
8. Глава 14.
9. Глава 8.
10. Глава 17.
11. Глава 29.
12. Глава 19.
13. Глава 20.
14. Глава 21.

1.6. Обозначения

По большей части обозначения, встречающиеся в книге, либо стандартны, либо определяются на месте. В этом разделе мы опишем основные применяемые соглашения и приведем сводную таблицу обозначений (табл. 1.1). Читатель может пропустить этот раздел и возвращаться к нему по ходу чтения книги, встретив непонятное обозначение.

Скаляры и абстрактные объекты обозначаются строчными буквами (например, x и λ). Если мы хотим подчеркнуть, что объект является вектором, то употребляем полужирный шрифт (например, $\mathbf{x}$ и $\boldsymbol{\lambda}$). i-й элемент вектора $\mathbf{x}$ обозначается x_i. Заглавными буквами обозначаются матрицы, множества и последовательности. О чем конкретно идет речь, всегда понятно из контекста. Как мы скоро увидим, на вход алгоритма обучения подается последовательность обучающих примеров. Мы обозначаем z абстрактный пример, а $S = z_1, \ldots, z_m$ – последовательность m примеров. Исторически сложилось, что S называют обучающим *набором*, или *множеством*, однако мы всегда предполагаем, что S – *последовательность*, а не множество. Последовательность m векторов обозначается $\mathbf{x}_1, \ldots, \mathbf{x}_m$, а i-й элемент вектора $\mathbf{x}_t$ – $x_{t,i}$.

В книге повсеместно используются понятия из теории вероятностей. Мы обозначаем $\mathcal{D}$ распределение на некотором множестве[1], например, Z. Запись $z \sim \mathcal{D}$ означает, что z выбрано из распределения $\mathcal{D}$. Если $f\colon Z \to \mathbb{R}$ – случайная величина, то ее математическое ожидание обозначается $\mathbb{E}_{z\sim\mathcal{D}}[f(z)]$. Иногда мы сокращаем эту запись до $\mathbb{E}[f]$, если зависимость от z очевидна из контекста. В случае, когда $f\colon Z \to \{\text{true, false}\}$, для обозначения $\mathcal{D}(\{z\colon f(z) = \text{true}\})$ используется также нотация $\mathbb{P}_{z\sim\mathcal{D}}[f(z)]$. В следующей главе мы введем также нотацию $\mathcal{D}^m$ для обозначения вероятности на Z^m, индуцированной выборкой $(z_1, \ldots, z_m)$, где каждая точка z_i выбирается из распределения $\mathcal{D}$ независимо от других точек.

Таблица 1.1. Сводка обозначений

Символ	Значение
$\mathbb{R}$	множество вещественных чисел
$\mathbb{R}^d$	множество d-мерных векторов над $\mathbb{R}$
$\mathbb{R}_+$	множество неотрицательных вещественных чисел
$\mathbb{N}$	множество натуральных чисел
$O, o, \Theta, \omega, \Omega, \tilde{O}$	асимптотическая нотация (см. в тексте)
$1_{[\text{булево выражение}]}$	индикаторная функция (равна 1, если выражение истинно, и 0 в противном случае)
$[a]_+$	$= \max(0, a)$
$[n]$	множество $\{1, \ldots, n\}$ (для $n \in \mathbb{N}$)
$\mathbf{x}, \mathbf{v}, \mathbf{w}$	векторы-столбцы
x_i, v_i, w_i	i-й элемент вектора
$\langle \mathbf{x}, \mathbf{v} \rangle$	$= \sum_{i=1}^{d} x_i v_i$ (скалярное произведение)
$\|\|\mathbf{x}\|\|_2$ или $\|\|\mathbf{x}\|\|$	$= \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$ (ℓ_2 – норма вектора $\mathbf{x}$)

[1] Математически правильнее было бы определять $\mathcal{D}$ на σ-алгебре подмножеств Z. Читатели, не знакомые с теорией меры, могут пропустить несколько сносок и замечаний, относящихся к формальным определениям и предположениям об измеримости.

Окончание табл. 1.1

Символ	Значение
$\|\|\mathbf{x}\|\|_1$	$= \sum_{i=1}^{d} \|x_i\|$ (ℓ_1 – норма вектора $\mathbf{x}$)
$\|\|\mathbf{x}\|\|_\infty$	$= \max_i \|x_i\|$ (ℓ_∞ – норма вектора $\mathbf{x}$)
$\|\|\mathbf{x}\|\|_0$	количество ненулевых элементов $\mathbf{x}$
$A \in \mathbb{R}^{d,k}$	матрица $d \times k$ над $\mathbb{R}$
$A^\top$	матрица, транспонированная к A
$A_{i,j}$	элемент (i, j) матрицы A
$\mathbf{x}\mathbf{x}^\top$	матрица A размера $d \times d$ такая, что $A_{i,j} = x_i x_j$ (где $\mathbf{x} \in \mathbb{R}^d$)
$\mathbf{x}_1, \ldots, \mathbf{x}_m$	последовательность m векторов
$x_{i,j}$	j-й элемент i-го вектора последовательности
$\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(T)}$	значения вектора $\mathbf{w}$ в итеративном алгоритме
$w_i^{(t)}$	i-й элемент вектора $\mathbf{w}^{(t)}$
$\mathcal{X}$	множество образцов
$\mathcal{Y}$	множество меток
Z	множество примеров
$\mathcal{H}$	класс гипотез (множество)
$\ell : \mathcal{H} \times Z \to \mathbb{R}_+$	функция потерь
$\mathcal{D}$	распределение на некотором множестве (обычно на Z или на $\mathcal{X}$)
$\mathcal{D}(A)$	вероятность множества $A \subseteq Z$ при распределении $\mathcal{D}$
$z \sim \mathcal{D}$	выборка z из распределения $\mathcal{D}$
$S = z_1, \ldots, z_m$	последовательность m примеров
$S \sim \mathcal{D}^m$	выборка $S = z_1, \ldots, z_m$ независимых и одинаково распределенных (с распределением $\mathcal{D}$) случайных величин
$P, \mathbb{E}$	вероятность и математическое ожидание случайной величины
$\mathbb{P}_{z \sim \mathcal{D}}[f(z)]$	$= \mathcal{D}(\{z: f(z) = \text{true}\})$ для $f : Z \to \{\text{true}, \text{false}\}$
$\mathbb{E}_{z \sim \mathcal{D}}[f(z)]$	математическое ожидание случайной величины $f : Z \to \mathbb{R}$
$N(\mu, C)$	нормальное распределение с математическим ожиданием μ и ковариацией C
$f'(x)$	производная функции $f : \mathbb{R} \to \mathbb{R}$ в точке x
$f''(x)$	вторая производная функции $f : \mathbb{R} \to \mathbb{R}$ в точке x
$\frac{\partial f(\mathbf{w})}{\partial w_i}$	частная производная функции $f: \mathbb{R}^d \to \mathbb{R}$ в точке $\mathbf{w}$ по w_i
$\nabla f(\mathbf{w})$	градиент функции $f : \mathbb{R}^d \to \mathbb{R}$ в точке $\mathbf{w}$
$\partial f(\mathbf{w})$	субдифференциал $f : \mathbb{R}^d \to \mathbb{R}$ в точке $\mathbf{w}$
$\min_{x \in C} f(x)$	$= \min\{f(x) : x \in C\}$ (минимальное значение f на множестве C)
$\max_{x \in C} f(x)$	$= \max\{f(x) : x \in C\}$ (максимальное значение f на множестве C)
$\operatorname{argmin}_{x \in C} f(x)$	множество $\{x \in C : f(x) = \min_{z \in C} f(z)\}$
$\operatorname{argmax}_{x \in C} f(x)$	множество $\{x \in C : f(x) = \max_{z \in C} f(z)\}$
log	натуральный логарифм

Вообще говоря, мы старались избегать асимптотической нотации. Но иногда все же используем ее, чтобы прояснить главные результаты. В частности, если имеются функции $f : \mathbb{R} \to \mathbb{R}_+$ и $g : \mathbb{R} \to \mathbb{R}_+$, то мы пишем $f = O(g)$, если существуют x_0 и $\alpha \in \mathbb{R}_+$ такие, что для любого $x > x_0$ имеет место неравенство $f(x) \leq \alpha g(x)$.

Мы пишем $f = o(g)$, если для любого $\alpha > 0$ существует x_0 такое, что для всех $x > x_0$ имеет место неравенство $f(x) \leq \alpha g(x)$. Мы пишем $f = \Omega(g)$, если существуют такие x_0 и $\alpha \in \mathbb{R}_+$, что для всех $x > x_0$ имеет место неравенство $f(x) \geq \alpha g(x)$. Нотация $f = \omega(g)$ определяется аналогично. Нотация $f = \Theta(g)$ означает, что $f = O(g)$ и $g = O(f)$. Наконец, нотация $f = \tilde{O}(g)$ означает, что существует $k \in \mathbb{N}$ такое, что $f(x) = O(g(x) \log^k(g(x)))$.

Скалярное произведение векторов $\mathbf{x}$ и $\mathbf{w}$ обозначается $\langle \mathbf{x}, \mathbf{w} \rangle$. Если явно не сказано, о каком векторном пространстве идет речь, то предполагается d-мерное евклидово пространство, и тогда $\langle \mathbf{x}, \mathbf{w} \rangle = \sum_{i=1}^{d} x_i w_i$. Евклидова (или ℓ_2) норма вектора $\mathbf{w}$ равна $\|\mathbf{w}\|_2 = \sqrt{\langle \mathbf{w}, \mathbf{w} \rangle}$. Мы опускаем нижний индекс в ℓ_2, если вид нормы понятен из контекста. Мы используем также другие нормы ℓ_p, $\|\mathbf{w}\|_p = (\sum_i |w_i|^p)^{1/p}$, и, в частности, $\|\mathbf{w}\|_1 = \sum_i |w_i|$ и $\|\mathbf{w}\|_\infty = \max_i |w_i|$.

Мы используем нотацию $\min_{x \in C} f(x)$ для обозначения минимального значения в множестве $\{f(x) : x \in C\}$. С точки зрения математики, следовало бы использовать $\inf_{x \in C} f(x)$ в случае, когда минимум не достигается. Но в контексте этой книги различие между минимумом и нижней гранью редко представляет интерес. Поэтому, чтобы не усложнять изложение, мы иногда употребляем нотацию min даже в тех случаях, когда inf было бы правильнее. Аналогичное замечание относится к max и sup.

Часть *I*

ОСНОВАНИЯ

Глава 2

МАЛЫЙ ВПЕРЕД

Начнем наш математический анализ с демонстрации того, насколько успешным может оказаться обучение в сравнительно простой ситуации. Представьте, что вы только что прибыли на маленький островок в Тихом океане. Очень скоро обнаружилось, что основным ингредиентом местных блюд является папайя. Но вы до сих пор ни разу не пробовали папайю. Необходимо научиться определять, будет ли папайя, предлагаемая на местном рынке, вкусной или нет. На основе прошлого опыта с другими фруктами вы приняли решение использовать два признака: цвет, который может быть темно-зеленым, оранжевым, красным и темно-коричневым, и мягкость – от каменно-твердой до кашицеобразной. Входными данными для выработки правила предсказания является выборка плодов папайи: вы смотрите на цвет и мягкость, затем пробуете и определяете, вкусным оказался образец или нет. Проанализируем эту задачу с точки зрения элементов, присутствующих в проблемах обучения.

Первый шаг – построить формальную модель, описывающую характеристики подобных задач.

2.1. Формальная модель – схема статистического обучения

Входные данные для обучаемого. В базовой схеме статистического обучения присутствуют следующие объекты.

- **Область образцов.** Произвольное множество $\mathcal{X}$. Это множество объектов, которые мы хотим помечать. Например, в задаче о классификации папайи, областью образцов является множество всех плодов папайи. Обычно точки этой области представляются вектором *признаков* (цвет и мягкость плода). Мы также будем называть точки $\mathcal{X}$ *образцами*, а саму область $\mathcal{X}$ – пространством образцов.
- **Область меток.** В этом обсуждении будем считать, что область меток состоит из двух элементов, обычно берется множество {0, 1} или {–1, +1}. Будем обозначать множество возможных меток $\mathcal{Y}$. В примере с папайей положим, что $\mathcal{Y}$ равно {0, 1}, где 1 означает, что плод вкусный, а 0 – что невкусный.

- **Обучающие данные.** $S = ((x_1, y_1) \ldots (x_m, y_m))$ – конечная последовательность пар из декартова произведения $\mathcal{X} \times \mathcal{Y}$, т. е. последовательность помеченных образцов. Это те входные данные, к которым имеет доступ обучаемый (в нашем примере набор плодов папайи и их цвет, мягкость и вкус). Такие помеченные примеры иногда называют *обучающими*. А S мы иногда будем называть обучающим набором[1].
- **Результат обучения.** От обучаемого требуется вывести *правило предсказания*, h: $\mathcal{X} \to \mathcal{Y}$. Эту функцию называют еще *предиктором*, *гипотезой* или *классификатором*. Предиктор можно использовать для предсказания метки новых точек из области образцов. В примере с папайей это то правило, которым будет руководствоваться обучаемый, чтобы предсказать вкус исследуемой папайи. Мы будем обозначать $A(S)$ гипотезу, которую алгоритм обучения A возвращает, получив на входе обучающую последовательность S.
- **Простая модель порождения данных.** Теперь объясним, как порождаются обучающие данные. Сначала предположим, что все образцы (плоды папайи) генерируются с помощью некоторого распределения вероятности (в данном случае представляющего окружающую среду). Обозначим $\mathcal{D}$ это распределение вероятностей на $\mathcal{X}$. Важно отметить: мы не предполагаем, что обучаемому что-то известно об этом распределении. Для тех типов задач обучения, которые мы обсуждаем, распределение может быть произвольным. Что касается меток, то в данном случае мы предполагаем, что существует некоторая «правильная» функция пометки f: $\mathcal{X} \to \mathcal{Y}$ такая, что $y_i = f(x_i)$ для любого i. В следующей главе это предположение будет ослаблено. Функция пометки неизвестна обучаемому. На самом деле, именно ее-то обучаемый и должен вывести. Таким образом, каждая пара обучающего набора S генерируется в два этапа: сначала из распределения $\mathcal{D}$ выбирается точка x_i, а затем она помечается функцией f.
- **Меры успеха.** *Ошибкой классификатора* мы называем вероятность предсказания неправильной метки для случайной точки, выбранной из вышеупомянутого распределения. То есть ошибка h – это вероятность выбрать из распределения $\mathcal{D}$ случайный образец x такой, что $h(x)$ не равно $f(x)$.
  Формально говоря, если дано подмножество области образцов[2] $A \subset \mathcal{X}$, то распределение вероятностей $\mathcal{D}$ назначает число $\mathcal{D}(A)$, определяющее, с какой вероятностью наблюдающий увидит точку $x \in A$. Во многих случаях мы будем называть A событием и выражать это с помощью функции $\pi : \mathcal{X} \to \{0, 1\}$: $A = \{x \in \mathcal{X}: \pi(x) = 1\}$. В данном случае для обозначения $\mathcal{D}(A)$ мы также применяем нотацию $\mathrm{P}_{x\sim\mathcal{D}}[\pi(x)]$.
  Определим ошибку правила предсказания $h : \mathcal{X} \to \mathcal{Y}$:

$$L_{\mathcal{D},f}(h) \stackrel{\text{def}}{=} \mathbb{P}_{x\sim\mathcal{D}}[h(x) \neq f(x)] \stackrel{\text{def}}{=} \mathcal{D}(\{x : h(x) \neq f(x)\}). \tag{2.1}$$

[1] Несмотря на название «набор» (или «множество»), S является последовательностью. В частности, один и тот же пример может встречаться в S несколько раз, а некоторые алгоритмы принимают во внимание порядок примеров в S.

[2] Строго говоря, следует быть аккуратнее и потребовать, чтобы A было членом некоторой σ-алгебры подмножеств $\mathcal{X}$, на которой определено $\mathcal{D}$. В следующей главе мы формально определим наши предположения об измеримости.

Иными словами, ошибка правила h – это вероятность случайно выбрать пример x, для которого $h(x) \neq f(x)$. Нижний индекс $(\mathcal{D}, f)$ означает, что ошибка измеряется относительно распределения вероятностей $\mathcal{D}$ и правильной функции пометки f. Мы будем опускать этот индекс, если он ясен из контекста. Для величины $L_{(\mathcal{D},f)}(h)$ встречаются и другие названия: ошибка обобщения, риск или истинная ошибка h – все они будут употребляться в этой книге как синонимы. Мы обозначаем ошибку буквой L, потому что рассматриваем ее как потерю (loss) обучаемого. Ниже мы обсудим и другие возможные определения потери.

- **Замечание об информации, доступной обучаемому.** Обучаемый ничего не знает о распределении $\mathcal{D}$ на области образцов и о функции пометки f. В нашем примере мы только что прибыли на остров и понятия не имеем, как распределены плоды папайи и как предсказать их вкус. Обучаемый взаимодействует с окружением только путем наблюдения обучающего набора.

В следующем разделе мы опишем простой алгоритм обучения в изложенной выше ситуации и проанализируем его производительность.

2.2. Минимизация эмпирического риска

Как уже было сказано, алгоритм обучения получает на входе обучающий набор S, выбранный из неизвестного распределения $\mathcal{D}$ и помеченный некоторой функцией f, а на выходе должен выдать предиктор h_S: $\mathcal{X} \to \mathcal{Y}$ (нижний индекс S подчеркивает тот факт, что результирующий предиктор зависит от S). Цель алгоритма – найти такую функцию h_S, которая минимизирует ошибку относительно неизвестных $\mathcal{D}$ и f.

Поскольку обучаемый не знает, что представляют собой $\mathcal{D}$ и f, истинная ошибка обучаемому неизвестна. Но он может вычислить полезную оценку ошибки – *ошибку обучения* классификатора на обучающем наборе:

$$L_S(h) \stackrel{\text{def}}{=} \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}, \tag{2.2}$$

где $[m] = \{1, \ldots, m\}$.

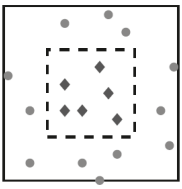
Эту величину также называют *эмпирической ошибкой* и *эмпирическим риском*.

Поскольку обучающий пример – это мгновенный снимок мира, доступный обучаемому, то имеет смысл поискать решение, которое хорошо работает на этих данных. Такая парадигма обучения – найти предиктор h, минимизирующий $L_S(h)$, – называется *минимизацией эмпирического риска* (Empirical Risk Minimization), или сокращенно ERM.

2.2.1. Не все коту масленица – переобучение

Правило ERM кажется совершенно естественным, но без должной аккуратности этот подход может доставить большие неприятности.

Чтобы продемонстрировать, в чем подвох, вернемся к задаче о том, как научиться предсказывать вкус папайи на основе мягкости и цвета. Рассмотрим набор данных, изображенный на следующем рисунке:



Предположим, что распределение вероятностей $\mathcal{D}$ таково, что образцы распределены равномерно внутри большого квадрата, а функция пометки f назначает образцу метку 1, если он находится во внутреннем квадрате, и 0 в противном случае. Площадь большого квадрата равна 2, а площадь внутреннего – 1. Рассмотрим такой предиктор:

$$h_S(x) = \begin{cases} y_i, \text{ если } \exists i \in [m] \text{ такое, что } x_i = x \\ 0 \text{ в противном случае} \end{cases}. \tag{2.3}$$

Этот предиктор может показаться несколько искусственным, но в упражнении 2.1 мы приведем его естественное представление в виде полиномов. Очевидно, что при любой выборке $L_S(h_S) = 0$, поэтому предиктор может быть найден алгоритмом ERM (это одна из гипотез с минимальным эмпирическим риском, никакой классификатор не может дать меньшей ошибки). С другой стороны, истинная ошибка любого классификатора, предсказывающего метку 1 только для конечного числа образцов, в данном случае равна 1/2. Таким образом, $L_{\mathcal{D}}(h_S) = 1/2$. Мы нашли предиктор, качество которого на обучающем наборе идеально, но при этом на «реальных» примерах он ведет себя очень плохо. Этот феномен называется *переобучением*. Интуитивно понятно, что переобучение имеет место, когда гипотеза «слишком хорошо» аппроксимирует данные (в обычной жизни человек, который дает детальные и безупречные объяснения каждому своему поступку, тоже вызывает подозрения).

2.3. Минимизация эмпирического риска с индуктивным смещением

Мы только что показали, что правило ERM может приводить к переобучению. Но чем отказываться от этой парадигмы, лучше подумаем, как ее можно исправить. Мы будем искать условия, при которых гарантируется, что ERM не дает переобучения, т. е. если ERM-предиктор показывает хорошее качество на обучающих данных, то с большой вероятностью он будет хорошо работать и на истинном распределении данных.

Общее решение заключается в том, чтобы применить правило ERM к ограниченному пространству поиска. Формально, обучаемый должен заранее (еще не видя данных) выбрать множество предикторов. Это множество называется

классом гипотез и обозначается $\mathcal{H}$. Каждый элемент $h \in \mathcal{H}$ – это функция, отображающая $\mathcal{X}$ в $\mathcal{Y}$. Для данного класса $\mathcal{H}$ и обучающей выборки S обучаемый $\text{ERM}_{\mathcal{H}}$ применяет правило ERM для выбора предиктора $h \in \mathcal{H}$ с наименьшей ошибкой на S. Формально

$$ERM_{\mathcal{H}}(S) \in \arg\min_{h \in \mathcal{H}} L_S(h),$$

где argmin обозначает множество гипотез из $\mathcal{H}$, которые доставляют минимум функции $L_S(h)$ на $\mathcal{H}$. Ограничив выбор обучаемого только предикторами из $\mathcal{H}$, мы смещаем алгоритм в сторону определенного множества предикторов. Такие ограничения часто называют *индуктивным смещением*. Поскольку ограничение выбирается еще до того, как обучаемый увидит обучающие данные, в идеале оно должно основываться на некотором априорном знании о проблеме. Например, в задаче о предсказании вкуса папайи можно включить в класс $\mathcal{H}$ только предикторы, определяемые осепараллельными прямоугольниками (в пространстве, где координатами являются цвет и мягкость). Позже мы увидим, что правило $\text{ERM}_{\mathcal{H}}$ над таким классом гарантированно не приводит к переобучению. С другой стороны, приведенный выше пример переобучения показывает, что выбор в качестве $\mathcal{H}$ класса предикторов, включающего все функции, которые назначают метку 1 конечному множеству точек из области определения, не гарантирует, что $\text{ERM}_{\mathcal{H}}$ не окажется переобученным.

Фундаментальный вопрос теории обучения заключается в нахождении классов гипотез, для которых правило $\text{ERM}_{\mathcal{H}}$ не приводит к переобучению. Мы будем изучать этот вопрос далее в этой книге.

Интуитивно понятно, что чем уже класс гипотез, тем надежнее мы защищены от переобучения, но при этом тем сильнее оказывается индуктивное смещение. Ниже мы еще вернемся к этому фундаментальному компромиссу.

2.3.1. Конечные классы гипотез

Проще всего наложить ограничение на размер класса (т. е. на количество предикторов h в $\mathcal{H}$). В этом разделе мы покажем, что если $\mathcal{H}$ – конечный класс, то $\text{ERM}_{\mathcal{H}}$ не приводит к переобучению при условии, что выборка достаточно велика (насколько велика, зависит от размера $\mathcal{H}$).

Разрешение выбирать правила предсказания только из конечного класса гипотез можно считать довольно мягким ограничением. Например, $\mathcal{H}$ может быть множеством всех таких предикторов, которые можно реализовать программой на C++, занимающей не более 109 бит кода. В примере с папайей мы упомянули класс осепараллельных прямоугольников. Это бесконечный класс, но если взять дискретное представление вещественных чисел, скажем, 64-разрядное представление с плавающей точкой, то класс гипотез станет конечным.

Теперь проанализируем качество правила обучения $\text{ERM}_{\mathcal{H}}$ в предположении конечности класса $\mathcal{H}$. Для обучающей выборки S, размеченной функцией $f : \mathcal{X} \to \mathcal{Y}$, обозначим h_S результат применения $\text{ERM}_{\mathcal{H}}$ к S, а именно

$$h_S \in \arg\min_{h \in \mathcal{H}} L_S(h). \tag{2.4}$$

В этой главе мы сделаем следующее упрощающее предположение (которое будет ослаблено в следующей главе).

Определение 2.1 (предположение о реализуемости). Существует $h^* \in \mathcal{H}$ такая, что $L_{(\mathcal{D},f)}(h^*) = 0$. Отметим, что из этого предположения следует, что с вероятностью 1 для случайной выборки S из распределения $\mathcal{D}$, элементы которой помечены функцией f, имеем $L_S(h^*) = 0$.

Из предположения о реализуемости вытекает, что для каждой ERM-гипотезы мы имеем[1] $L_S(h_S) = 0$. Однако нас интересует *истинный* риск гипотезы h_S, $L_{(\mathcal{D},f)}(h_S)$, а не ее эмпирический риск.

Очевидно, что любая гарантия ошибки относительно истинного распределения $\mathcal{D}$, которую можно дать для алгоритма, имеющего доступ только к выборке S, должна зависеть от соотношения между $\mathcal{D}$ и S. В статистическом машинном обучении принимается предположение, что при порождении обучающего набора S точки выбираются из распределения $\mathcal{D}$ независимо. Формально:

Предположение о независимости и одинаковом распределении. Примеры в обучающем наборе независимы и имеют одинаковое распределение $\mathcal{D}$. Это означает, что каждый пример x_i, вошедший в S, выбирается из $\mathcal{D}$ независимо от остальных и затем помечается функцией пометки f. Будем обозначать это предположение $S \sim \mathcal{D}^m$, где m – размер S, а $\mathcal{D}^m$ обозначает распределение вероятностей на кортежах длины m, индуцированное применением $\mathcal{D}$ для выборки каждого элемента кортежа независимо от остальных.

Интуитивно обучающий набор S можно представлять себе как окно, через которое обучаемый получает частичную информацию о полном распределении $\mathcal{D}$ и о функции пометки f. Чем больше выборка, тем вероятнее, что она точнее отражает распределение и функцию пометки, с помощью которых получена.

Поскольку $L_{(\mathcal{D},f)}(h_S)$ зависит от обучающего набора S, а этот обучающий набор выбирается случайным процессом, то выбор предиктора h_S и, следовательно, риск $L_{(\mathcal{D},f)}(h_S)$ носит случайный характер. Формально мы говорим, что это случайная величина. Не стоит ожидать, что S обязательно направит обучаемого в сторону хорошего (с точки зрения $\mathcal{D}$) классификатора, поскольку всегда остается вероятность, что выборочные обучающие данные будут совершенно нерепрезентативны для распределения $\mathcal{D}$. Если вернуться к примеру с папайей, то всегда имеется (небольшой) шанс, что все плоды, которые мы пробовали, окажутся невкусными, несмотря на то, что, скажем, 70% плодов папайи на острове хороши на вкус. В таком случае $\text{ERM}_{\mathcal{H}}(S)$ может оказаться постоянной функцией, которая помечает любую папайю как «невкусную» (и будет иметь ошибку 70% относительно истинного распределения папайи на острове). Поэтому мы учитываем *вероятность* выборки обучающего набора, для которого величина $L_{(\mathcal{D},f)}(h_S)$ не слишком велика. Обычно вероятность получить нерепрезентативную выборку обозначается δ, а $(1 - \delta)$ называется *параметром уверенности* предсказания.

[1] Строго говоря, это имеет место с вероятностью 1. Но, чтобы упростить изложение, мы будем опускать слова «с вероятностью 1».

Кроме того, поскольку мы не можем гарантировать идеально точное предсказание метки, вводится еще один параметр качества предсказания – *верность* (accuracy), обычно обозначаемый ϵ. Событие $L_{(\mathcal{D},f)}(h_S) > \epsilon$ интерпретируется как неудачное обучение, а если $L_{(\mathcal{D},f)}(h_S) \leq \epsilon$, то мы считаем выход алгоритма приблизительно правильным предиктором. Поэтому (если зафиксировать некоторую функцию пометки $f : \mathcal{X} \to \mathcal{Y}$) нас интересует верхняя граница вероятности того, что на выборке из m образцов обучаемый потерпит неудачу. Формально пусть $S|_x = (x_1, \ldots, x_m)$ – образцы в обучающем наборе. Мы хотим ограничить сверху величину

$$\mathcal{D}^m(\{S|_x : L_{(\mathcal{D},f)}(h_S) > \epsilon\}).$$

Обозначим $\mathcal{H}_B$ множество «плохих» гипотез, т. е.

$$\mathcal{H}_B = \{h \in \mathcal{H} : L_{(\mathcal{D},f)}(h) > \epsilon\}.$$

Кроме того, обозначим

$$M = \{S|_x : \exists h \in \mathcal{H}_B, L_S(h) = 0\}$$

множество дезориентирующих выборок. Точнее, для каждого $S|_x \in M$ существует «плохая» гипотеза $h \in \mathcal{H}_B$, которая выглядит «хорошей» на $S|_x$. Теперь вспомним, что мы хотели ограничить вероятность события $L_{(\mathcal{D},f)}(h_S) > \epsilon$. Но поскольку из предположения о реализуемости следует, что $L_S(h_S) = 0$, то событие $L_{(\mathcal{D},f)}(h_S) > \epsilon$ может произойти, только если для некоторого $h \in \mathcal{H}_B$ имеет место $L_S(h) = 0$. Иными словами, это событие произойдет, только если наша выборка принадлежит множеству дезориентирующих выборок M. Формально мы только что доказали, что

$$\{S|_x : L_{(\mathcal{D},f)}(h_S) > \epsilon\} \leq M.$$

Отметим, что M можно переписать в виде

$$M = \bigcup_{h \in \mathcal{H}_B} \{S|_x : L_S(h) = 0\}. \tag{2.5}$$

Отсюда

$$\mathcal{D}^m(\{S|_x : L_{(\mathcal{D},f)}(h_S) > \epsilon\}) \leq \mathcal{D}^m(M) = \mathcal{D}^m(\cup_{h \in \mathcal{H}_B}\{S|_x : L_S(h) = 0\}). \tag{2.6}$$

Далее мы ограничим сверху правую часть предыдущего неравенства, воспользовавшись *границей объединения* – базовым свойством вероятностей.

Лемма 2.2 (граница объединения). *Для любых двух множеств A и B и распределения $\mathcal{D}$ справедливо неравенство*

$$\mathcal{D}(A \cup B) \leq \mathcal{D}(A) + \mathcal{D}(B).$$

Применение этой леммы к правой части (2.6) дает

$$\mathcal{D}^m(\{S|_x : L_{(\mathcal{D},f)}(h_S) > \epsilon\}) \leq \sum_{h \in \mathcal{H}_B} \mathcal{D}^m(\{S|_x : L_S(h) = 0\}). \tag{2.7}$$

Далее ограничим каждое слагаемое в правой части этого неравенства. Зафиксируем какую-нибудь «плохую» гипотезу $h \in \mathcal{H}_B$. Событие $L_S(h) = 0$ эквивалент-

но событию $\forall i h(x_i) = f(x_i)$. Поскольку примеры в обучающем наборе независимы и одинаково распределены, то

$$\mathcal{D}^m(\{S|_x : L_S(h) = 0\}) = \mathcal{D}^m(\{S|_x : \forall i, h(x_i) = f(x_i)\}).$$
$$= \prod_{i=1}^{m} \mathcal{D}(\{x_i : h(x_i) = f(x_i)\}). \quad (2.8)$$

Для каждого элемента обучающего набора в отдельности имеем

$$\mathcal{D}(\{x_i : h(x_i) = y_i\}) = 1 - L_{(\mathcal{D},f)}(h) \leq 1 - \epsilon,$$

где последнее неравенство следует из того факта, что $h \in \mathcal{H}_B$. Объединяя это равенство с равенством (2.8) и воспользовавшись тем, что $1 - \epsilon \leq e^{-\epsilon}$, получаем, что для каждой гипотезы $h \in \mathcal{H}_B$

$$\mathcal{D}^m(\{S|_x : L_S(h) = 0\}) \leq (1 - \epsilon)^m \leq e^{-\epsilon m}. \quad (2.9)$$

Объединяя с неравенством (2.7), приходим к выводу, что

$$\mathcal{D}^m(\{S|_x : L_{(\mathcal{D},f)}(h_S) > \epsilon\}) \leq |\mathcal{H}_B| e^{-\epsilon m} \leq |\mathcal{H}| e^{-\epsilon m}.$$

На рис. 2.1 приведена графическая иллюстрация, объясняющая, как мы воспользовались границей объединения.

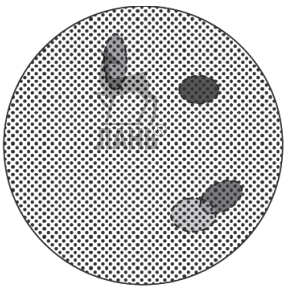


Рис. 2.1. Каждая точка внутри большой окружности представляет один из возможных кортежей образцов длины m. Каждый окрашенный овал представляет множество «дезориентирующих» кортежей длины m для некоторого «плохого» предиктора $h \in \mathcal{H}_B$. Правило ERM потенциально может привести к переобучению всякий раз, как получает дезориентирующий обучающий набор S. То есть для некоторого $h \in \mathcal{H}_B$ мы имеем $L_S(h) = 0$. Неравенство (2.9) гарантирует, что для каждой плохой гипотезы $h \in \mathcal{H}_B$ доля дезориентирующих обучающих наборов не превосходит $(1 - \epsilon)^m$. В частности, чем больше m, тем меньше каждый из окрашенных овалов. Граница объединения формализует тот факт, что площадь области, представляющей обучающие наборы, дезориентирующие относительно некоторой гипотезы $h \in \mathcal{H}_B$ (т. е. обучающие наборы, принадлежащие M), не превосходит сумму площадей окрашенных овалов. Поэтому она ограничена сверху произведением $|\mathcal{H}_B|$ на максимальную площадь окрашенного овала. Никакая выборка S вне окрашенных овалов не может привести к переобучению, если применяется правило ERM

Следствие 2.3. *Пусть $\mathcal{H}$ – конечный класс гипотез. Пусть $\delta \in (0, 1)$, и $\epsilon > 0$, а m – целое число, удовлетворяющее условию*

$$m \geq \frac{\log(|H|/\delta)}{\epsilon}.$$

Тогда для любой функции пометки f и для любого распределения $\mathcal{D}$, удовлетворяющих предположению о реализуемости (т. е. для некоторой $h \in \mathcal{H}$ верно, что $L_{(\mathcal{D},f)}(h) = 0$) для случайной выборки m независимых и одинаково распределенных образцов S с вероятностью не ниже $1 - \delta$ для каждой ERM-гипотезы h_S будет справедливо неравенство

$$L_{(\mathcal{D},f)}(h_S) \leq \epsilon.$$

ЛАНЬ®

Это следствие утверждает, что для достаточно больших m правило $\mathrm{ERM}_{\mathcal{H}}$ над конечным классом гипотез *вероятно* (с параметром уверенности $1 - \delta$) *почти* (с ошибкой не более ϵ) корректно. В следующей главе мы формально определим модель вероятно почти корректного (Probably Approximately Correct – PAC) обучения.

2.4. Упражнения

2.1. Переобучение полиномиальной аппроксимации. Мы видели, что предиктор, определенный формулой (2.3), приводит к переобучению. Хотя этот предиктор кажется совершенно неестественным, цель настоящего упражнения – показать, что его можно описать срезанным полиномом. Докажите, что для заданного обучающего набора $S = \{(\mathbf{x}_i, f(\mathbf{x}_i))\}_{i=1}^m \subseteq (\mathbb{R}^d \times \{0, 1\})^m$ существует полином p_S такой, что $h_S(\mathbf{x}) = 1$ тогда и только тогда, когда $p_S(\mathbf{x}) \geq 0$, где h_S определено формулой (2.3). Отсюда следует, чтобы обучение класса всех срезанных полиномов по правилу ERM может приводить к переобучению.

2.2. Пусть $\mathcal{H}$ – класс бинарных классификаторов над областью образцов $\mathcal{X}$. Пусть $\mathcal{D}$ – неизвестное распределение над $\mathcal{X}$, и пусть f – целевая гипотеза в $\mathcal{H}$. Зафиксируем некоторую гипотезу $h \in \mathcal{H}$. Покажите, что математическое ожидание $L_S(h)$ на выборках $S|_x$ равно $L_{(\mathcal{D},f)}(h)$, т. е.

$$\mathop{\mathbb{E}}_{S|_x \sim \mathcal{D}^m}[L_S(h)] = L_{(\mathcal{D},f)}(h).$$

ЛАНЬ

2.3. Осепараллельные прямоугольники. Плоский осепараллельный классификатор назначает точке метку 1 тогда и только тогда, когда она находится внутри некоторого осепараллельного прямоугольника на плоскости. Точнее, если даны вещественные числа $a_1 \leq b_1$, $a_2 \leq b_2$, то классификатор $h_{(a_1,b_1,a_2,b_2)}$ определяется формулой

$$h_{(a_1,b_1,a_2,b_2)}(x_1, x_2) = \begin{cases} 1, & \text{если } a_1 \leq x_1 \leq b_1 \text{ и } a_2 \leq x_2 \leq b_2 \\ 0, & \text{в противном случае} \end{cases}. \qquad (2.10)$$

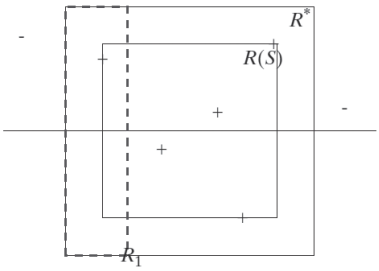


Рис. 2.2. Осепараллельные прямоугольники

Класс всех осепараллельных прямоугольников на плоскости определяется следующим образом:

$$\mathcal{H}^2_{\text{rec}} = \{h_{(a_1, b_1, a_2, b_2)} : a_1 \leq b_1 \text{ и } a_2 \leq b_2\}.$$

Отметим, что этот класс гипотез бесконечен. В этом упражнении мы считаем справедливым предположение о реализуемости.

1. Пусть A – алгоритм, который возвращает наименьший прямоугольник. Покажите, что A дает минимальный эмпирический риск.
2. Покажите, что если A получает на входе обучающий набор размера $\geq \frac{4\log(4/\delta)}{\epsilon}$, то с вероятностью не меньше $1 - \delta$ он возвращает гипотезу с ошибкой не больше ϵ.
   Указание. Зафиксируем некоторое распределение $\mathcal{D}$ на $\mathcal{X}$, и пусть $R^* = R(a^*_1, b^*_1, a^*_2, b^*_2)$ – прямоугольник, генерирующий метки, а f – соответствующая гипотеза. Пусть $a_1 \geq a^*_1$ – такое число, что масса вероятности (относительно $\mathcal{D}$) прямоугольника $R_1 = R(a^*_1, a_1, a^*_2, b^*_2)$ в точности равна $\epsilon/4$. Аналогично, пусть b_1, a_2, b_2 – такие числа, что массы вероятности прямоугольников $R2 = R(b_1, b^*_1, a^*_2, b^*_2)$, $R3 = R(a^*_1, b^*_1, a^*_2, a_2)$, $R4 = R(a^*_1, b^*_1, b_2, b^*_2)$ в точности равны $\epsilon/4$. Пусть $R(S)$ – прямоугольник, возвращенный алгоритмом A. См. рис. 2.2.
   - Покажите, что $R(S) \subseteq R^*$.
   - Покажите, что если S содержит (положительные) примеры в каждом из прямоугольников R_1, R_2, R_3, R_4, то ошибка гипотезы, возвращенной A, не превышает ϵ.
   - Для каждого $i \in \{1, \ldots, 4\}$ найдите верхнюю границу вероятности того, что S не содержит примеров из R_i.
   - Воспользуйтесь границей объединения, чтобы завершить доказательство.
3. Повторите предыдущее упражнение для класса осепараллельных параллелепипедов в $\mathbb{R}^d$.
4. Покажите, что время работы вышеупомянутого алгоритма A полиномиально зависит от d, $1/\epsilon$ и $\log(1/\delta)$.

ФОРМАЛЬНАЯ МОДЕЛЬ ОБУЧЕНИЯ

В этой главе мы определим формальную модель обучения, главную в этой книге, – модель PAC и ее обобщения. Другие подходы к обучаемости будут рассмотрены в главе 7.

ЛАНЬ®

3.1. Вероятно почти корректное обучение

В предыдущей главе мы показали, что если имеется конечный класс гипотез и правило ERM относительно этого класса применяется к достаточно большой обучающей выборке (размер которой не зависит ни от истинного распределения, ни от функции пометки), то результирующая гипотеза, вероятно, будет почти корректной. Теперь определим, что такое *вероятно почти корректное* (Probably Approximately Correct – PAC) обучение в более общем виде.

Определение 3.1 (PAC-обучаемость). Класс гипотез $\mathcal{H}$ называется PAC-обучаемым, если существует функция $m_{\mathcal{H}} : (0, 1)^2 \to \mathbb{N}$ и алгоритм обучения, обладающий следующим свойством: для любых $\epsilon, \delta \in (0, 1)$, для любого распределения $\mathcal{D}$ над $\mathcal{X}$ и для любой функции пометки $f : \mathcal{X} \to \{0, 1\}$, если выполняется предположение о реализуемости относительно $\mathcal{H}$, $\mathcal{D}$, f, то при подаче на вход алгоритма обучения $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ независимых примеров, выбранных из распределения $\mathcal{D}$ и помеченных функцией f, алгоритм вернет гипотезу h такую, что с вероятностью не меньше $1 - \delta$ (для случайной выборки обучающих примеров) $L_{(\mathcal{D},f)}(h) \leq \epsilon$.

В определение вероятно почти корректной обучаемости входит два параметра аппроксимации. Параметр верности (accuracy) ϵ определяет, насколько выходной классификатор далек от оптимального (он соответствует фразе «почти корректное»), а параметр уверенности δ показывает, насколько вероятно, что классификатор будет удовлетворять требованию верности (это часть «вероятно» в определении PAC). При той модели доступа к данным, которую мы исследуем, эти аппроксимации неизбежны. Поскольку обучающий набор генерируется случайно, всегда имеется небольшой шанс, что он окажется неинформативным (например, будет содержать всего одну точку из области образцов, которая выбирается снова и снова). Но, даже если нам повезло и мы получили набор, точно представляющий $\mathcal{D}$, все равно это не более чем конечная выборка, которая могла не уловить некоторые тонкие детали. Параметр верности позволяет «прощать» небольшие ошибки обученного классификатора.

ЛАНЬ

Выборочная сложность

Функция $m_{\mathcal{H}} : (0, 1)^2 \to \mathbb{N}$ определяет *выборочную сложность* обучения $\mathcal{H}$, т. е. сколько примеров необходимо, чтобы гарантировать вероятно почти корректное решение. Выборочная сложность – это функция от параметров верности (ϵ) и уверенности (δ). Она также зависит от свойств класса гипотез $\mathcal{H}$: например, мы показали, что для конечного класса выборочная сложность зависит от логарифма размера $\mathcal{H}$.

Отметим, что если класс $\mathcal{H}$ является PAC-обучаемым, то существует много функций $m_{\mathcal{H}}$, удовлетворяющих требованиям, сформулированным в определении PAC-обучаемости. Поэтому в определении выборочной сложности обучения $\mathcal{H}$ уточним, что это «минимальная функция» в том смысле, что для любых ϵ, δ значение $m_{\mathcal{H}}(\epsilon, \delta)$ – это минимальное целое число, удовлетворяющее требованиям PAC-обучаемости с верностью ϵ и уверенностью δ.

Теперь вспомним, к какому заключению мы пришли в ходе анализа конечных классов гипотез в предыдущей главе. Его можно переформулировать следующим образом.

Следствие 3.2. *Любой конечный класс гипотез является PAC-обучаемым с выборочной сложностью*

$$m_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \right\rceil.$$

Существуют также бесконечные обучаемые классы (см., например, упражнение 3.3). Ниже мы увидим, что PAC-обучаемость класса определяется не конечностью, а комбинаторной характеристикой, называемой *VC-размерностью*.

3.2. Более общая модель обучения

Описанную выше модель легко можно обобщить на более широкий круг задач обучения. Мы рассмотрим обобщения в двух направлениях.

Отказ от предположения о реализуемости

Мы требовали, чтобы алгоритм обучения успешно завершался для пары – распределение данных $\mathcal{D}$ и функция пометки f – при условии, что выполнено предположение о реализуемости. В практических задачах обучения это предположение может оказаться слишком сильным (действительно, можно ли гарантировать, что в пространстве «цвет–твердость» существует прямоугольник, который *полностью определяет*, какие папайи вкусные?). В следующем подразделе мы опишем *агностическую* модель, в которой предположение о реализуемости отброшено.

Проблемы обучения, не сводящиеся к бинарной классификации

До сих пор мы обсуждали предсказание бинарной метки заданного примера (например, вкусный или невкусный). Но многие проблемы обучения формули-

руются по-другому. Например, иногда нужно предсказать вещественное число (скажем, температуру завтра в 9 вечера) или метку, выбираемую из конечного множества (например, главную тему номера в завтрашней газете). Оказывается, что наш анализ обучения без труда обобщается на такие и многие другие сценарии, нужно только взять другую функцию потерь. Мы обсудим этот вопрос в разделе 3.2.2 ниже.

ЛАНЬ®

3.2.1. Отказ от предположения о реализуемости – агностическое PAC-обучение

Более реалистичная модель распределения, порождающего данные

Напомним, что предположение о реализуемости требует, чтобы существовала гипотеза $h^* \in \mathcal{H}$ такая, что $\mathbb{P}_{x\sim\mathcal{D}}[h^*(x) = f(x)] = 1$. Во многих практических проблемах это предположение не выполняется. Кроме того, возможно, ближе к реальности не предполагать, что метки полностью определены признаками, наблюдаемыми для входных примеров (в случае папайи очень может статься, что два плода одинакового цвета и мягкости на вкус сильно различаются). Далее мы ослабим предположение о реализуемости, заменив «целевую функцию пометки» более гибким понятием – распределением, порождающим метки данных.

Начиная с этого момента, $\mathcal{D}$ будет обозначать распределение вероятностей на $\mathcal{X} \times \mathcal{Y}$, где, как и раньше, $\mathcal{X}$ – область образцов, а $\mathcal{Y}$ – область меток (обычно берется $\mathcal{Y} = \{0, 1\}$). Таким образом, $\mathcal{D}$ – *совместное распределение* точек из областей образцов и меток. Можно считать, что такое распределение состоит из двух частей: распределение $\mathcal{D}_x$ непомеченных точек (иногда называемое *маргинальным распределением*) и *условное* распределение вероятностей меток для каждой точки, $\mathcal{D}((x, y)|x)$. В примере с папайей $\mathcal{D}_x$ определяет вероятность встретить папайю, цвет и твердость которой попадают в определенную область пространства «цвет–твердость», а условное распределение дает вероятность того, что папайя, цвет и твердость которой представлены точкой x, окажется вкусной. При такой модели две папайи с одинаковым цветом и твердостью могут принадлежать разным вкусовым категориям.

Еще раз об эмпирической и истинной ошибке

Имея распределение вероятностей $\mathcal{D}$ на $\mathcal{X} \times \mathcal{Y}$, можно измерить, с какой вероятностью h допускает ошибку, когда помеченные точки случайным образом выбираются из $\mathcal{D}$. Переопределим истинную ошибку (или риск) правила предсказания h следующим образом:

ЛАНЬ®

$$L_{\mathcal{D}}(h) \overset{\text{def}}{=} \mathbb{P}_{(x,y)\sim\mathcal{D}}[h(x) \neq y] \overset{\text{def}}{=} \mathcal{D}(\{(x, y) : h(x) \neq y\}). \tag{3.1}$$

Мы ищем предиктор h, для которого эта ошибка минимальна. Однако обучаемому неизвестно распределение $\mathcal{D}$, порождающее данные. Единственное, к чему обучаемый имеет доступ, – это обучающие данные S. Определение эмпирического риска остается прежним:

$$L_S(h) \stackrel{\text{def}}{=} \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}.$$

Зная S, обучаемый может вычислить $L_S(h)$ для любой функции $h : \mathcal{X} \to \{0, 1\}$. Отметим, что $L_S(h) = L_{\mathcal{D}(\text{равномерное на } S)}(h)$.

ЛАНЬ®

Цель

Мы хотим найти такую гипотезу $h: \mathcal{X} \to \mathcal{Y}$, которая минимизирует (быть может, приблизительно) истинный риск $L_{\mathcal{D}}(h)$.

Оптимальный байесовский предиктор

Если задано произвольное распределение вероятностей $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$, то наилучшей функцией из $\mathcal{X}$ в $\{0, 1\}$, предсказывающей метку, будет

$$f_{\mathcal{D}}(x) = \begin{cases} 1, & \text{если } \mathbb{P}[y = 1 \mid x] \geq 1/2 \\ 0, & \text{в противном случае} \end{cases}.$$

Легко проверить (см. упражнение 3.7), что для любого распределения вероятностей $\mathcal{D}$ оптимальный байесовский предиктор $f_{\mathcal{D}}$ действительно оптимален в том смысле, что никакой другой классификатор $g : \mathcal{X} \to \{0, 1\}$ не дает меньшей ошибки. Таким образом, для любого классификатора g имеем $L_{\mathcal{D}}(f_{\mathcal{D}}) \leq L_{\mathcal{D}}(g)$.

К сожалению, поскольку мы не знаем $\mathcal{D}$, мы не можем воспользоваться этим оптимальным предиктором. Обучаемый имеет доступ только к обучающей выборке. Теперь мы можем дать формальное определение агностической PAC-обучаемости, которое представляет собой естественное обобщение PAC-обучаемости на более реалистичную конфигурацию обучения, без предположения о реализуемости.

ЛАНЬ®

Очевидно, нельзя надеяться, что алгоритм обучения найдет гипотезу с ошибкой меньше минимально возможной, т. е. той, что дает байесовский предиктор. Кроме того, как мы впоследствии докажем, без априорных предположений о порождающем данные распределении никакой алгоритм не может гарантировать нахождение предиктора, который был бы так же хорош, как оптимальный байесовский. Вместо этого мы требуем, чтобы алгоритм обучения нашел предиктор, который дает ошибку, ненамного большую, чем наилучшая возможная ошибка предиктора из некоторого заданного эталонного класса гипотез. Конечно, сила этого требования зависит от выбора класса гипотез.

Определение 3.3 (агностическая PAC-обучаемость). Класс гипотез $\mathcal{H}$ называется агностически PAC-обучаемым, если существует функция $m_{\mathcal{H}} : (0, 1)^2 \to \mathbb{N}$ и алгоритм обучения, обладающий следующим свойством: для любых ϵ, $\delta \in (0, 1)$ и для любого распределения $\mathcal{D}$ на $\mathcal{X} \times \mathcal{Y}$ при подаче на вход алгоритма обучения $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ независимых примеров, выбранных из распределения $\mathcal{D}$, алгоритм возвращает гипотезу h такую, что с вероятностью не меньше $1 - \delta$ (для случайной выборки m обучающих примеров)

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon.$$

Очевидно, что если предположение о реализуемости выполняется, то агностическое PAC-обучение дает те же гарантии, что обычное. В этом смысле агностическое PAC-обучение обобщает определение PAC-обучения. Если же предположение о реализуемости не выполняется, то никакой обучаемый не может гарантировать сколь угодно малую ошибку. Тем не менее, при таком определении агностического PAC-обучения обучаемый все же может объявить об успехе, если ошибка не намного больше наилучшей ошибки, достижимой предикторами из класса $\mathcal{H}$. В этом состоит отличие от PAC-обучения, при котором от обучаемого требуется, чтобы ошибка была мала абсолютно, а не относительно наилучшей ошибки, достижимой классом гипотез.

3.2.2. Круг моделируемых проблем обучения

Далее мы обобщим нашу модель, так чтобы ее можно было применить к широкому кругу задач обучения. Рассмотрим некоторые примеры таких задач.

- **Многоклассовая классификация.** Классификация не обязана быть бинарной. Возьмем, к примеру, классификацию документов. Мы хотим разработать программу, которая будет классифицировать документы по тематике (новости, спорт, биология, медицина и т. д.). Алгоритм обучения для такой задачи будет иметь доступ к примерам правильно классифицированных документов и в результате должен вывести программу, которая принимает новый документ и назначает ему тему. В данном случае *областью определения* является множество всех потенциальных документов. Как и раньше, мы обычно представляем документ набором *признаков*, в который могут входить ключевые слова и другие релевантные характеристики, например, размер документа и его происхождение. *Множеством меток* в этой задаче является набор возможных тем (т. е. $\mathcal{Y}$ будет некоторым большим конечным множеством). После того, как область определения и множество меток определены, остальные компоненты системы выглядят так же, как в примере с предсказанием вкуса папайи. *Обучающая выборка* будет конечной последовательностью пар (вектор признаков, метка), результатом обучения – функция, отображающая область образцов в область меток, а в качестве показателя успеха можно использовать вероятность на множестве пар (документ, тема) события, заключающегося в предсказании неверной метки.
- **Регрессия.** В этой задаче требуется найти в данных простой *паттерн* – функциональную зависимость между компонентами данных $\mathcal{X}$ и $\mathcal{Y}$. Например, найти линейную функцию, которая точнее всего предсказывает вес ребенка при рождении на основе данных УЗИ об окружности головы, окружности живота и длине бедра. Здесь область образцов $\mathcal{X}$ – подмножество $\mathbb{R}^3$ (три измерения УЗИ), а область «меток» $\mathcal{Y}$ – это множество вещественных чисел вес в граммах). В этом контексте $\mathcal{Y}$ более естественно называть *целевым множеством* (target set). Обучающие данные и результат обучения не изменились – соответственно конечная последовательность пар (x, y) и функция, отображающая $\mathcal{X}$ в $\mathcal{Y}$. Однако показатель успешности стал другим. Качество гипотезы, функции $h : \mathcal{X} \to \mathcal{Y}$, можно оценить с по-

мощью *математического ожидания разности квадратов* истинных меток и предсказанных значений.

$$L_{\mathcal{D}}(h) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} (h(x)-y)^2. \tag{3.2}$$

Чтобы охватить как можно более широкий круг задач обучения, мы обобщим формальное определение показателя успешности.

Обобщенные функции потерь

Пусть дано произвольное множество $\mathcal{H}$ (которое играет роль множества гипотез, или моделей) и некоторое множество примеров Z. Будем называть *функцией потерь* произвольную функцию ℓ, отображающую $\mathcal{H} \times Z$ в множество неотрицательных вещественных чисел, $\ell : \mathcal{H} \times Z \to \mathbb{R}_+$.

Отметим, что в задачах предсказания $Z = \mathcal{X} \times \mathcal{Y}$. Но наше понятие функции потерь относится не только к задачам предсказания, поэтому Z может быть любой областью, из которой выбираются примеры (так, в задачах обучения без учителя, в т. ч. описанной в главе 22, Z не является декартовым произведением области образцов и области меток).

Теперь определим *функцию риска* как математическое ожидание потери классификатора $h \in \mathcal{H}$ относительно распределения вероятностей $\mathcal{D}$ на Z:

$$L_{\mathcal{D}}(h) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{z\sim\mathcal{D}} [\ell(h,z)]. \tag{3.3}$$

То есть мы рассматриваем математическое ожидание потери на объектах z, случайно выбранных из распределения $\mathcal{D}$. Аналогично определим *эмпирический риск* как математическое ожидание потери на заданной выборке $S = (z_1, \ldots, z_m) \in Z_m$:

$$L_S(h) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{i=1}^{m} \ell(h, z_i). \tag{3.4}$$

В ранее встречавшихся примерах задач классификации и регрессии функции потерь имеют вид:

- **Бинарная функция потерь.** Здесь случайная величина z пробегает множество пар $\mathcal{X} \times \mathcal{Y}$, а функция потерь равна

$$\ell_{0-1}(h,(x,y)) \stackrel{\text{def}}{=} \begin{cases} 0, & \text{если } h(x) = y \\ 1, & \text{если } h(x) \neq y \end{cases}. \tag{3.4}$$

  Эта функция потерь используется в проблемах бинарной и многоклассовой классификации. Отметим, что если случайная величина α принимает значения {0, 1}, то $\mathbb{E}_{\alpha\sim D}[\alpha] = \mathrm{P}_{\alpha\sim D}[\alpha = 1]$. Следовательно, для этой функции потерь определения (3.1) и (3.3) функции риска $L_{\mathcal{D}}(h)$ совпадают.
- **Квадратичная функция потерь.** Здесь случайная величина z пробегает множество пар $\mathcal{X} \times \mathcal{Y}$, а функция потерь равна

$$\ell_{\text{sq}}(h,(x,y)) \stackrel{\text{def}}{=} (h(x)-y)^2.$$

Эта функция потерь используется в проблемах регрессии.

Позже мы встретим другие примеры полезных функций потерь.

Подводя итоги, дадим формальное определение агностической PAC-обучаемости для функций потерь общего вида.

Определение 3.4 (агностическая PAC-обучаемость для функций потерь общего вида). Класс гипотез $\mathcal{H}$ называется агностически PAC-обучаемым относительно множества Z и функции потерь $\ell : \mathcal{H} \times Z \to \mathbb{R}_+$, если существует функция $m_{\mathcal{H}} : (0, 1)^2 \to \mathbb{N}$ и алгоритм обучения, обладающий следующим свойством: для любых $\epsilon, \delta \in (0, 1)$ и для любого распределения $\mathcal{D}$ на Z при подаче на вход алгоритма обучения $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ независимых примеров, выбранных из распределения $\mathcal{D}$, алгоритм возвращает гипотезу $h \in \mathcal{H}$ такую, что с вероятностью не меньше $1 - \delta$ (для случайной выборки m обучающих примеров)

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon,$$

где $L_{\mathcal{D}}(h) = \mathbb{E}_{z \sim \mathcal{D}}[\ell(h, z)]$.

Замечание 3.1 (замечание об измеримости*). В приведенном выше определении мы для любого $h \in \mathcal{H}$ рассматриваем функцию $\ell(h, \cdot)$: $Z \to \mathbb{R}_+$ как случайную величину и определяем $L_{\mathcal{D}}(h)$ как математическое ожидание этой величины. Для этого необходимо потребовать, чтобы функция $\ell(h, \cdot)$ была измеримой. Точнее, мы предполагаем, что существует σ-алгебра подмножеств Z, на которой определено распределение вероятностей $\mathcal{D}$ и что прообраз каждого начального отрезка $\mathbb{R}_+$ принадлежит этой σ-алгебре. В частном случае бинарной классификации с бинарной функцией потерь σ-алгебра определена на подмножествах $\mathcal{X} \times \{0, 1\}$, и наше предположение о ℓ эквивалентно предположению о том, что для любого h множество $\{(x, h(x)) : x \in \mathcal{X}\}$ принадлежит этой σ-алгебре.

Замечание 3.2 (сравнение собственного и независимого от представления обучения*). В приведенном выше определении мы потребовали, чтобы алгоритм возвращал гипотезу из $\mathcal{H}$. В некоторых ситуациях $\mathcal{H}$ является подмножеством некоторого множества $\mathcal{H}'$, а функция потерь естественно продолжается до функции, отображающей $\mathcal{H}' \times Z$ в множество вещественных чисел. В таком случае можно разрешить алгоритму возвращать гипотезу $h' \in \mathcal{H}'$ при условии, что выполняется условие $L_{\mathcal{D}}(h') \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$. Если алгоритму разрешено возвращать гипотезу из $\mathcal{H}'$, то мы говорим об обучении, *независимом от представления*, а если он может возвращать только гипотезу из $\mathcal{H}$, то о собственном обучении (proper learning). Обучение, независимое от представления, иногда называют «несобственным» (improper learning).

3.3. Резюме

В этой главе мы определили основную формальную модель обучения, которой будем заниматься, – PAC-обучение. Базовая модель основана на предположении

о реализуемости, а в ее агностическом варианте не налагается никаких ограничений на истинное распределение примеров. Мы также обобщили модель PAC на произвольные функции потерь. Иногда мы будем называть наиболее общую модель просто PAC-обучением, опуская уточнение «агностическое» и предлагая читателю определить функцию потерь из контекста. Желая подчеркнуть, что речь идет именно об исходном определении PAC-обучения, мы будем напоминать, что выполняется предположении о реализуемости. В главе 7 мы обсудим другие подходы к обучаемости.

3.4. Библиографические сведения

Наше самое общее определение агностического PAC-обучения с функциями потерь общего вида заимствовано из работ Владимира Вапника и Алексея Червоненкиса (Vapnik and Chervonenkis, 1971). В частности, мы следуем общей конфигурации обучения, предложенной Вапником (Vapnik, 1982; Vapnik, 1992; Vapnik, 1995; Vapnik, 1998).

Термин «PAC-обучение» был введен Валиантом (Valiant, 1984). Валиант был удостоен премии Тюринга 2010 г. за изобретение модели PAC. В определении Валианта требуется, чтобы выборочная сложность полиномиально зависела от $1/\epsilon$ и $1/\delta$, а также от размера представления гипотез из класса (см. также Kearns and Vazirani, 1994). В главе 6 мы увидим, что если задача вообще допускает PAC-обучение, то выборочная сложность полиномиально зависит от $1/\epsilon$ и $\log(1/\delta)$. В определении Валианта также требуется, чтобы время работы алгоритма обучения полиномиально зависело от тех же величин. Мы же предпочитаем разделять статистический и вычислительный аспекты обучения. О вычислительном аспекте мы подробнее поговорим в главе 8. И, наконец, формализация агностического PAC-обучения изложена по работе Haussler (1992).

3.5. Упражнения

3.1. Монотонность выборочной сложности. Пусть $\mathcal{H}$ – класс гипотез для задачи бинарной классификации. Предположим, что $\mathcal{H}$ является PAC-обучаемым и выборочная сложность описывается функцией $m_{\mathcal{H}}(\cdot, \cdot)$. Покажите, что $m_{\mathcal{H}}$ монотонно возрастает по каждому параметру. Иначе говоря, покажите, что при заданных $\delta \in (0, 1)$ и $0 < \epsilon_1 \leq \epsilon_2 < 1$ справедливо неравенство $m_{\mathcal{H}}(\epsilon_1, \delta) \geq m_{\mathcal{H}}(\epsilon_2, \delta)$. Аналогично покажите, что при заданных $\epsilon \in (0, 1)$ и $0 < \delta_1 \leq \delta_2 < 1$ справедливо неравенство $m_{\mathcal{H}}(\epsilon, \delta_1) \geq m_{\mathcal{H}}(\epsilon, \delta_2)$.

3.2. Пусть $\mathcal{X}$ – дискретная область образцов, а $\mathcal{H}_{\text{Singleton}} = \{h_z : z \in \mathcal{X}\} \cup \{h^-\}$, где для любого $z \in \mathcal{X}$, h_z – функция, определенная следующим образом: $h_z(x) = 1$, если $x = z$ и $h_z(x) = 0$, если $x \neq z$. h^- – это просто отрицательная гипотеза, т. е. $\forall x \in \mathcal{X} h^-(x) = 0$. Здесь из предположения о реализуемости следует, то истинная гипотеза f помечает отрицательной меткой все примеры из области образцов, кроме, быть может, одного.

1. Опишите алгоритм, который реализует правило ERM для обучения $\mathcal{H}_{\text{Singleton}}$ в реализуемой конфигурации.

2. Покажите, что $\mathcal{H}_{\text{Singleton}}$ допускает PAC-обучение. Предложите верхнюю границу выборочной сложности.

3.3. Пусть $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \{0, 1\}$ и пусть $\mathcal{H}$ – класс концентрических окружностей на плоскости, т. е. $\mathcal{H} = \{h_r: r \in \mathbb{R}_+\}$, где $h_r(x) = \mathbb{1}_{[\|x\| \le r]}$. Докажите, что $\mathcal{H}$ допускает PAC-обучение (в предположении о реализуемости) и выборочная сложность ограничена сверху величиной

$$m_{\mathcal{H}}(\epsilon, \delta) \le \left\lceil \frac{\log(1/\delta)}{\epsilon} \right\rceil.$$

3.4. В этом упражнении мы изучаем класс *булевых конъюнкций*, определенный следующим образом. Пространство образцов $\mathcal{X} = \{0, 1\}^d$, а множество меток $\mathcal{Y} = \{0, 1\}$. Литералом от переменных $x_1, \ldots, x_d$ называется простая булева функция вида $f(\mathbf{x}) = x_i$ для некоторого $i \in [d]$ или $f(\mathbf{x}) = 1 - x_i$ для некоторого $i \in [d]$. Мы будем употреблять нотацию $\bar{x}_i$ для обозначения $1 - x_i$. Конъюнкцией называется произведение литералов. В булевой логике произведение обозначается знаком $\wedge$. Например, функция $h(\mathbf{x}) = x_1 \cdot (1 - x_2)$ записывается в виде $x_1 \wedge \bar{x}_2$.

Мы рассматриваем класс гипотез, состоящий из всех конъюнкций литералов от d переменных. Пустая конъюнкция интерпретируется как положительная гипотеза (т. е. функция $h(\mathbf{x})$, возвращающая 1 для всех $\mathbf{x}$). Конъюнкция $x_1 \wedge \bar{x}_1$ (как и любая другая конъюнкция, содержащая одновременно литерал и его отрицание) допустима и интерпретируется как отрицательная гипотеза (т. е. функция $h(\mathbf{x})$, возвращающая 0 для всех $\mathbf{x}$). Мы предполагаем реализуемость, т. е. предполагаем, что существует булева конъюнкция, которая порождает все метки. Таким образом, каждый пример $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ состоит из присваивания значений d булевым переменным $x_1, \ldots, x_d$ и соответственного значения истинности (0 – ложь, 1 – истина).

Например, пусть $d = 3$ и предположим, что истинная конъюнкция имеет вид $x_1 \wedge \bar{x}_2$. Тогда обучающий набор S мог бы содержать такие образцы:

$$((1, 1, 1), 0), ((1, 0, 1), 1), ((0, 1, 0), 0)((1, 0, 0), 1).$$

Докажите, что класс гипотез, содержащий все конъюнкции от d переменных, является PAC-обучаемым, и ограничьте сверху выборочную сложность. Предложите алгоритм, который реализует правило ERM, с временем работы, полиномиально зависящим от $d \cdot m$.

3.5. Пусть $\mathcal{X}$ – область образцов, а $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_m$ – последовательность распределений на $\mathcal{X}$. Пусть $\mathcal{H}$ – конечный класс бинарных классификаторов над $\mathcal{X}$ и $f \in \mathcal{H}$. Предположим, что мы получаем выборку S, содержащую m образцов, которые независимы, но распределены неодинаково: i-й образец выбирается из распределения $\mathcal{D}_i$, а затем y_i присваивается значение $f(x_i)$. Обозначим $\mathcal{D}_m$ среднее, т. е. $\overline{\mathcal{D}}_m = (\mathcal{D}_1 + \cdots + \mathcal{D}_m)/m$.

Зафиксируем параметр верности $\epsilon \in (0,1)$. Покажите, что

$$\mathbb{P}\left[\exists h \in \mathcal{H} \text{ s.t. } L_{(\overline{\mathcal{D}}_m, f)}(h) > \epsilon \text{ и } L_{(S,f)}(h) = 0\right] \le |\mathcal{H}|\, e^{-\epsilon m}.$$

Указание. Воспользуйтесь неравенством между средним арифметическим и средним геометрическим.

3.6. Пусть $\mathcal{H}$ – класс бинарных классификаторов. Покажите, что если $\mathcal{H}$ допускает агностическое PAC-обучение, то $\mathcal{H}$ допускает также PAC-обучение. Более того, если A – успешный алгоритм агностического PAC-обучения для $\mathcal{H}$, то A – также успешный алгоритм PAC-обучения для $\mathcal{H}$.

3.7. (*) **Оптимальный байесовский предиктор.** Покажите, что для любого распределения вероятности $\mathcal{D}$ оптимальный байесовский предиктор действительно является оптимальным в том смысле, что для любого классификатора $g : \mathcal{X} \to \{0, 1\}$ справедливо неравенство $L_{\mathcal{D}}(f_{\mathcal{D}}) \le L_{\mathcal{D}}(g)$.

3.8. (*) Говорят, что алгоритм обучения A лучше B относительно распределения вероятностей $\mathcal{D}$, если

$$L(A(S)) \le L_{\mathcal{D}}(B(S))$$

для всех выборок $S \in (X \times \{0, 1\})^m$. Говорят, что алгоритм обучения A *лучше* B, если A лучше B относительно любого распределения вероятностей $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$.

1. Вероятностным предиктором меток называется функция, которая назначает каждой точке x из области определения значение $h(x) \in [0, 1]$, равное вероятности, с которой будет предсказана метка 1. То есть, имея такую функцию h и входной пример x, мы подбрасываем асимметричную монету с вероятностью выпадения орла $h(x)$ и предсказываем метку 1, если выпадает орел. Формально вероятностный предиктор меток определяется как функция h: $\mathcal{X} \to [0, 1]$. Потеря такой функции h на примере (x, y) по определению равна $|h(x) - y|$, а это в точности вероятность того, что предсказание h не совпадает с y. Отметим, что если функция h детерминированная, т. е. возвращает значения из множества $\{0, 1\}$, то $|h(x) - y| = \mathbb{1}_{[h(x) \neq y]}$. Докажите, что для любого порождающего данные распределения $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$ оптимальный байесовский предиктор дает наименьший риск (относительно функции потерь $\ell(h, (x, y)) = |h(x) - y)|$) среди всех возможных предикторов меток, включая и вероятностные.
2. Пусть $\mathcal{X}$ – область образцов, а $\{0, 1\}$ – множество меток. Докажите, что для любого распределения $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$ существует алгоритм обучения $A_{\mathcal{D}}$, который лучше любого другого алгоритма обучения относительно $\mathcal{D}$.
3. Докажите, что для любого алгоритма обучения A существует распределение вероятностей $\mathcal{D}$ и алгоритм обучения B такие, что A не лучше B относительно $\mathcal{D}$.

3.9. Рассмотрим вариант модели PAC, в котором имеется два оракула примеров: один генерирует положительные примеры, другой – отрицательные, и оба выбирают примеры из истинного распределения $\mathcal{D}$ на $\mathcal{X}$. Формально говоря, задана целевая функция $f : \mathcal{X} \to \{0, 1\}$ и распределение $\mathcal{D}^+$ на $\mathcal{X}^+ = \{x \in \mathcal{X} : f(x) = 1\}$, определенное формулой $\mathcal{D}^+(A) = \mathcal{D}(A) / \mathcal{D}^+(\mathcal{X}^+)$ для любого $A \subset \mathcal{X}^+$. Аналогично $\mathcal{D}^-$ – распределениена $\mathcal{X}^-$, индуцированное $\mathcal{D}$.

Определение PAC-обучаемости в модели с двумя оракулами такое же, как определение стандартной PAC-обучаемости с тем отличием, что здесь обучае-

мый имеет доступ к $m^{+}_{\mathcal{H}}(\epsilon, \delta)$ независимым и одинаково распределенным примерам с распределением $\mathcal{D}^{+}$ и $m^{-}_{\mathcal{H}}(\epsilon, \delta)$ независимым и одинаково распределенным примерам с распределением $\mathcal{D}^{-}$. Цель обучаемого – найти гипотезу h такую, что с вероятностью не менее $1 - \delta$ (для двух случайных обучающих наборов и, возможно, недетерминированных решений, принимаемых алгоритмом обучения) имеет место одновременно $L_{(\mathcal{D}^{+}, f)}(h) \leq \epsilon$ и $L_{(\mathcal{D}^{-}, f)}(h) \leq \epsilon$.

1. (*) Покажите, что если $\mathcal{H}$ допускает PAC-обучение (в стандартной модели с одним оракулом), то $\mathcal{H}$ допускает PAC-обучение и в модели с двумя оракулами).
2. (**) Пусть гипотеза h^{+} всегда возвращает плюс, а гипотеза h^{-} всегда возвращает минус. Предположим, что $h^{+}, h^{-} \in \mathcal{H}$. Покажите, что если $\mathcal{H}$ допускает PAC-обучение в модели с двумя оракулами, то $\mathcal{H}$ допускает PAC-обучениеи в стандартной модели с одним оракулом.

ЛАНЬ®

ОБУЧАЕМОСТЬ И РАВНОМЕРНАЯ СХОДИМОСТЬ

ЛАНЬ®

Первой рассмотренной нами формальной моделью обучения была модель PAC. В главе 2 мы показали, что в предположении о реализуемости любой конечный класс гипотез является PAC-обучаемым. В этой главе мы разработаем общий инструмент, *равномерную сходимость*, и применим его, чтобы показать, что любой конечный класс обучаем в агностической модели PAC с функциями потерь общего вида при условии, что область значений функции потерь ограничена.

4.1. Равномерная сходимость – достаточное условие обучаемости

ЛАНЬ®

Идея, стоящая за условием обучения, обсуждаемым в этой главе, очень проста. Напомним, что при заданном классе гипотез $\mathcal{H}$ парадигма обучения ERM применяется следующим образом: получив обучающую выборку S, обучаемый вычисляет риск (или ошибку) каждой гипотезы h из $\mathcal{H}$ на этой выборке и выводит в качестве результата ту гипотезу, для которой достигается минимум эмпирического риска. Мы надеемся, что h, доставляющая минимум эмпирическому риску на S, будет также давать минимальный (или близкий к минимальному) риск относительно истинного распределения вероятностей данных. Для этого достаточно, чтобы эмпирический риск каждого элемента $\mathcal{H}$ был хорошей аппроксимацией его истинного риска. Иначе говоря, мы хотим, чтобы эмпирический риск был близок к истинному равномерно по всем гипотезам из рассматриваемого класса. Формализуем эту идею в следующем определении.

Определение 4.1 (ϵ-репрезентативная выборка). Обучающий набор S называется ϵ-репрезентативным(относительно множества примеров Z, класса гипотез $\mathcal{H}$, функции потерь ℓ и распределения $\mathcal{D}$), если

$$\forall h \in \mathcal{H}, |L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon.$$

Следующая простая лемма утверждает, что если выборка $(\epsilon/2)$-репрезентативна, то правило обучения ERM гарантированно возвращает хорошую гипотезу.

Лемма 4.2. *Предположим, что обучающий набор S является $(\epsilon/2)$-репрезентативным (относительно множества примеров Z, класса гипотез $\mathcal{H}$, функции потерь ℓ и распределения $\mathcal{D}$). Тогда любой результат применения правила* $\mathrm{ERM}_{\mathcal{H}}(S)$, *т. е. любая гипотеза* $h_S \in \operatorname{argmin}_{h \in \mathcal{H}} L_S(h)$, *удовлетворяет неравенству:*

$$L_{\mathcal{D}}(h_S) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon.$$

Доказательство. Для любой $h \in \mathcal{H}$ имеем

$$L_{\mathcal{D}}(h_S) \leq L_S(h_S) + \frac{\epsilon}{2} \leq L_S(h) + \frac{\epsilon}{2} \leq L_{\mathcal{D}}(h) + \frac{\epsilon}{2} + \frac{\epsilon}{2} = L_{\mathcal{D}}(h) + \epsilon,$$

где первое и третье неравенство следуют из предположения о $(\epsilon/2)$-репрезентативности S (определение 4.1), а второе выполняется, потому что h_S – ERM-предиктор.

Из этой леммы вытекает, что для того, чтобы правило ERM было агностическим PAC-обучаемым, достаточно показать, что с вероятностью не менее $1 - \delta$ случайно выбранный обучающий набор будет ϵ-репрезентативным.

Определение 4.3 (равномерная сходимость). Говорят, что класс гипотез $\mathcal{H}$ обладает *свойством равномерной сходимости* (относительно множества примеров Z и функции потерь ℓ), если существует функция $m^{\mathrm{UC}}_{\mathcal{H}}\colon (0, 1)^2 \to \mathbb{N}$ такая, что для любых $\epsilon, \delta \in (0, 1)$ и для любого распределения вероятностей $\mathcal{D}$ на Z справедливо следующее утверждение: всякая выборка S, содержащая $m \geq m^{\mathrm{UC}}_{\mathcal{H}}(\epsilon, \delta)$ независимых и одинаково распределенных примеров с распределением D, с вероятностью не менее $1 - \delta$ является ϵ-репрезентативной.

Как и в определении выборочной сложности PAC-обучения, функция $m^{\mathrm{UC}}_{\mathcal{H}}$ измеряет (минимальную) выборочную сложность, необходимую для достижения равномерной сходимости, т. е. сколько примеров необходимо, чтобы с вероятностью не менее $1 - \delta$ случайная выборка являлась ϵ-репрезентативной.

Термин *равномерная* в данном случае означает, что имеется фиксированный размер выборки, который относится к любым элементам $\mathcal{H}$ и любым распределениям вероятностей над множеством примеров.

Приведенное ниже следствие непосредственно вытекает из леммы 4.2 и определения равномерной сходимости.

Следствие 4.4. *Если класс $\mathcal{H}$ обладает свойством равномерной сходимости с функцией $m^{UC}_{\mathcal{H}}$, то этот класс допускает агностическое PAC-обучение с выборочной сложностью $m_{\mathcal{H}}(\epsilon, \delta) \leq m^{UC}_{\mathcal{H}}\mathcal{H}(\epsilon/2, \delta)$. Кроме того, в таком случае парадигма* $\mathrm{ERM}_{\mathcal{H}}$ *является успешным агностическим PAC-обучаемым для $\mathcal{H}$.*

ЛАНЬ®

4.2. Конечные классы допускают агностическое PAC-обучение

Ввиду следствия 4.4 утверждение о том, что любой конечный класс гипотез допускает агностическое PAC-обучение, будет справедливо, если мы установим для таких классов факт равномерной сходимости.

Доказательство равномерной сходимости будет состоять из двух шагов, как и доказательство в главе 2. На первом шаге мы применим лемму о границе объединения, а на втором воспользуемся неравенством для концентрации меры. Рассмотрим эти шаги детально.

Зафиксируем некоторые ϵ и δ. Требуется найти размер выборки m, гарантирующий, что для любого распределения $\mathcal{D}$ с вероятностью не менее $1 - \delta$ для случайной выборки $S = (z_1, \ldots, z_m)$ независимых и одинаково распределенных примеров с распределением $\mathcal{D}$ неравенство $|L_S(h) - L_\mathcal{D}(h)| \leq \epsilon$ будет справедливо для любого $h \in \mathcal{H}$. Иными словами,

$$\mathcal{D}^m(\{S : \forall h \in \mathcal{H}, |L_S(h) - L_\mathcal{D}(h)| \leq \epsilon\}) \geq 1 - \delta.$$

Эквивалентно, надо показать, что

$$\mathcal{D}^m(\{S : \exists h \in \mathcal{H}, |L_S(h) - L_\mathcal{D}(h)| > \epsilon\}) < \delta.$$

Записав

$$\{S : \exists h \in \mathcal{H}, |L_S(h) - L_\mathcal{D}(h)| > \epsilon\} = \cup_{h \in \mathcal{H}} \{S : |L_S(h) - L_\mathcal{D}(h)| > \epsilon\}$$

и применив лемму о границе объединения (лемма 2.2), получим

$$\mathcal{D}^m(\{S : \exists h \in \mathcal{H}, |L_S(h) - L_\mathcal{D}(h)| > \epsilon\}) \leq \sum_{h \in \mathcal{H}} \mathcal{D}^m(\{S : |L_S(h) - L_\mathcal{D}(h)| > \epsilon\}). \tag{4.1}$$

На втором шаге мы докажем, что каждое слагаемое в правой части этого неравенства достаточно мало (при достаточно большом m). То есть мы покажем, что для любой фиксированной гипотезы h (которая выбирается еще до выборки обучающего набора) разность между истинным и эмпирическим риском $|L_S(h) - L_\mathcal{D}(h)|$, скорее всего, будет мала.

Напомним, что $L_\mathcal{D}(h) = \mathbb{E}_{z \sim \mathcal{D}}[\ell(h, z)]$ и что $L_S(h) = (1/m)\sum_{i=1}^m \ell(h, z_i)$. Так как все z_i независимы и выбраны из одного и того же распределения $\mathcal{D}$, то математическое ожидание случайной величины $\ell(h, z_i)$ равно $L_\mathcal{D}(h)$. Из свойства линейности математического ожидания следует, что $L_\mathcal{D}(h)$ является также математическим ожиданием $L_S(h)$. Поэтому величина $|L_\mathcal{D}(h) - L_S(h)|$ – отклонение случайной величины $L_S(h)$ от ее математического ожидания. Поэтому остается показать, что мера $L_S(h)$ *сконцентрирована* в окрестности ее математического ожидания.

Одна из основных теорем математической статистики, *закон больших чисел*, утверждает, что когда m стремится к бесконечности, эмпирические средние сходятся к математическому ожиданию. Это верно для величины $L_S(h)$, т. к. она является эмпирическим средним m независимых и одинаково распределенных величин. Но, будучи всего лишь асимптотическим результатом, закон больших чисел ничего не говорит об отклонении эмпирической оценки ошибки от ее истинного значения при любом заданном конечном размере выборки.

Поэтому мы вместо него воспользуемся неравенством Хёфдинга для концентрации вероятностной меры, которое дает оценку отклонения эмпирического среднего от ожидаемого значения.

Лемма 4.5 (неравенство Хёфдинга). *Пусть $\theta_1, \ldots, \theta_m$ – последовательность независимых и одинаково распределенных случайных величин, и предположим, что для любого $i \mathbb{E}[\theta_i] = \mu$ и $\mathrm{P}[a \leq \theta_i \leq b] = 1$. Тогда для любого $\epsilon > 0$*

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m}\theta_i-\mu\right|>\epsilon\right]\le 2\exp(-2m\epsilon^2/(b-a)^2).$$

Доказательство приведено в приложении В.

Вернемся к нашей задаче. Пусть θ_i – случайная величина $\ell(h, z_i)$. Так как h фиксирована и примеры z_1, ..., z_m независимы и одинаково распределены, то θ_1, ..., θ_m – также независимые и одинаково распределенные случайные величины. Кроме того, $L_S(h) = (1/m)\sum_{i=1}^{m}\theta_i$ и $L_{\mathcal{D}}(h) = \mu$. Предположим далее, что область значений ℓ – отрезок [0, 1], так что $\theta_i \in [0, 1]$. Тогда получаем, что

$$\mathcal{D}^m(\{S : |L_S(h) - L_{\mathcal{D}}(h)| > \epsilon\}) = \mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m}\theta_i-\mu\right|>\epsilon\right]\le 2\exp(-2m\epsilon^2). \tag{4.2}$$

Объединяя с неравенством (4.1), получаем

$$\begin{aligned}\mathcal{D}^m(\{S : \exists h\in\mathcal{H}, |L_S(h) - L_{\mathcal{D}}(h)| > \epsilon\}) &\le \sum_{h\in\mathcal{H}} 2\exp(-2m\epsilon^2)\\ &= 2|\mathcal{H}|\exp(-2m\epsilon^2).\end{aligned}$$

Наконец, если взять

$$m\ge\frac{\log(2|\mathcal{H}|/\delta)}{2\epsilon^2},$$

то

$$\mathcal{D}^m(\{S : \exists h\in\mathcal{H}, |L_S(h) - L_{\mathcal{D}}(h)| > \epsilon\}) \le \delta.$$

Следствие 4.6. *Пусть $\mathcal{H}$ – конечный класс гипотез, Z – область примеров и $\ell : \mathcal{H}\times Z \to [0, 1]$ – функция потерь. Тогда $\mathcal{H}$ обладает свойством равномерной сходимости с выборочной сложностью*

$$m_{\mathcal{H}}^{UC}(\epsilon,\delta)\le\left\lceil\frac{\log(2|\mathcal{H}|/\delta)}{2\epsilon^2}\right\rceil.$$

Кроме того, этот класс допускает агностическое PAC-обучение с помощью алгоритма ERM с выборочной сложностью

$$m_{\mathcal{H}}(\epsilon,\delta)\le m_{\mathcal{H}}^{UC}(\epsilon/2,\delta)\le\left\lceil\frac{2\log(2|\mathcal{H}|/\delta)}{\epsilon^2}\right\rceil.$$

Замечание 4.1 (дискретизация). Хотя следствие 4.6 применимо только к конечным классам гипотез, существует простой трюк, позволяющий получить очень хорошую оценку практической выборочной сложности бесконечного класса гипотез. Рассмотрим класс гипотез, параметризованный d параметрами. Например, пусть $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \{\pm 1\}$, а класс гипотез $\mathcal{H}$ – это множество всех функций вида $h_\theta(x) = \text{sign}(x - \theta)$. То есть каждая гипотеза параметризована одним параметром $\theta\in\mathbb{R}$ и выводит 1 для всех образцов, больших θ, и −1 для образцов, меньших θ. Размер этого класса гипотез бесконечен. Но если мы собираемся обучить этот класс на практике, с помощью компьютера, то, наверное, будем представлять ве-

щественные числа в формате с плавающей точкой, скажем, 64 битами. Следовательно, на практике наш класс гипотез параметризован набором скаляров, представимых 64-разрядными числами с плавающей точкой. Таких чисел не более 2^{64}, и, значит, размер класса гипотез не превышает 2^{64d}. Согласно следствию 4.6 получаем, что выборочная сложность таких классов ограничена сверху величиной $(128d + 2\log(2/\delta))/\epsilon^2$. У этой верхней границы есть недостаток – она зависит от конкретного представления вещественных чисел на нашей машине. В главе 6 мы познакомимся со строгим способом анализа выборочной сложности бесконечных классов гипотез. Но, так или иначе, дискретизацию можно использовать для получения грубой оценки выборочной сложности во многих встречающихся на практике случаях.

4.3. Резюме

Если класс гипотез $\mathcal{H}$ обладает свойством равномерной сходимости, то в большинстве случаев эмпирические риски гипотез из $\mathcal{H}$ достаточно точно представляют истинные риски. Равномерной сходимости достаточно для агностической PAC-обучаемости по правилу ERM. Мы показали, что конечные классы гипотез обладают свойством равномерной сходимости и, следовательно, допускают агностическое PAC-обучение.

4.4. Библиографические сведения

Классы функций, для которых выполняется свойство равномерной сходимости, называют также классами Гливенко–Кантелли в честь Валерия Ивановича Гливенко и Франческо Паоло Кантелли, которые доказали первые результаты о равномерной сходимости в 1930-х гг. (см. Dudley, Gine & Zinn, 1991). Связь между равномерной сходимостью и обучаемостью была обстоятельно изучена Вапником, (см. Vapnik, 1992; Vapnik, 1995; Vapnik, 1998). Как мы увидим в главе 6, фундаментальная теорема теории обучения утверждает, что в задачах бинарной классификации равномерная сходимость является не только достаточным, но и необходимым условием обучаемости. Для общих проблем обучения это уже не так (см. Shalev-Shwartz, Shamir, Srebro & Sridharan, 2010).

4.5. Упражнения

4.1. В этом упражнении мы покажем, что выраженное в терминах (ϵ, δ) требование о сходимости ошибок в наших определениях PAC-обучаемости на самом деле очень близко к более простому на вид требованию о средних (или математических ожиданиях). Докажите, что следующие два утверждения эквивалентны (для любого алгоритма обучения A, любого распределения вероятностей $\mathcal{D}$ и любой функций потерь с областью значений [0, 1]):

1. Для любых $\epsilon, \delta > 0$ существует $m(\epsilon, \delta)$ такое, что $\forall m \geq m(\epsilon, \delta)$

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S)) > \epsilon] < \delta.$$

2. $$\lim_{m\to\infty} \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S))] = 0$$

(где $\mathbb{E}_{S\sim\mathcal{D}^m}$ обозначает математическое ожидание по выборкам S размера m).

4.2. Ограниченные функции потерь. В следствии 4.6 мы предполагали, что область значений функции потерь – отрезок [0, 1]. Докажите, что если область значений функции потерь – отрезок $[a, b]$, то выборочная сложность удовлетворяет условиям

$$m_{\mathcal{H}}(\epsilon, \delta) \leq m^{UC}_{\mathcal{H}}(\epsilon/2, \delta) \leq \left\lceil \frac{2\log(2|\mathcal{H}|/\delta)(b-a)^2}{\epsilon^2} \right\rceil.$$

ЛАНЬ®

Глава 5

КОМПРОМИСС МЕЖДУ СМЕЩЕНИЕМ И СЛОЖНОСТЬЮ

В главе 2 мы видели, что при недостаточно аккуратном выборе обучающих данных может иметь место переобучение. Для решения этой проблемы мы ограничили пространство поиска некоторым классом гипотез $\mathcal{H}$. Такой класс можно рассматривать как априорное знание обучаемого о задаче – предположение о том, что в классе $\mathcal{H}$ существует модель задачи с малой ошибкой. Так, в примере о вкусе папайи мы на основе прошлого опыта с другими фруктами можем предположить, что вкус можно предсказать (хотя бы приблизительно) с помощью некоторого прямоугольника на плоскости «цвет–твердость».

Действительно ли априорные знания необходимы для успеха обучения? Быть может, существует некоторый универсальный алгоритм обучения, который не имеет априорных знаний о конкретных задачах и готов принять любой вызов? Давайте уточним постановку. Конкретная задача обучения определяется неизвестным распределением $\mathcal{D}$ на $\mathcal{X} \times \mathcal{Y}$, а цель обучаемого – найти предиктор $h : \mathcal{X} \to \mathcal{Y}$ с достаточно малым риском $L_{\mathcal{D}}(h)$. Таким образом, вопрос заключается в том, существует ли алгоритм обучения A и размер обучающего набора m такие, что для любого распределения $\mathcal{D}$ выполняется следующее утверждение: если A получает m независимых примеров, выбранных из распределения $\mathcal{D}$, то с высокой вероятностью он порождает предиктор h с малым риском.

В первой части этой главы этот вопрос изучается формально. Теорема об отсутствии бесплатных завтраков утверждает, что такого универсального алгоритма обучения не существует. Точнее, утверждается, что в задачах бинарной классификации для любого алгоритма обучения существует распределение, на котором он терпит неудачу. Мы говорим, что обучаемый терпит неудачу, если при получении независимых и одинаково распределенных примеров его выходная гипотеза с высокой вероятностью имеет большой риск, скажем, $\geq 0{,}3$, и в то же время для того же самого распределения существует другой обучаемый, который порождает гипотезу с малым риском. Иными словами, теорема утверждает, что никакой обучаемый не может работать успешно на всех допускающих обучение задачах – для каждого обучаемого найдется задача, на которой он терпит неудачу, тогда как другие обучаемые решают ее правильно.

Поэтому, приступая к решению конкретной проблемы обучения, определяемой некоторым распределением $\mathcal{D}$, мы должны иметь какие-то априорные знания о $\mathcal{D}$. Примером таких априорных знаний может служить знание о том, что $\mathcal{D}$ принадлежит некоторому параметрическому семейству распределений. Мы будем изучать обучение при таких предположениях в главе 24. Другой тип априорного знания о $\mathcal{D}$, с которым мы столкнулись при определении модели PAC-обучения, состоит в том, что в некотором предопределенном классе гипотез $\mathcal{H}$ существует гипотеза h, для которой $L_{\mathcal{D}}(h) = 0$. Более слабое априорное знание о $\mathcal{D}$ заключается в предположении о том, что $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$ мал. В некотором смысле это более слабое предположение о $\mathcal{D}$ – необходимое условие для использования агностической модели PAC-обучения, в которой мы требуем, чтобы риск выходной гипотезы был не намного больше $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$.

Во второй части этой главы мы изучим достоинства и недостатки использования класса гипотез как средства формализовать априорное знание. Мы разложим ошибку алгоритма ERM по классу $\mathcal{H}$ на две составляющих. Первая будет отражать качество априорного знания и измеряться минимальным риском гипотезы из нашего класса, $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$. Эта компонента называется также *ошибкой аппроксимации*, или *смещением* алгоритма в сторону выбора гипотезы из класса $\mathcal{H}$. Вторая компонента – ошибка из-за переобучения, она зависит от размера или сложности класса $\mathcal{H}$ и называется *ошибкой оценивания*. Тем самым возникает задача нахождения компромисса между выбором более сложного класса $\mathcal{H}$ (что уменьшает смещение, но увеличивает риск переобучения) и менее сложного класса $\mathcal{H}$ (что увеличивает смещение, но уменьшает риск переобучения).

5.1. Теорема об отсутствии бесплатных завтраков

В этой части мы докажем, что не существует универсального алгоритма обучения (обучаемого). Для этого мы покажем, что никакой обучаемый не может быть успешным на всех задачах обучения.

Теорема 5.1 (об отсутствии бесплатных завтраков). *Пусть A – алгоритм обучения для задачи бинарной классификации с бинарной функцией потерь в области образцов $\mathcal{X}$. Пусть m – любое число, меньшее $|\mathcal{X}|/2$, которое представляет размер обучающего набора. Тогда существует распределение $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$ такое, что:*

1) *существует функция $f : \mathcal{X} \to \{0, 1\}$, для которой $L_{\mathcal{D}}(f) = 0$;*
2) *с вероятностью не меньше $1/7$ для случайной выборки $S \sim \mathcal{D}^m$ имеем $L_{\mathcal{D}}(A(S)) \geq 1/8$.*

Эта теорема утверждает, что для любого обучаемого существует задача, на которой он терпит неудачу, хотя другой обучаемый может успешно обучиться на ней. Действительно, тривиальным успешным обучаемым в данном случае будет ERM-обучаемый с классом гипотез $\mathcal{H} = \{f\}$ или, более общо, ERM-обучаемый с любым конечным классом гипотез, который содержит f и размер которого $m \geq 8\log(7|\mathcal{H}|/6)$ (см. следствие 2.3).

Доказательство. Пусть C – подмножество $\mathcal{X}$ размера $2m$. Идея доказательства состоит в том, что любой алгоритм обучения, который видит только половину

образцов из C, не имеет никакой информации о том, какими должны быть метки остальных образцов. Следовательно, существует «реальность», т. е. целевая функция f, противоречащая меткам, которые $A(S)$ предсказывает на не виденных им образцах C.

Заметим, что существует $T = 2^{2m}$ возможных функций $C \to \{0, 1\}$. Обозначим эти функции $f_1, \ldots, f_T$. Для каждой такой функции обозначим $\mathcal{D}_i$ распределение на $C \times \{0, 1\}$, определенное следующим образом

$$\mathcal{D}_i(\{(x, y)\}) = \begin{cases} 1/|C|, & \text{если } y = f_i(x) \\ 0, & \text{в противном случае} \end{cases}.$$

Это означает, что вероятность выбрать пару (x, y) равна $1/|C|$, если метка y действительно совпадает со значением f_i, и нулю, если $y \neq f_i(x)$. Очевидно, что $L_{\mathcal{D}_i}(f_i) = 0$.

Мы покажем, что для любого алгоритма A, который принимает обучающий набор, содержащий m примеров из $C \times \{0, 1\}$, и возвращает функцию $A(S) : C \to \{0, 1\}$, справедливо неравенство

$$\max_{i \in [T]} \mathbb{E}_{S \sim \mathcal{D}_i^m} [L_{\mathcal{D}_i}(A(S))] \geq 1/4. \tag{5.1}$$

Очевидно, это означает, что для любого алгоритма A', который принимает обучающий набор содержащий m примеров из $\mathcal{X} \times \{0, 1\}$, существует функция $f : \mathcal{X} \to \{0, 1\}$ и распределение $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$ такие, что $L_{\mathcal{D}}(f) = 0$ и

$$\mathbb{E}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(A'(S))] \geq 1/4. \tag{5.2}$$

Легко проверить, что этого достаточно, чтобы показать, что $\mathrm{P}[L_{\mathcal{D}}(A'(S)) \geq 1/8] \geq 1/7$, а это как раз то, что нам нужно доказать (см. упражнение 5.1).

Теперь займемся доказательством неравенства (5.1). Существует $k = (2m)^m$ последовательностей m примеров из C. Обозначим их $S_1, \ldots, S_k$. Далее, если $S_j = (x_1, \ldots, x_m)$, то обозначим S_j^i последовательность, содержащую примеры из S_j, помеченные функцией f_i, т. е. $S_j^i = ((x_1, f_i(x_1)), \ldots, (x_m, f_i(x_m)))$. Если выборка производится из распределения $\mathcal{D}_i$, то возможные обучающие наборы, которые может получать A, – это $S_1^i, \ldots, S_k^i$, и вероятность выбора каждого из этих наборов одинакова. Поэтому

$$\mathbb{E}_{S \sim \mathcal{D}_i^m} [L_{\mathcal{D}_i}(A(S))] = \frac{1}{k} \sum_{j=1}^{k} L_{\mathcal{D}_i}(A(S_j^i)). \tag{5.3}$$

Поскольку среднее больше минимума и меньше максимума, то получаем:

$$\begin{aligned} \max_{i \in [T]} \frac{1}{k} \sum_{j=1}^{k} L_{\mathcal{D}_i}(A(S_j^i)) &\geq \frac{1}{k} \sum_{j=1}^{k} L_{\mathcal{D}_i}(A(S_j^i)) \\ &= \frac{1}{k} \sum_{j=1}^{k} \frac{1}{T} \sum_{i=1}^{T} L_{\mathcal{D}_i}(A(S_j^i)) \\ &\geq \min_{j \in [k]} \frac{1}{T} \sum_{j=1}^{T} L_{\mathcal{D}_i}(A(S_j^i)). \end{aligned} \tag{5.4}$$

Далее зафиксируем некоторое $j \in [k]$. Обозначим $S_j = (x_1, \ldots, x_m)$, и пусть $v_1, \ldots, v_p$ – примеры из C, отсутствующие в S_j. Очевидно, что $p \geq m$. Следовательно, для любой функции $h : C \to \{0, 1\}$ и любого j имеем:

$$\begin{aligned} L_{\mathcal{D}_j}(h) &= \frac{1}{2m} \sum_{x \in C} \mathbb{1}_{[h(x) \neq f_j(x)]} \\ &\geq \frac{1}{2m} \sum_{r=1}^{p} \mathbb{1}_{[h(v_r) \neq f_j(v_r)]} \\ &\geq \frac{1}{2p} \sum_{r=1}^{p} \mathbb{1}_{[h(v_r) \neq f_j(v_r)]}. \end{aligned} \tag{5.5}$$

Отсюда

$$\begin{aligned} \frac{1}{T} \sum_{i=1}^{T} L_{\mathcal{D}_i}(A(S_j^i)) &\geq \frac{1}{T} \sum_{i=1}^{T} \frac{1}{2p} \sum_{r=1}^{p} \mathbb{1}_{[A(S_j^i)(v_r) \neq f_i(v_r)]} \\ &= \frac{1}{2p} \sum_{r=1}^{p} \frac{1}{T} \sum_{i=1}^{T} \mathbb{1}_{[A(S_j^i)(v_r) \neq f_i(v_r)]} \\ &\geq \frac{1}{2} \cdot \min_{r \in [p]} \frac{1}{T} \sum_{i=1}^{T} \mathbb{1}_{[A(S_j^i)(v_r) \neq f_i(v_r)]}. \end{aligned} \tag{5.6}$$

Теперь зафиксируем некоторое $r \in [p]$. Мы можем разбить все множество функций $f_1, \ldots, f_T$ на $T/2$ непересекающихся пар, где для пары $(f_i, f_{i'})$ справедливо утверждение: для любого $c \in C$, $f_i(c) \neq f_{i'}(c)$ тогда и только тогда, когда $c = v_r$. Поскольку для такой пары мы должны иметь $S_j^i = S_j^{i'}$, то отсюда следует

$$\mathbb{1}_{[A(S_j^i)(v_r) \neq f_i(v_r)]} + \mathbb{1}_{[A(S_j^{i'})(v_r) \neq f_{i'}(v_r)]} = 1,$$

что дает

$$\frac{1}{T} \sum_{i=1}^{T} \mathbb{1}_{[A(S_j^i)(v_r) \neq f_i(v_r)]} = \frac{1}{2}.$$

Объединяя это с формулами (5.6), (5.4) и (5.3), получаем, что выполняется неравенство (5.1). Это завершает наше доказательство. □

5.1.1. Теорема о бесплатных завтраках и априорное знание

Как теорема о бесплатных завтраках соотносится с необходимостью в априорном знании? Рассмотрим ERM-предиктор над классом гипотез $\mathcal{H}$, состоящим из всех функций $f : \mathcal{X} \to \{0, 1\}$. Этот класс представляет отсутствие априорного знания. Любая возможная функция, отображающая область образцов в множество меток, считается хорошим кандидатом. Согласно теореме о бесплатных завтраках, любой алгоритм, который выбирает выходную гипотезу из $\mathcal{H}$, и, в частности, ERM-предиктор на некоторой задаче обучения терпит неудачу. Следовательно, этот класс не допускает PAC-обучения, что формализовано в таком следствии.

Следствие 5.2. *Пусть $\mathcal{X}$ – бесконечное множество и $\mathcal{H}$ – множество всех функций из $\mathcal{X}$ в* $\{0, 1\}$. *Тогда $\mathcal{H}$ не является PAC-обучаемым.*

Доказательство. Предположим противное – что класс $\mathcal{H}$ является обучаемым. Возьмем какие-то $\epsilon < 1/8$ и $\delta < 1/7$. По определению PAC-обучаемости, должен существовать алгоритм обучения A и целое число $m = m(\epsilon, \delta)$ такие, что для любого порождающего данные распределения на $\mathcal{X} \times \{0, 1\}$ справедливо утверждение: если для некоторой функции $f : \mathcal{X} \to \{0, 1\}$ $L_{\mathcal{D}}(f) = 0$, то с вероятностью больше $1 - \delta$ применение алгоритма A к выборкам S размера m, состоящим из независимых примеров с одним и тем же распределением $\mathcal{D}$, дает $L_{\mathcal{D}}(A(S)) \leq \epsilon$. Однако, т. к. $|\mathcal{X}| > 2m$, то по теореме об отсутствии бесплатных завтраков для любого алгоритма обучения (и в т. ч. алгоритма A) существует распределение $\mathcal{D}$ такое, что с вероятностью больше $1/7 > \delta$ риск $L_{\mathcal{D}}(A(S)) > 1/8 > \epsilon$, что противоречит исходному предположению. □

Как можно предотвратить такие неудачи? Уйти от рисков, на которые указывает теорема об отсутствии бесплатных завтраков, можно, воспользовавшись априорным знанием о конкретной задаче обучения, чтобы избежать распределений, которые могут привести к неудаче при обучении для этой задачи. Выразить априорное знание можно, ограничив класс гипотез.

Но как выбрать хороший класс гипотез? С одной стороны, мы хотим верить, что этот класс содержит гипотезу, которая вообще не дает ошибки (в случае PAC-обучения) или, по крайней мере, что наименьшая ошибка, достижимая на гипотезах из этого класса, действительно мала (в случае агностического PAC-обучения). С другой стороны, мы только что видели, что не можем просто взять самый богатый класс, т. е. класс всех функций над заданной областью образцов. Этот компромисс обсуждается в следующем разделе.

5.2. Разложение ошибки

Чтобы ответить на поставленный вопрос, мы разложим ошибку $\mathrm{ERM}_{\mathcal{H}}$-предиктора на две компоненты. Пусть h_S – гипотеза $\mathrm{ERM}_{\mathcal{H}}$. Тогда мы можем написать

$$L_{\mathcal{D}}(h_S) = \epsilon_{\text{app}} + \epsilon_{\text{est}}, \text{ где } \epsilon_{\text{app}} = \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h), \quad \epsilon_{\text{est}} = L_{\mathcal{D}}(h_S) - \epsilon_{\text{app}}. \tag{5.7}$$

- **Ошибка аппроксимации** – минимальный риск, достижимый предиктором из данного класса гипотез. Этот член измеряет риск, возникающий из-за того, что мы ограничились определенным классом, т. е. величину *индуктивного смещения*. Ошибка аппроксимации не зависит от размера выборки и определяется только выбранным классом гипотез. Увеличение класса гипотез может уменьшить ошибку аппроксимации.
  В предположении о реализуемости ошибка аппроксимации равна нулю. Но в агностическом случае она может быть велика[1].

[1] На самом деле она всегда включает ошибку оптимального байесовского предиктора (см. главу 3), минимальную, но неизбежную, в силу возможной недетерминированности мира в этой модели. Иногда в литературе термином «ошибка аппроксимации» называют не $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$, а превышение этой величины над оптимальным байесовским предиктором, т. е. $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) - \epsilon_{\text{Bayes}}$.

- **Ошибка оценивания** – разность между ошибкой аппроксимации и ошибкой ERM-предиктора. Ошибка оценивания возникает, потому что эмпирический риск (т. е. ошибка обучения) – это всего лишь оценка истинного риска, и потому предиктор, минимизирующий эмпирический риск, является всего лишь оценкой предиктора, минимизирующего истинный риск. Качество оценки зависит от размера обучающего набора и размера (сложности) класса гипотез. Как мы показали, для конечного класса гипотез ϵ_{est} увеличивается (логарифмически) с ростом $|\mathcal{H}|$ и уменьшается с ростом m. Можно считать размер класса $\mathcal{H}$ мерой его сложности. В следующих главах мы определим другие меры сложности классов гипотез.

Поскольку наша цель – минимизировать полный риск, мы должны выбрать *компромисс между смещением и сложностью*. С одной стороны, если в качестве $\mathcal{H}$ выбрать очень богатый класс, то ошибка аппроксимации уменьшится, но одновременно может увеличиться ошибка оценивания из-за опасности *переобучения*. С другой стороны, если взять слишком маленький класс $\mathcal{H}$, то ошибка оценивания уменьшится, но может возрасти ошибка аппроксимации, т. е. будет иметь место *недообучение*. Конечно, идеальным $\mathcal{H}$ был бы класс, содержащий только один классификатор – оптимальный байесовский. Но оптимальный байесовский классификатор зависит от истинного распределения $\mathcal{D}$, которое нам неизвестно (а если бы мы его знали, то и обучаться было бы незачем).

Теория обучения исследует, насколько богатым можно сделать класс $\mathcal{H}$, чтобы ошибка оценивания оставалась разумной. Многие эмпирические исследования посвящены проектированию хороших классов гипотез для определенной предметной области. В данном случае «хороший» означает, что ошибка аппроксимации для класса не слишком велика. Идея в том, что, хотя мы не узкие специалисты и не знаем, как сконструировать оптимальный классификатор, все равно мы располагаем некоторым априорным знанием о решаемой задаче, которое позволяет проектировать классы гипотез с не слишком большими ошибками аппроксимации и оценивания. Так, в примере с папайей мы не знаем, как именно по цвету и твердости плода можно предсказать его вкус, но знаем, что прямоугольник в пространстве «цвет–твердость» может оказаться хорошим предиктором.

5.3. Резюме

Теорема о бесплатных завтраках утверждает, что не существует универсального алгоритма обучения. Каждый алгоритм должен быть «заточен» под конкретную задачу, т. е. для успеха он должен использовать априорные знания о задаче. До сих пор мы моделировали априорные знания, требуя, чтобы выходная гипотеза принадлежала некоторому ограниченному классу гипотез. При выборе такого класса мы сталкиваемся с компромиссом между большим, т. е. более сложным классом, для которого, скорее всего, характерна малая ошибка аппроксимации, и более узким классом, гарантирующим, что будет мала ошибка оценивания. В следующей главе мы более подробно изучим поведение ошибки оценивания. А в главе 7 обсудим альтернативные способы выразить априорное знание.

5.4. Библиографические сведения

В работе Wolpert & Macready (1997) доказано несколько теорем об отсутствии бесплатных завтраков для оптимизации, но все они довольно сильно отличаются от теоремы из этой главы. Доказанная нами теорема тесно связана с нижними границами в теории Вапника–Червоненкиса, которую мы будем рассматривать в следующей главе.

5.5. Упражнения

5.1. Докажите, что неравенства (5.2) достаточно, чтобы показать, что $\mathbb{P}[L_{\mathcal{D}}(A(S)) \geq 1/8] \geq 1/7$.

Указание. Пусть θ – случайная величина, принимающая значения из отрезка $[0, 1]$, с математическим ожиданием $\mathbb{E}[\theta] \geq 1/4$. Применяя лемму B.1, покажите, что $\mathbb{P}[\theta \geq 1/8] \geq 1/7$.

5.2. Допустим, вас попросили спроектировать алгоритм обучения, который предсказывает, случится ли у пациента сердечный приступ. Релевантными для алгоритма признаками могут быть артериальное давление (АД), индекс массы тела (ИМТ), возраст (В), уровень физической активности (Ф) и доход (Д).

Вы можете остановиться на одном из двух алгоритмов: выбрать осепараллельный прямоугольник на плоскости АД–ИМТ или выбрать осепараллельный параллелепипед в пятимерном пространстве всех пяти признаков.

1. Объясните плюсы и минусы каждого алгоритма.
2. Объясните, как на ваш выбор может повлиять количество доступных помеченных примеров.

5.3. Докажите, что если $|\mathcal{X}| \geq km$ для некоторого целого положительного $k \geq 2$, то в теореме об отсутствии бесплатных завтраков можно заменить нижнюю границу $1/4$ на $(k-1)/2k = 1/2 - 1/2k$. Точнее, пусть A – алгоритм обучения для задачи бинарной классификации. Пусть m – произвольное число, меньшее $|\mathcal{X}|/k$, представляющее размер обучающего набора. Тогда существует распределение $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$ такое, что:

- существует функция $f : \mathcal{X} \to \{0, 1\}$, для которой $L_{\mathcal{D}}(f) = 0$;
- $\mathbb{E}_{S \sim \mathcal{D}^m}[L_{\mathcal{D}}(A(S))] \geq 1/2 - 1/2k$.

Глава 6

VC-РАЗМЕРНОСТЬ

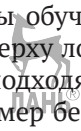
В предыдущей главе мы разложили ошибку правила $\mathrm{ERM}_{\mathcal{H}}$ на ошибку аппроксимации и ошибку оценивания. Ошибка аппроксимации зависит от степени соответствия наших априорных знаний (выраженных в выборе класса гипотез $\mathcal{H}$) неизвестному истинному распределению. А в определении PAC-обучаемости требуется, чтобы ошибка оценивания была равномерно ограничена по всем распределениям.

Наша очередная цель – определить, какие классы $\mathcal{H}$ являются PAC-обучаемыми, и точно охарактеризовать выборочную сложность обучения заданного класса гипотез. До сих пор мы видели, что конечные классы допускают обучение, а класс всех функций (над областью определения бесконечного размера) не допускает. Что делает один класс обучаемым, а другой – нет? Могут ли быть обучаемыми классы бесконечного размера, и если да, то что определяет их выборочную сложность?

В начале этой главы мы покажем, что бесконечные классы таки могут быть обучаемыми, и, следовательно, конечность класса гипотеза не является необходимым условием обучаемости. Затем мы представим удивительно элегантную характеристику семейства обучаемых классов в случае бинарной классификации с бинарной функцией потерь. Эта характеристика была открыта Владимиром Вапником и Алексеем Червоненкисом в 1970 г. и опирается на комбинаторное понятие размерности Вапника–Червоненкиса (VC-размерности). Мы дадим формальное определение VC-размерности, приведем несколько примеров, а затем сформулируем фундаментальную теорему статистической теории обучения, которая объединяет концепции обучаемости, VC-размерности, правила ERM и равномерной сходимости.

6.1. Бесконечные классы могут быть обучаемыми

В главе 4 мы видели, что конечные классы обучаемы и что выборочная сложность такого класса гипотез ограничена сверху логарифмом его размера. Чтобы показать, что размер класса гипотез – неподходящая характеристика выборочной сложности, мы приведем простой пример бесконечного класса гипотез, который тем не менее допускает обучение.



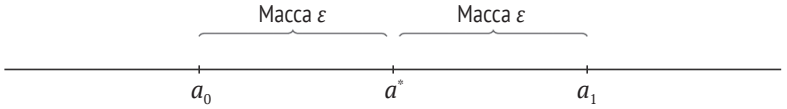
Пример 6.1. Пусть $\mathcal{H}$ – множество ступенчатых функций на вещественной прямой, т. е. $\mathcal{H} = \{h_a : a \in \mathbb{R}\}$, где $h_a : \mathbb{R} \to \{0, 1\}$ определена следующим образом:

$h_a(x) = \mathbb{1}_{[x<a]}$. Напомним, что функция $\mathbb{1}_{[x<a]}$ равна 1, если $x < a$, и 0 в противном случае. Очевидно, что размер $\mathcal{H}$ бесконечен. Тем не менее, следующая лемма показывает, что $\mathcal{H}$ допускает обучение в модели PAC по правилу ERM.

Лемма 6.1. *Пусть $\mathcal{H}$ – определенный выше класс ступенчатых функций. Тогда $\mathcal{H}$ является PAC-обучаемым по правилу ERM с выборочной сложностью $m_{\mathcal{H}}(\epsilon, \delta) \le \lceil \log(2/\delta)/\epsilon \rceil$.*

Доказательство. Пусть a^* – ступенчатая функция такая, что на гипотезе $h^*(x) = \mathbb{1}_{[x<a^*]}$ достигается риск $L_{\mathcal{D}}(h^*) = 0$. Пусть $\mathcal{D}_x$ – маргинальное распределение на области образцов $\mathcal{X}$, и пусть $a_0 < a^* < a_1$ таковы, что

$$\mathbb{P}_{x\sim\mathcal{D}_x}[x \in (a_0, a^*)] = \mathbb{P}_{x\sim\mathcal{D}_x}[x \in (a^*, a_1)] = \epsilon.$$



(Если $\mathcal{D}_x(-\infty, a^*) \le \epsilon$, то полагаем $a_0 = -\infty$ и аналогично для a_1). Для заданного обучающего набора S обозначим $b_0 = \max\{x: (x, 1) \in S\}$ и $b_1 = \min\{x: (x, 0) \in S\}$ (если ни один пример из S не является положительным, то полагаем $b_0 = -\infty$, а если в S нет отрицательных примеров, то полагаем $b_1 = \infty$). Пусть b_S – ступенчатая функция, соответствующая ERM-гипотезе h_S, и, следовательно, $b_S \in (b_0, b_1)$. Поэтому для того, чтобы выполнялось неравенство $L_{\mathcal{D}}(h_S) \le \epsilon$, достаточно, чтобы одновременно имело место $b_0 \ge a_0$ и $b_1 \le a_1$. Иными словами,

$$\mathbb{P}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S) > \epsilon] \le \mathbb{P}_{S\sim\mathcal{D}^m}[b_0 < a_0 \vee b_1 > a_1] = \epsilon,$$

и, применяя лемму о границе объединения, мы можем ограничить правую часть этого неравенства сверху:

$$\mathbb{P}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S) > \epsilon] \le \mathbb{P}_{S\sim\mathcal{D}^m}[b_0 < a_0] + \mathbb{P}_{S\sim\mathcal{D}^m}[b_1 > a_1]. \tag{6.1}$$

Событие $b_0 < a_0$ имеет место тогда и только тогда, когда все примеры из S находятся вне интервала (a_0, a^*), масса вероятности которого по определению равна ϵ, т. е.

$$\mathbb{P}_{S\sim\mathcal{D}^m}[b_0 < a_0] = \mathbb{P}_{S\sim\mathcal{D}^m}[\forall(x, y) \in S, x \notin (a_0, a^*)] = (1 - \epsilon)^m \le e^{-\epsilon m}].$$

Поскольку мы предположили, что $m > \log(2/\delta)/\epsilon$, отсюда следует, что правая часть этого неравенства не больше $\delta/2$. Точно так же легко показать, что $\mathbb{P}_{S\sim\mathcal{D}^m}[b_1 > a_1] \le \delta/2$. Вместе с неравенством (6.1) это завершает доказательство. □

6.2. VC-размерность

Таким образом, мы видим, что конечность $\mathcal{H}$ является достаточным, но не необходимым условием обучаемости. Как мы покажем, правильную характеристику обучаемости дает свойство, называемое VC-размерностью класса гипотез. Чтобы

подойти к определению VC-размерности, вспомним теорему об отсутствии бесплатных завтраков (теорема 5.1) и ее доказательство. Тогда мы показали, что если не вводить ограничений на класс гипотез, то для любого алгоритма обучения противник может построить распределение, на котором алгоритм будет работать плохо, и в то же время существует другой алгоритм обучения, который на этом распределении работает хорошо. Для этого противник взял конечное множество $C \subset \mathcal{X}$ и рассмотрел семейство распределений, сконцентрированных в элементах C. Каждое распределение было произведено от «истинной» целевой функции, отображающей C в {0,1}. Чтобы заставить алгоритм ошибиться, противник воспользовался предоставленной ему возможностью выбирать целевую функцию из множества *всех* возможных функций, отображающих C в {0,1}.

В случае PAC-обучаемости класса гипотез $\mathcal{H}$ противник ограничен построением распределений, для которых на некоторой гипотезе $h \in \mathcal{H}$ достигается нулевой риск. Поскольку мы рассматриваем распределения, сконцентрированные в элементах C, то должны изучить, как $\mathcal{H}$ ведет себя на C. Это приводит к следующему определению.

Определение 6.2 (ограничение $\mathcal{H}$ на C). Пусть $\mathcal{H}$ – класс функций из $\mathcal{X}$ в {0, 1}, и пусть $C = \{c_1, \ldots, c_m\} \subset \mathcal{X}$. Ограничением $\mathcal{H}$ на C называется следующее множество функций из C в {0, 1}:

$$\mathcal{H}_C = \{(h(c_1), \ldots, h(c_m)) : h \in \mathcal{H}\},$$

где каждая функция из C в {0, 1} представляется в виде вектора $\{0, 1\}^{|C|}$.

Если ограничение $\mathcal{H}$ на C – это множество всех функций из C в {0,1}, то мы говорим, что $\mathcal{H}$ разбивает множество C. Дадим формальное определение.

Определение 6.3 (разбиение). Класс гипотез $\mathcal{H}$ разбивает конечное множество $C \subset \mathcal{X}$, если ограничение $\mathcal{H}$ на C является множеством всех функций из C в {0, 1}. Таким образом, $|\mathcal{H}_C| = 2^{|C|}$.

Пример 6.2. Пусть $\mathcal{H}$ – класс ступенчатых функций над $\mathbb{R}$. Возьмем множество $C = \{c_1\}$. Если взять $a = c_1 + 1$, то получим $h_a(c_1) = 1$, а если взять $a = c_1 - 1$, то получим $h_a(c_1) = 0$. Поэтому $\mathcal{H}_C$ – множество всех функций из C в {0, 1}, и $\mathcal{H}$ разбивает C. Теперь возьмем множество $C = \{c_1, c_2\}$, где $c_1 \le c_2$. Никакая гипотеза $h \in \mathcal{H}$ не может объяснить пометку (0, 1), потому что любая ступенчатая функция, назначающая метку 0 примеру c_1, должна назначить метку 0 и примеру c_2. Поэтому не все функции $C \to \{0, 1\}$ входят в $\mathcal{H}_C$, следовательно, C не разбивается классом $\mathcal{H}$.

Возвращаясь к построению противником распределения, как в доказательстве теоремы об отсутствии бесплатных завтраков, мы видим, что если некоторое множество C разбивается классом $\mathcal{H}$, то противник не ограничен гипотезами из класса $\mathcal{H}$, поскольку может построить распределение на C, основываясь на *любой* целевой функции из C в {0, 1}, и при этом не нарушить предположение о реализуемости. Отсюда немедленно вытекает

Следствие 6.4. *Пусть $\mathcal{H}$ – класс гипотез, состоящий из функций, отображающих $\mathcal{X}$ в {0, 1}. Обозначим m размер обучающего набора. Предположим, что существует*

множество $C \subset \mathcal{X}$ размера $2m$, которое разбивается $\mathcal{H}$. Тогда для любого алгоритма обучения A существует распределение $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$ и предиктор $h \in \mathcal{H}$ такие, что $L_{\mathcal{D}}(h) = 0$, но с вероятностью не менее $1/7$ для случайной выборки $S \sim \mathcal{D}^m$ имеем $L_{\mathcal{D}}(A(S)) \geq 1/8$.

Следствие 6.4 говорит, что если $\mathcal{H}$ разбивает некоторое множество C размера $2m$, то мы не можем обучить $\mathcal{H}$ на m примерах. Интуитивно понятно, что если множество C разбивается классом $\mathcal{H}$ и мы получаем выборку, содержащую половину образцов из C, то метки этих образцов не дают нам никакой информации о метках образцов из второй половины – любую возможную пометку оставшихся образцов можно объяснить некоторой гипотезой из $\mathcal{H}$. Можно высказать такую философскую мысль:

Если кто-то может объяснить любое явление, то все его объяснения бесполезны.

Это прямо подводит нас к определению VC-размерности.

Определение 6.5 (VC-размерность). VC-размерностью класса гипотез $\mathcal{H}$, обозначаемой $\text{VCdim}(\mathcal{H})$, называется максимальный размер множества $C \subset \mathcal{X}$, которое может быть разбито классом $\mathcal{H}$. Если $\mathcal{H}$ может разбивать множества произвольно большого размера, то говорят, что VC-размерность $\mathcal{H}$ бесконечна.

Таким образом, из следствия 6.4 непосредственно вытекает

Теорема 6.6. *Пусть $\mathcal{H}$ – класс, имеющий бесконечную VC-размерность. Тогда $\mathcal{H}$ не является PAC-обучаемым.*

Доказательство. Поскольку VC-размерность $\mathcal{H}$ бесконечна, то для любого обучающего набора размера m существует разбитый набор размера $2m$, и утверждение вытекает из следствия 6.4. □

Ниже в этой главе мы увидим, что верно и обратное: конечная VC-размерность гарантирует обучаемость. Поэтому VC-размерность характеризует PAC-обучаемость. Но, прежде чем дальше углубляться в теорию, приведем несколько примеров.

6.3. Примеры

В этом разделе мы вычислим VC-размерность нескольких классов гипотез. Чтобы установить, что $\text{VCdim}(\mathcal{H}) = d$, мы должны показать, что:

1) существует множество C размера d, разбитое классом $\mathcal{H}$;
2) никакое множество C размера $d + 1$ не может быть разбито классом $\mathcal{H}$.

6.3.1. Ступенчатые функции

Пусть $\mathcal{H}$ – класс ступенчатых функций над $\mathbb{R}$. Напомним, что в примере 6.2 мы показали, что для произвольного множества $C = \{c_1\}$ $\mathcal{H}$ разбивает C, поэтому $\text{VCdim}(\mathcal{H}) \geq 1$. Мы также показали, что для произвольного множества $C = \{c_1, c_2\}$, где $c_1 \leq c_2$, $\mathcal{H}$ не разбивает C. Поэтому мы можем заключить, что $\text{VCdim}(\mathcal{H}) = 1$.

6.3.2. Интервалы

Пусть $\mathcal{H}$ – класс интервалов над $\mathbb{R}$, т. е. $\mathcal{H} = \{h_{a,b} : a, b \in \mathrm{R}, a < b\}$, где $h_{a,b} : \mathbb{R} \to \{0, 1\}$ – функция такая, что $h_{a,b}(x) = \mathbb{1}_{[x \in (a,b)]}$. Возьмем множество $C = \{1, 2\}$. Тогда $\mathcal{H}$ разбивает C (убедитесь, что понимаете, почему), и, следовательно, $\mathrm{VCdim}(\mathcal{H}) \geq 2$. Теперь возьмем произвольное множество $C = \{c_1, c_2, c_3\}$ и предположим без ограничения общности, что $c_1 \leq c_2 \leq c_3$. Тогда никакой интервал не может получить пометку (1, 0, 1), поэтому $\mathcal{H}$ не разбивает C. Таким образом, мы делаем вывод, что $\mathrm{VCdim}(\mathcal{H}) = 2$.

ЛАНЬ®

6.3.3. Осепараллельные прямоугольники

Пусть $\mathcal{H}$ – класс осепараллельных прямоугольников:

$$\mathcal{H} = \{h_{(a_1,a_2,b_1,b_2)}: a_1 \leq a_2 \text{ и } b_1 \leq b_2\},$$

где

$$h_{(a_1, a_2, b_1, b_2)}(x_1, x_2) = \begin{cases} 1, & \text{если } a_1 \leq x_1 \leq a_2 \text{ и } b_1 \leq x_2 \leq b_2 \\ 0, & \text{в противном случае} \end{cases}. \tag{6.2}$$

Мы покажем, что $\mathrm{VCdim}(H) = 4$. Чтобы доказать это, необходимо найти множество из 4 точек, которое разбивается классом $\mathcal{H}$, и показать, что никакое множество из 5 точек не может быть разбито $\mathcal{H}$. Найти множество из 4 разделяемых точек просто (см. рис. 6.1). Теперь рассмотрим произвольное множество $C \subset \mathbb{R}^2$ из 5 точек. Возьмем в C самую левую точку (с наименьшей первой координатой), самую правую точку (с наибольшей первой координатой), самую нижнюю точку (с наименьшей второй координатой) и самую верхнюю точку (с наибольшей второй координатой). Без ограничения общности обозначим $C = \{c_1, \ldots, c_5\}$, считая, что c_5 не совпадает ни с одной из четырех вышеупомянутых точек. Теперь определим пометку (1, 1, 1, 1, 0). Такую пометку невозможно получить с помощью осепараллельного прямоугольника. Действительно, этот прямоугольник должен содержать точки $c_1, \ldots, c_4$, но в таком случае он должен содержать и c_5, потому что ее координаты находятся внутри интервалов, определенных выбранными точками. Следовательно, C не разбивается классом $\mathcal{H}$ и потому $\mathrm{VCdim}(\mathcal{H}) = 4$.

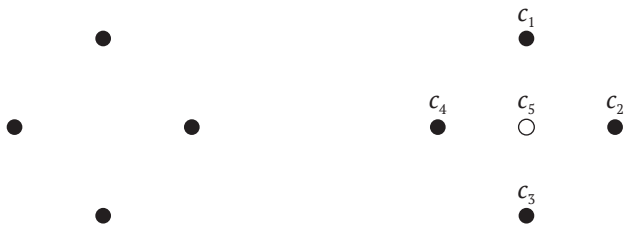


Рис. 6.1. Слева: 4 точки разделяются осепараллельными прямоугольниками. Справа: никакой осепараллельный прямоугольник не может пометить точку 5 нулем, а остальные точки – единицей

6.3.4. Конечные классы

Пусть $\mathcal{H}$ конечный класс. Очевидно, что для любого множества C имеем $|\mathcal{H}_C| \leq |\mathcal{H}|$, поэтому C нельзя разбить, если $|\mathcal{H}| < 2^{|C|}$. Отсюда следует, что $\text{VCdim}(\mathcal{H}) \leq \log_2(|\mathcal{H}|)$. Тем самым показано, что PAC-обучаемость конечных классов вытекает из более общего утверждения о PAC-обучаемости классов с конечной VC-размерностью, каковой факт мы установим в следующем разделе. Отметим, однако, что VC-размерность конечного класса $\mathcal{H}$ не может быть значительно меньше $\log_2(|\mathcal{H}|)$. Например, пусть $\mathcal{X} = \{1, \ldots, k\}$ для некоторого k. Рассмотрим класс ступенчатых функций (определенный в примере 6.2). Тогда $|\mathcal{H}| = k$, но $\text{VCdim}(\mathcal{H}) = 1$. Поскольку k может быть произвольно велико, разрыв между $\log_2(|\mathcal{H}|)$ и $\text{VCdim}(\mathcal{H})$ может быть сколь угодно большим.

6.3.5. VC-размерность и количество параметров

В приведенных выше примерах VC-размерность совпадала с количеством параметров, определяющих класс гипотез. Так бывает часто, но не всегда. Рассмотрим, к примеру, область образцов $\mathcal{X} = \mathbb{R}$ и класс гипотез $\mathcal{H} = \{h_\theta : \theta \in \mathbb{R}\}$, где $h_\theta : \mathcal{X} \to \{0, 1\}$ определено формулой $h_\theta(x) = \lfloor 0{,}5 \sin(\theta x) \rfloor$. Можно доказать, что $\text{VCdim}(\mathcal{H}) = \infty$, т. е. *для любого* d можно найти d точек, которые разделяются $\mathcal{H}$ (см. упражнение 6.8).

6.4. Фундаментальная теорема PAC-обучения

Мы уже показали, что класс, имеющий бесконечную VC-размерность, не является обучаемым. Обратное утверждение также верно, что приводит к фундаментальной теореме статистической теории обучения.

Теорема 6.7 (фундаментальная теорема статистического обучения). *Пусть $\mathcal{H}$ – класс гипотез, состоящий из функций, отображающих область образцов $\mathcal{X}$ в* $\{0, 1\}$, *и пусть в качестве функции потерь используется бинарная функция. Тогда следующие утверждения эквивалентны:*

1) *$\mathcal{H}$ обладает свойством равномерной сходимости;*
2) *любое правило ERM является успешным алгоритмом агностического PAC-обучения для $\mathcal{H}$;*
3) *$\mathcal{H}$ допускает агностическое PAC-обучение;*
4) *$\mathcal{H}$ допускает PAC-обучение;*
5) *любое правило ERM является успешным алгоритмом PAC-обучения для $\mathcal{H}$;*
6) *$\mathcal{H}$ имеет конечную VC-размерность.*

Доказательство теоремы приведено в следующем разделе.

VC-размерность не только характеризует PAC-обучаемость, но и определяет выборочную сложность.

Теорема 6.8 (фундаментальная теорема статистического обучения – количественный вариант). *Пусть $\mathcal{H}$ – класс гипотез, состоящий из функций, отобража-*

ющих область образцов $\mathcal{X}$ в $\{0, 1\}$, *и пусть в качестве функции потерь используется бинарная функция. Предположим, что* $\text{VCdim}(\mathcal{H}) = d < \infty$. *Тогда существуют константы* C_1 *и* C_2 *такие, что:*

1) $\mathcal{H}$ *обладает свойством равномерной сходимости с выборочной сложностью*

$$C_1 \frac{d + \log(1/\delta)}{\epsilon^2} \leq m^{UC}_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d + \log(1/\delta)}{\epsilon^2};$$

2) $\mathcal{H}$ *допускает агностическое PAC-обучение с выборочной сложностью*

$$C_1 \frac{d + \log(1/\delta)}{\epsilon^2} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d + \log(1/\delta)}{\epsilon^2};$$

3) $\mathcal{H}$ *допускает PAC-обучение с выборочной сложностью*

$$C_1 \frac{d + \log(1/\delta)}{\epsilon} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d + \log(1/\epsilon) + \log(1/\delta)}{\epsilon}.$$

Доказательство будет приведено в главе 28.

Замечание 6.3. Мы сформулировали фундаментальную теорему для задач бинарной классификации. Аналогичный результат имеет место для других проблем обучения, например, регрессии с абсолютной или квадратичной потерей. Однако эта теорема неверна для всех задач обучения. В частности, обучаемость иногда возможна, даже если свойство равномерной сходимости не выполняется (пример мы увидим в упражнениях к главе 13). Более того, бывают ситуации, когда правило ERM не действует, но добиться обучаемости все-таки можно с помощью других правил обучения.

6.5. Доказательство теоремы 6.7

Импликацию 1 → 2 мы уже доказали в главе 4. Импликации 2 → 3 и 3 → 4 тривиальны, как и импликация 2 → 5. Импликации 4 → 6 и 5 → 6 следуют из теоремы об отсутствии бесплатных завтраков. Самая трудная часть – доказательство импликации 6 → 1. Оно основано на двух утверждениях:

- если $\text{VCdim}(\mathcal{H}) = d$, то, даже если класс $\mathcal{H}$ бесконечен, при ограничении его на конечное множество $C \subset \mathcal{X}$ «эффективный» размер $|\mathcal{H}_C|$ составляет только $O(|C|^d)$. Иначе говоря, размер $\mathcal{H}_C$ растет полиномиально, а не экспоненциально с ростом $|C|$. Это утверждение часто называют *леммой Зауэра*, хотя оно было независимо сформулировано и доказано также Шелахом и Перлесом. Его точная формулировка будет приведена в разделе 6.5.1 позже;
- в разделе 4 мы показали, что конечные классы гипотез обладают свойством равномерной сходимости. Ниже, в разделе 6.5.2 мы обобщим этот результат и покажем, что равномерная сходимость имеет место, когда класс гипотез имеет «малый эффективный размер». Под этим понимаются классы, для которых $|\mathcal{H}_C|$ растет полиномиально вместе с ростом $|C|$.

6.5.1. Лемма Зауэра и функция роста

Мы определили понятие *разбиения*, рассматривая ограничение $\mathcal{H}$ на конечное множество примеров. Функция роста измеряет максимальный «эффективный» размер $\mathcal{H}$ на множестве m примеров. Дадим формальное определение.

Определение 6.9 (функция роста). Пусть $\mathcal{H}$ – класс гипотез. *Функция роста* $\mathcal{H}$, обозначаемая $\tau_{\mathcal{H}} : \mathbb{N} \to \mathbb{N}$, определяется следующим образом

$$\tau_{\mathcal{H}}(m) = \max_{C \subset \mathcal{X}, |C|=m} |\mathcal{H}_C|.$$

Словами: $\tau_{\mathcal{H}}(m)$ – это число различных функций, отображающих множество C размера m в {0, 1}, которые можно получить путем ограничения $\mathcal{H}$ на C.

Очевидно, что если $\text{VCdim}(\mathcal{H}) = d$, то для любого $m \le d$ имеет место $\tau_{\mathcal{H}}(m) = 2^m$. В таких случаях $\mathcal{H}$ индуцирует все возможные функции из C в {0, 1}. Следующая красивая лемма, независимо доказанная Зауэром, Шелахом и Перлесом, показывает, что, когда m становится больше VC-размерности, функция роста растет полиномиально, а не экспоненциально с ростом m.

Лемма 6.10 (Зауэра–Шелаха–Перлеса). *Пусть $\mathcal{H}$ – класс гипотез, имеющий размерность* $\text{VCdim}(\mathcal{H}) \le d < \infty$. *Тогда для всех m* $\tau_{\mathcal{H}}(m) \le \sum_{i=0}^{d} \binom{m}{i}$. *В частности, если* $m > d + 1$, *то* $\tau_{\mathcal{H}}(m) \le (em/d)^d$.

Доказательство леммы Зауэра*

Для доказательства леммы достаточно доказать следующее более сильное утверждение: для любого множества $C = \{c_1, \ldots, c_m\}$

$$\forall \mathcal{H}, |\mathcal{H}_C| \le |\{B \subseteq C : \mathcal{H} \text{ разбивает } B\}|. \tag{6.3}$$

Этого действительно достаточно, потому что если $\text{VCdim}(\mathcal{H}) \le d$, то никакое множество, размер которого больше d, не разбивается классом $\mathcal{H}$, и, следовательно,

$$|\{B \subseteq C : \mathcal{H} \text{ разбивает } B\}| \le \sum_{i=0}^{d} \binom{m}{i}.$$

Если $m > d + 1$, то правая часть этого неравенства не превышает $(em/d)^d$ (см. лемму А.5 в приложении А).

Осталось доказать неравенство (6.3), и мы сделаем это по индукции. Для $m = 1$ вне зависимости от $\mathcal{H}$ обе части неравенства либо равны 1, либо 2 (считается, что пустое множество разбивается классом $\mathcal{H}$). Предположим, что неравенство (6.3) выполняется для множеств размера $k < m$, и докажем его для множеств размера m. Зафиксируем $\mathcal{H}$ и $C = \{c_1, \ldots, c_m\}$. Обозначим $C' = \{c_2, \ldots, c_m\}$ и определим еще два множества:

$$Y_0 = \{(y_2, \ldots, y_m) : (0, y_2, \ldots, y_m) \in \mathcal{H}_C \vee (1, y_2, \ldots, y_m) \in \mathcal{H}_C\},$$

и

$$Y_1 = \{(y_2, \ldots, y_m) : (0, y_2, \ldots, y_m) \in \mathcal{H}_C \wedge (1, y_2, \ldots, y_m) \in \mathcal{H}_C\}.$$

Легко проверить, что $|\mathcal{H}_C| = |Y_0| + |Y_1|$. Кроме того, $Y_0 = \mathcal{H}_{C'}$, поэтому по предположению индукции (примененному к $\mathcal{H}$ и C') получаем

$$|Y_0| = |\mathcal{H}_{C'}| \le |\{B \subseteq C' : \mathcal{H} \text{ разбивает } B\}| = |\{B \subseteq C : c_1 \notin B \wedge \mathcal{H} \text{ разбивает } B\}|.$$

Далее определим $\mathcal{H}' \subseteq \mathcal{H}$:

$$\mathcal{H}' = \{h \in \mathcal{H}: \exists h' \in \mathcal{H} \text{ такое, что } (1 - h'(c_1), h'(c_2), \ldots, h'(c_m)) = (h(c_1), h(c_2), \ldots, h(c_m)\},$$

т. е. $\mathcal{H}'$ содержит пары гипотез, которые согласуются на C' и различаются на c_1. Из этого определения ясно, что если $\mathcal{H}'$ разбивает множество $B \subseteq C'$, то он разбивает и множество $B \cup \{c_1\}$ и наоборот. В сочетании с тем фактом, что $Y_1 = \mathcal{H}'_{C'}$, и используя предположение индукции (теперь примененное к $\mathcal{H}'$ и C'), получаем, что

$$\begin{aligned} |Y_1| = |\mathcal{H}'_{C'}| &\le |\{B \subseteq C' : \mathcal{H}' \text{ разбивает } B\}| = |\{B \subseteq C' : \mathcal{H}' \text{ разбивает } B \cup \{c_1\}\}| \\ &= |\{B \subseteq C : c_1 \in B \wedge \mathcal{H}' \text{ разбивает } B\}| \le |\{B \subseteq C: c_1 \in B \wedge \mathcal{H} \text{ разбивает } B\}|. \end{aligned}$$

Таким образом, мы показали, что

$$\begin{aligned} |\mathcal{H}_C| &= |Y_0| + |Y_1| \\ &\le |\{B \subseteq C: c_1 \notin B \wedge \mathcal{H} \text{ разбивает } B\}| + |\{B \subseteq C: c_1 \in B \wedge \mathcal{H} \text{ разбивает } B\}| \\ &= |\{B \subseteq C: \mathcal{H} \text{ разбивает } B\}|, \end{aligned}$$

что и завершает доказательство.

6.5.2. Равномерная сходимость для классов небольшого эффективного размера

В этом разделе мы докажем, что если $\mathcal{H}$ имеет небольшой эффективный размер, то он обладает свойством равномерной сходимости.

Теорема 6.11. *Пусть $\mathcal{H}$ – некоторый класс, а $\tau_{\mathcal{H}}$ – его функция роста. Тогда для любого распределения $\mathcal{D}$ и для любого $\delta \in (0, 1)$ с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ имеем*

$$|L_{\mathcal{D}}(h) - L_S(h)| \le \frac{4 + \sqrt{\log(\tau_{\mathcal{H}}(2m))}}{\delta\sqrt{2m}}.$$

Прежде чем доказывать эту теорему, завершим доказательство теоремы 6.7.

Доказательство теоремы 6.7. Достаточно доказать, что если VC-размерность конечна, то имеет место свойство равномерной сходимости. Мы докажем, что

$$m_{\mathcal{H}}^{UC}(\epsilon, \delta) \le 4\frac{16d}{(\delta\epsilon)^2}\log\left(\frac{16d}{(\delta\epsilon)^2}\right) + \frac{16d\log(2e/d)}{(\delta\epsilon)^2}.$$

Из леммы Зауэра мы знаем, что для $m > d$ выполняется неравенство $\tau_{\mathcal{H}}(2m) \le (2em/d)^d$. В сочетании с теоремой 6.11 получаем, что с вероятностью не менее $1 - \delta$

$$|L_S(h) - L_{\mathcal{D}}(h)| \leq \frac{4 + \sqrt{d \log(2em/d)}}{\delta\sqrt{2m}}.$$

Для простоты предположим, что $\sqrt{d \log(2em/d)}$; тогда

$$|L_S(h) - L_{\mathcal{D}}(h)| \leq \frac{1}{\delta}\sqrt{\frac{2d \log(2em/d)}{m}}.$$

Чтобы правая часть этого неравенства была не меньше ϵ, необходимо, чтобы

$$m \geq \frac{2d \log(m)}{(\delta\epsilon)^2} + \frac{2d \log(2e/d)}{(\delta\epsilon)^2}.$$

С помощью стандартных алгебраических преобразований (см. лемму A.2 в приложении A) легко показать, что достаточным условием для этого является выполнение неравенства

$$m \geq 4\frac{2d}{(\delta\epsilon)^2} \log\left(\frac{2d}{(\delta\epsilon)^2}\right) + \frac{4d \log(2e/d)}{(\delta\epsilon)^2}. \qquad \square$$

Замечание 6.4. Верхняя граница $m_{\mathcal{H}}^{UC}$, которую мы вывели в доказательстве теоремы 6.7, не является максимально точной. В главе 28 приведен более детальный анализ, который дает границы, указанные в теореме 6.8.

*Доказательство теоремы 6.11**

Для начала покажем, что

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} |L_{\mathcal{D}}(h) - L_S(h)|\right] \leq \frac{4 + \sqrt{\log(\tau_{\mathcal{H}}(2m))}}{\sqrt{2m}}. \tag{6.4}$$

Поскольку случайная величина $\sup_{h\in\mathcal{H}}|L_{\mathcal{D}}(h) - L_S(h)|$ неотрицательна, теорема 6.11 непосредственно вытекает отсюда, если воспользоваться неравенством Маркова (см. раздел B.1).

Чтобы ограничить левую часть неравенства 6.4, сначала заметим, что для любого $h \in \mathcal{H}$ можно записать $L_{\mathcal{D}}(h) = \mathbb{E}_{S'\sim\mathcal{D}^m}[L_{S'}(h)]$, где $S' = z'_1, \ldots, z'_m$ – дополнительная независимая и одинаково распределенная выборка. Поэтому

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} |L_{\mathcal{D}}(h) - L_S(h)|\right] = \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} \left|\mathop{\mathbb{E}}_{S'\sim\mathcal{D}^m} L_{S'}(h) - L_S(h)\right|\right].$$

Согласно обобщенному неравенству треугольника

$$\left|\mathop{\mathbb{E}}_{S'\sim\mathcal{D}^m}[L_{S'}(h) - L_S(h)]\right| \leq \mathop{\mathbb{E}}_{S'\sim\mathcal{D}^m}|L_{S'}(h) - L_S(h)|,$$

а из того, что верхняя грань математического ожидания не превышает математического ожидания верхних граней, следует, что

$$\sup_{h\in\mathcal{H}} \mathbb{E}_{S'\sim\mathcal{D}^m} |L_{S'}(h) - L_S(h)| \le \mathbb{E}_{S'\sim\mathcal{D}^m} \sup_{h\in\mathcal{H}} |L_{S'}(h) - L_S(h)|.$$

Формально оба предыдущих неравенства следуют из неравенства Йенсена. Объединяя все вместе, получаем

$$\mathbb{E}_{S'\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} |L_{\mathcal{D}}(h) - L_S(h)|\right] \le \mathbb{E}_{S,S'\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} |L_{S'}(h) - L_S(h)|\right]$$
$$= \mathbb{E}_{S,S'\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} \frac{1}{m}\left|\sum_{i=1}^{m} (\ell(h, z_i') - \ell(h, z_i)\right|\right]. \tag{6.5}$$

Математическое ожидание в правой части берется по двум независимым и одинаково распределенным выборкам $S = z_1, \ldots, z_m$ и $S' = z_1', \ldots, z_m'$. Поскольку все эти $2m$ векторов независимы и одинаково распределены, ничего не изменится, если заменить случайный вектор z_i случайным вектором z_i'. Если мы так поступим, то вместо члена $(\ell(h, z_i') - \ell(h, z_i))$ в формуле (6.5) появится член $-(\ell(h, z_i') - \ell(h, z_i))$. Отсюда следует, что для любого $\sigma \in \{\pm 1\}^m$ правая часть (6.5) равна

$$\mathbb{E}_{S,S'\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} \frac{1}{m}\left|\sum_{i=1}^{m} s_i(\ell(h, z_i') - \ell(h, z_i))\right|\right].$$

Поскольку это справедливо для любого $\sigma \in \{\pm 1\}^m$, то останется справедливым и тогда, когда мы случайно выбираем каждую компоненту σ из равномерного распределения на $\{\pm 1\}$, обозначаемого $U_{\pm}$. Поэтому правую часть (6.5) можно записать также в виде

$$\mathbb{E}_{\sigma\sim U_{\pm}^m} \mathbb{E}_{S,S'\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}} \frac{1}{m}\left|\sum_{i=1}^{m} s_i(\ell(h, z_i') - \ell(h, z_i))\right|\right],$$

и в силу линейности математического ожидания это равно

$$\mathbb{E}_{S,S'\sim\mathcal{D}^m} \mathbb{E}_{\sigma\sim U_{\pm}^m}\left[\sup_{h\in\mathcal{H}} \frac{1}{m}\left|\sum_{i=1}^{m} s_i(\ell(h, z_i') - \ell(h, z_i))\right|\right].$$

Далее зафиксируем S и S', и пусть C – образцы, встречающиеся в S и S'. Тогда можно взять верхнюю грань только по $h \in \mathcal{H}_C$. Поэтому

$$\mathbb{E}_{\sigma\sim U_{\pm}^m}\left[\sup_{h\in\mathcal{H}} \frac{1}{m}\left|\sum_{i=1}^{m} s_i(\ell(h, z_i') - \ell(h, z_i))\right|\right] = \mathbb{E}_{\sigma\sim U_{\pm}^m}\left[\max_{h\in\mathcal{H}_C} \frac{1}{m}\left|\sum_{i=1}^{m} s_i(\ell(h, z_i') - \ell(h, z_i))\right|\right].$$

Зафиксируем некоторое $h \in \mathcal{H}_C$ и обозначим $\theta_h = (1/m)\sum_{i=1}^m \sigma_i(\ell(h, z_i') - \ell(h, z_i))$. Так как $\mathbb{E}[\theta_h] = 0$ и θ_h – среднее независимых величин, каждая из которых принимает значения из отрезка $[-1, 1]$, то в силу неравенства Хёфдинга для любого $\rho > 0$

$$\mathbb{P}[|\theta_h| > \rho] \le 2\exp(-2m\rho^2).$$

Применив лемму о границе объединения к $h \in \mathcal{H}_C$, получим, что для любого $\rho > 0$

$$\mathbb{P}\left[\max_{h \in \mathcal{H}_C} |\theta_h| > \rho\right] \leq 2|\mathcal{H}_C| \exp(-2m\rho^2).$$

Наконец, по лемме A.4 из приложения A из предыдущего неравенства следует, что

$$\mathbb{E}\left[\max_{h \in \mathcal{H}_C} |\theta_h|\right] \leq \frac{4 + \sqrt{\log(|\mathcal{H}_C|)}}{\sqrt{2m}}.$$

Если вспомнить определение $\tau_{\mathcal{H}}$, то доказанный результат можно записать в виде:

$$\mathop{\mathrm{E}}_{S \sim \mathcal{D}^m}\left[\sup_{h \in \mathcal{H}} |L_{\mathcal{D}}(h) - L_S(h)|\right] \leq \frac{4 + \sqrt{\log(\tau_{\mathcal{H}}(2m))}}{\sqrt{2m}}.$$

6.6. Резюме

Фундаментальная теорема теории обучения характеризует PAC-обучаемость классов бинарных классификаторов в терминах VC-размерности. VC-размерность класса – это комбинаторное свойство, равное максимальному размеру выборки, допускающей разбиение этим классом. Фундаментальная теорема утверждает, что класс является PAC-обучаемым тогда и только тогда, когда его VC-размерность конечна, и описывает выборочную сложность PAC-обучения. Теорема также говорит, что если проблема вообще допускает обучение, то выполняется свойство равномерной сходимости, и потому проблема обучаема с помощью ERM-правила.

6.7. Библиографические сведения

Определение VC-размерности и ее связь с обучаемостью и равномерной сходимостью впервые изучены в основополагающей работе Vapnik and Chervonenkis (1971). Связь с определением PAC-обучаемости исследована в работе Blumer, Ehrenfeucht, Haussler, and Warmuth (1989).

Было предложено несколько обобщений VC-размерности. Например, fat-размерность (fat-shattering dimension) характеризует обучаемость некоторых проблем регрессии (Kearns, Schapire & Sellie, 1994; Alon, Ben-David, Cesa-Bianchi & Haussler, 1997; Bartlett, Long & Williamson, 1994; Anthony & Bartlet, 1999), а размерность Натараджана – обучаемость некоторых проблем многоклассового обучения (Natarajan, 1989). Однако в общем случае обучаемость и равномерная сходимость не эквивалентны (см. (Shalev-Shwartz, Shamir, Srebro & Sridharan, 2010; Daniely, Sabato, Ben-David & Shalev-Shwartz, 2011).

Лемма Зауэра была доказана Зауэром для решения проблемы Эрдёша (Sauer, 1972). Шелах (и Перлес) доказали ее в качестве полезной леммы в теории

устойчивых моделей Шелаха (Shelah, 1972). По словам Гиля Калаи[1], позже Бенджи Вайсс задал Перлесу вопрос о подобном результате в контексте эргодической теории, и Перлес, забыв о том, что когда-то уже доказывал это, доказал его снова. Вапник и Червоненкис доказали эту лемму в контексте статистической теории обучения.

6.8. Упражнения

6.1. Докажите следующее свойство монотонности VC-размерности: для любых двух классов гипотез из $\mathcal{H}' \subseteq \mathcal{H}$ следует, что $\text{VCdim}(\mathcal{H}') \leq \text{VCdim}(\mathcal{H})$.

6.2. Пусть дана конечная область определения $\mathcal{X}$ и число $k \leq |\mathcal{X}|$. Вычислите VC-размерность следующих классов (и докажите свои результаты):

1. $\mathcal{H}^{\mathcal{X}}_{=k} = \{h \in \{0, 1\}^{\mathcal{X}} : |\{x : h(x) = 1\}| = k\}$, т. е. множество всех функций, которые сопоставляют значение 1 ровно k элементам $\mathcal{X}$.
2. $\mathcal{H}_{at\text{-}most\text{-}k} = \{h \in \{0, 1\}^{\mathcal{X}} : |\{x : h(x) = 1\}| \leq k \text{ или } |\{x : h(x) = 0\}| \leq k\}$.

6.3. Пусть $\mathcal{X}$ – булев гиперкуб $\{0, 1\}^n$. Для множества $I \subseteq \{1, 2, \ldots, n\}$ определим *функцию четности* h_I следующим образом. Для двоичного вектора $\mathbf{x} = (x_1, x_2, \ldots, xn) \in \{0, 1\}^n$

$$h_I(\mathbf{x}) = \left(\sum_{i \in I} x_i\right) \bmod 2.$$

(То есть h_I вычисляет четность суммы битов, номера которых принадлежат I.) Чему равна VC-размерность класса всех таких функций четности, $\mathcal{H}_{n\text{-parity}} = \{h_I : I \subseteq \{1, 2, \ldots, n\}\}$?

6.4. Мы доказали лемму Зауэра, показав, что для любого класса $\mathcal{H}$ конечной VC-размерности d и любого подмножества A области определения

$$|\mathcal{H}_A| \leq |\{B \subseteq A : \mathcal{H} \text{ разбивает } B\}| \leq \sum_{i=0}^{d} \binom{|A|}{i}.$$

Покажите, что существуют случаи, когда оба эти неравенства строгие (т. е. $\leq$ можно заменить на $<$), а также случаи, когда они обращаются в равенства. Продемонстрируйте все четыре комбинации $=$ и $<$.

6.5. VC-размерность осепараллельных прямоугольников в $\mathbb{R}^d$: пусть $\mathcal{H}^d_{\text{rec}}$ – класс осепараллельных прямоугольников в $\mathbb{R}^d$. Мы уже видели, что $\text{VCdim}(\mathcal{H}^2_{\text{rec}}) = 4$. Докажите, что в общем случае $\text{VCdim}(\mathcal{H}^d_{\text{rec}}) = 2d$.

6.6. VC-размерность класса булевых конъюнкций: пусть $\mathcal{H}^d_{\text{con}}$ – класс булевых конъюнкций от переменных $x_1, \ldots, x_d$ ($d \geq 2$). Мы уже знаем, что этот класс конечный и потому (агностически) PAC-обучаемый. В этом упражнении мы вычислим $\text{VCdim}(\mathcal{H}^d_{\text{con}})$.

[1] http://gilkalai.wordpress.com/2008/09/28/extremal-combinatorics-iii-some-basic-theorems.

1. Покажите, что $|\mathcal{H}^d_{\text{con}}| \le 3^d + 1$.
2. Выведите отсюда, что $\text{VCdim}(\mathcal{H}) \le d\log 3$.
3. Покажите, что $\mathcal{H}^d_{\text{con}}$ разбивает множество единичных векторов $\{\mathbf{e}_i : i \le d\}$.
4. (**) Покажите, что $\text{VCdim}(\mathcal{H}^d_{\text{con}}) \le d$.

Указание. Предположим противное – что существует множество $C = \{c_1, \ldots, c_{d+1}\}$, которое разбивается классом $\mathcal{H}^d_{\text{con}}$. Пусть $h_1, \ldots, h_{d+1}$ – гипотезы из $\mathcal{H}^d_{\text{con}}$, удовлетворяющие условию

$$\forall i, j \in [d+1], h_i(c_j) = \begin{cases} 0, & i = j \\ 1, & \text{в противном случае} \end{cases}.$$

Для любого $i \in [d + 1]$ гипотеза h_i (точнее, соответствующая h_i конъюнкция) содержит некоторый литерал ℓ_i, принимающий значение false на c_i и true на всех c_j, для которых $j \ne i$. Применив принцип Дирихле, покажите, что должна существовать пара $i < j \le d + 1$ такая, что ℓ_i и ℓ_j используют одно и то же x_k, и, воспользовавшись этим фактом, придите к противоречию с определением конъюнкций h_i и h_j.

5. Рассмотрим класс $\mathcal{H}^d_{\text{mcon}}$ монотонных булевых конъюнкций над $\{0, 1\}^d$. Здесь монотонность означает, что конъюнкция не содержит отрицаний. Как и в случае $\mathcal{H}^d_{\text{mcon}}$, пустая конъюнкция интерпретируется как всюду положительная гипотеза. Мы дополним класс $\mathcal{H}^d_{\text{mcon}}$ всюду отрицательной гипотезой h^-. Покажите, что $\text{VCdim}(\mathcal{H}^d_{\text{mcon}}) = d$.

6.7. Мы показали, что для конечного класса гипотез $\mathcal{H}\text{VCdim}(\mathcal{H}) \le \lfloor\log(|\mathcal{H}|)\rfloor$. Но это всего лишь верхняя граница. VC-размерность класса может быть гораздо меньше.

1. Приведите пример бесконечного класса $\mathcal{H}$ функций над отрезком вещественной прямой $\mathcal{X} = [0, 1]$ такого, что $\text{VCdim}(\mathcal{H}) = 1$.
2. Приведите пример конечного класса гипотез $\mathcal{H}$ над отрезком $\mathcal{X} = [0, 1]$ такого, что $\text{VCdim}(\mathcal{H}) = \lfloor\log_2(|\mathcal{H}|)\rfloor$.

6.8. (*) Часто бывает, что VC-размерность класса гипотез равна (или может быть ограничена сверху) количеству параметров, которые необходимо задать, чтобы определить принадлежащую классу гипотезу. Например, если $\mathcal{H}$ – класс осепараллельных прямоугольников в $\mathbb{R}^d$, то $\text{VCdim}(\mathcal{H}) = 2d$, т. е. равна количеству параметров, определяющих прямоугольник в $\mathbb{R}^d$. А в этом упражнении мы приведем пример, показывающий, что не всегда это так. Мы увидим, что класс гипотез может быть очень сложным и даже необучаемым, хотя параметров в нем совсем немного.

Рассмотрим область образцов $\mathcal{X} = \mathbb{R}$ и класс гипотез

$$\mathcal{H} = \{x \mapsto \lceil\sin(\theta x)\rceil : \theta \in \mathbb{R}\}$$

(мы полагаем, что $\lceil -1 \rceil = 0$). Докажите, что $\text{VCdim}(\mathcal{H}) = \infty$.

Указание. Этот результат можно доказать несколькими способами. Один из вариантов – применить следующую лемму: если $0.x_1x_2x_3\ldots$ – двоичное представление числа $x \in (0, 1)$, то для любого натурального числа $m \lceil\sin(2^m \pi x)\rceil = (1 - x_m)$ при условии, что $\exists k \ge m$ такое, что $x_k = 1$.

6.9. Пусть $\mathcal{H}$ – класс интервалов со знаком, т. е. $\mathcal{H} = \{h_{a,b,s} : a \le b,\ s \in \{-1, 1\}\}$, где

$$h_{a,b,s}(x) = \begin{cases} s, & \text{если } x \in [a,b] \\ -s, & \text{если } x \notin [a,b] \end{cases}.$$

Вычислите VCdim($\mathcal{H}$).

6.10. Пусть $\mathcal{H}$ – класс функций, отображающих $\mathcal{X}$ в {0, 1}.

1. Докажите, что если VCdim($\mathcal{H}$) ≥ d для какого-нибудь d, то для некоторого распределения вероятностей $\mathcal{D}$ на $\mathcal{X} \times \{0, 1\}$ и для любого размера выборки m

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S))] \ge \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + \frac{d-m}{2d}.$$

Указание. Воспользуйтесь упражнением 5.3 в главе 5.

2. Докажите, что для любого PAC-обучаемого класса $\mathcal{H}$ VCdim(H) < ∞. (Заметьте, что это импликация 3 → 6 в теореме 6.7.)

6.11. VC-размерность объединения. Пусть $\mathcal{H}_1, \ldots, \mathcal{H}_r$ – классы гипотез над некоторой фиксированной областью определения $\mathcal{X}$. Обозначим $d = \max_i \text{VCdim}(\mathcal{H}_i)$ и предположим для простоты, что $d \ge 3$.

1. Докажите, что

$$\text{VCdim}(\cup_{i=1}^{r} \mathcal{H}_i) \le 4d\log(2d) + 2\log(r).$$

Указание. Возьмем множество k примеров и предположим, что оно разбивается классом объединения. Следовательно, класс объединения может породить все 2^k возможных вариантов пометки этих примеров. С помощью леммы Зауэра докажите, что класс объединения не может порождать более r^{kd} вариантов пометки. Поэтому $2^k \le r^{kd}$. Теперь воспользуйтесь леммой A.2.

2. (*) Докажите, что для $r = 2$ имеет место неравенство

$$\text{VCdim}(\mathcal{H}_1 \cup \mathcal{H}_2) \le 2d + 1.$$

6.12. Классы Дадли. В этом упражнении мы рассмотрим алгебраическую структуру для определения классов концептов над $\mathbb{R}^n$ и продемонстрируем связь между VC-размерностью таких классов и их алгебраическими свойствами. Для заданной функции $f : \mathbb{R}^n \to \mathbb{R}$ определим соответствующую функцию $POS(f)(x) = \mathbb{1}_{[f(x)>0]}$. Для класса $\mathcal{F}$ вещественных функций определим соответствующий класс $POS(\mathcal{F}) = \{POS(f) : f \in \mathcal{F}\}$. Говорят, что семейство $\mathcal{F}$ вещественных функций *линейно замкнуто*, если для любых $f, g \in \mathcal{F}$ и $r \in \mathbb{R}$ $(f + rg) \in \mathcal{F}$ (сложение и умножение функций на скаляр определены поточечно, т. е. для любого $x \in \mathbb{R}^n$ $(f + rg)(x) = f(x) + rg(x)$). Отметим, что если семейство функций линейно замкнуто, то его можно рассматривать как векторное пространство над полем вещественных чисел. Для функции $g : \mathbb{R}^n \to \mathbb{R}$ и семейства функций $\mathcal{F}$ определим $\mathcal{F} + g = \{f + g : f \in \mathcal{F}\}$. Классы гипотез, имеющие представление вида $POS(\mathcal{F} + g)$ при некотором векторном пространстве функций $\mathcal{F}$ и некоторой функции g, называются *классами Дадли*.

1. Покажите, что для любой функции $g : \mathbb{R}^n \to \mathbb{R}$ и любого векторного пространства функций $\mathcal{F}$, определенного выше, VCdim($POS(\mathcal{F} + g)$) = VCdim($POS(\mathcal{F})$).
2. (**) Для любого линейно замкнутого семейства вещественных функций $\mathcal{F}$ VC-размерность соответствующего класса $POS(\mathcal{F})$ равна линейной размерности $\mathcal{F}$ (как векторного пространства). *Указание.* Пусть $f_1, \ldots, f_d$ – базис векторного пространства $\mathcal{F}$. Рассмотрим отображение $x \mapsto (f_1(x), \ldots, f_d(x))$ (из $\mathbb{R}^n$ в $\mathbb{R}^d$). Это отображение индуцирует соответствие между функциями над $\mathbb{R}^n$ вида $POS(f)$ и однородными линейными полупространствами в $\mathbb{R}^d$ (VC-размерность класса однородных линейных полупространств анализируется в главе 9).
3. Покажите, что каждый из следующих классов можно представить в виде класса Дадли:
   1) класс HS_n полупространств в $\mathbb{R}^n$ (см. главу 9);
   2) класс HHS_n всех однородных полупространств в $\mathbb{R}^n$ (см. главу 9);
   3) класс B_d всех функций, определенных (открытыми) шарами в $\mathbb{R}^d$. Воспользуйтесь представлением Дадли для вычисления VC-размерности этого класса;
   4) обозначим P_n^d класс функций, определенных полиномиальными неравенствами степени $\leq d$, а именно:

$$P_n^d = \{h_p : p - \text{полином степени} \leq d \text{ от переменных } x_1, \ldots, x_n\},$$

   где для $\mathbf{x} = (x_1, \ldots, x_n)$ $h_p(\mathbf{x}) = \mathbb{1}_{[p(\mathbf{x}) \geq 0]}$ (степень полинома от нескольких переменных определяется как максимальная сумма показателей степени в каждом члене. Например, степень $p(\mathbf{x}) = 3x_1^3x_2^2 + 4x_3x_7^2$ равна 5).

1. Воспользуйтесь представлением Дадли для вычисления VC-размерности класса P_1^d – класса всех полиномов степени d над $\mathbb{R}$.
2. Докажите, что класс всех полиномиальных классификаторов над $\mathbb{R}$ имеет бесконечную VC-размерность.
3. Воспользуйтесь представлением Дадли для вычисления VC-размерности класса P_n^d (в виде функции от d и n).

Глава 7

НЕРАВНОМЕРНАЯ ОБУЧАЕМОСТЬ

В обсуждавшемся до сих пор понятии PAC-обучаемости размер выборки может зависеть от параметров верности и уверенности, но он является равномерным относительно правила обучения и истинного распределения данных. Следовательно, классы, допускающие обучение в этом смысле, ограничены (они должны иметь конечную VC-размерность, как утверждает теорема 6.7). В этой главе мы рассмотрим более слабые понятия обучаемости. Мы поговорим об их полезности и приведем характеристику классов концептов, допускающих обучение в смысле этих определений.

Начнем с определения понятия «неравномерной обучаемости», при котором размер выборки может зависеть от гипотезы, выдвинутой обучаемым. Затем мы приведем характеристику неравномерной обучаемости и покажем, что она действительно слабее агностической PAC-обучаемости. Мы также докажем, что достаточное условие неравномерной обучаемости – класс $\mathcal{H}$ является счетным объединением классов гипотез, каждый из которых обладает свойством равномерной сходимости. Эти результаты будут доказаны в разделе 7.2, где мы введем в рассмотрение новую парадигму обучения – структурную минимизацию риска (Structural Risk Minimization – SRM). В разделе 7.3 мы опишем парадигму SRM для счетных классов гипотез, что подведет нас к парадигме минимальной длины описания (Minimum Description Length – MDL). Парадигма MDL дает формальное обоснование философскому принципу бритвы Оккама. Затем в разделе 7.4 мы познакомимся с *согласованностью* – еще более слабым понятием обучаемости. Наконец, мы обсудим важность и полезность различных понятий обучаемости.

7.1. Неравномерная обучаемость

«Неравномерная обучаемость» допускает зависимость размера выборки от конкурирующих гипотез. Говорят, что гипотеза h является (ϵ, δ)-конкурирующей с гипотезой h', если с вероятностью больше $1 - \delta$ имеет место неравенство

$$L(h) \leq L_{\mathcal{D}}(h') + \epsilon.$$

В случае PAC-обучаемости понятие «конкуренции» не очень полезно, поскольку мы ищем гипотезу с абсолютно низким риском (если выполняется предположение о реализуемости) или с риском, низким по сравнению минимально до-

стижимым на гипотезах из нашего класса (в случае агностического обучения). Поэтому размер выборки зависит только от параметров верности и уверенности. Но в случае неравномерной обучаемости мы разрешаем, чтобы размер выборки имел вид $m_{\mathcal{H}}(\epsilon, \delta, h)$, т. е. он может также зависеть от гипотезы h, с которой мы конкурируем. Дадим формальное определение.

Определение 7.1. Класс гипотез $\mathcal{H}$ называется *неравномерно обучаемым*, если существует алгоритм обучения A и функция $m_{\mathcal{H}}^{\text{NUL}} : (0, 1)^2 \times \mathcal{H} \to \mathbb{N}$ такая, что для любых $\epsilon, \delta \in (0, 1)$ и любой $h \in \mathcal{H}$ справедливо утверждение: если $m \geq m_{\mathcal{H}}^{\text{NUL}}(\epsilon, \delta, h)$, то для любого распределения $\mathcal{D}$ с вероятностью не ниже $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$

$$L(A(S)) \leq L_{\mathcal{D}}(h) + \epsilon.$$

Сейчас полезно вспомнить определение агностической PAC-обучаемости (определение 3.3):

Класс гипотез $\mathcal{H}$ называется агностически PAC-обучаемым, если существует алгоритм обучения и функция $m_{\mathcal{H}}: (0, 1)^2 \to \mathbb{N}$ такие, что для любых $\epsilon, \delta \in (0, 1)$ и для любого распределения $\mathcal{D}$ справедливо утверждение: если $m \geq m_{\mathcal{H}}(\epsilon, \delta)$, то с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$

$$L_{\mathcal{D}}(A(S)) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon.$$

Отметим, что отсюда следует, что для любой гипотезы $h \in \mathcal{H}$

$$L(A(S)) \leq L_{\mathcal{D}}(h) + \epsilon.$$

В обоих случаях мы требуем, чтобы выходная гипотеза была (ϵ, δ)-конкурирующей с любой другой гипотезой в классе. Но разница между ними в том, может ли размер выборки m зависеть от гипотезы h, с которой сравнивается ошибка $A(S)$. Заметим, что неравномерная обучаемость – это более слабая форма агностической PAC-обучаемости. То есть, если класс допускает агностическое PAC-обучение, то он допускает и неравномерное обучение.

ЛАНЬ®

7.1.1. Характеристика неравномерной обучаемости

Наша цель – охарактеризовать неравномерную обучаемость. В предыдущей главе мы нашли элегантную характеристику PAC-обучаемых классов, показав, что класс бинарных классификаторов допускает агностическое PAC-обучение тогда и только тогда, когда его VC-размерность конечна. А следующая теорема характеризует неравномерно обучаемые классы для задачи бинарной классификации.

Теорема 7.2. *Класс гипотез $\mathcal{H}$ бинарных классификаторов допускает неравномерное обучение тогда и только тогда, когда он является счетным объединением агностически PAC-обучаемых классов гипотез.*

Доказательство теоремы 7.2 опирается на следующий результат, представляющий самостоятельный интерес.

Теорема 7.3. *Пусть $\mathcal{H}$ – класс гипотез, который можно представить в виде счетного объединения классов гипотез $\mathcal{H} = \bigcup_{n\in\mathbb{N}}\mathcal{H}_n$, каждый из которых обладает свойством равномерной сходимости. Тогда $\mathcal{H}$ допускает неравномерное обучение.*

Напомним, что в главе 4 мы показали, что равномерная сходимость – достаточное условие PAC-обучаемости. Теорема 7.3 обобщает этот результат на неравномерную обучаемость. Доказательство мы приведем в следующем разделе, когда введем новую парадигму обучения. А сейчас вернемся к доказательству теоремы 7.2.

Доказательство теоремы 7.2. Сначала предположим, что $\mathcal{H} = \bigcup_{n\in\mathbb{N}}\mathcal{H}_n$, где каждый класс $\mathcal{H}_n$ допускает агностическое PAC-обучение. Из фундаментальной теоремы статистического обучения следует, что каждый $\mathcal{H}_n$ обладает свойством равномерной сходимости. Поэтому из теоремы 7.3 получаем, что $\mathcal{H}$ является неравномерно обучаемым.

Обратно, предположим, что $\mathcal{H}$ допускает неравномерное обучение с помощью некоторого алгоритма A. Для любого $n\in\mathbb{N}$ обозначим $\mathcal{H}_n = \{h\in\mathcal{H} : m^{\mathrm{NUL}}_{\mathcal{H}}(1/8, 1/7, h) \le n\}$. Очевидно, что $\mathcal{H}_n = \bigcup_{n\in\mathbb{N}}\mathcal{H}_n$. Кроме того, согласно определению $m^{\mathrm{NUL}}_{\mathcal{H}}$, для любого распределения $\mathcal{D}$, которое удовлетворяет предположению о реализуемости относительно $\mathcal{H}_n$, с вероятностью не ниже 6/7 для случайной выборки $S \sim \mathcal{D}^n$ имеем $L_{\mathcal{D}}(A(S)) \le 1/8$. Согласно фундаментальной теореме статистического обучения, отсюда следует, что VC-размерность $\mathcal{H}_n$ должна быть конечной, поэтому $\mathcal{H}_n$ является PAC-обучаемым. □

Следующий пример показывает, что неравномерная обучаемость действительно является строгим ослаблением агностической PAC-обучаемости, т. е. существуют классы гипотез, которые являются неравномерно обучаемыми, но не являются PAC-обучаемыми

Пример 7.1. Рассмотрим проблему бинарной классификации, в которой область образцов $\mathcal{X} = \mathbb{R}$. Для любого $n\in\mathbb{N}$ обозначим $\mathcal{H}_n$ класс полиномиальных классификаторов степени n, т. е. $\mathcal{H}_n$ – множество всех классификаторов вида $h(x) = \mathrm{sign}(p(x))$, где $p : \mathbb{R} \to \mathbb{R}$ – полином степени n. Пусть $\mathcal{H} = \bigcup_{n\in\mathbb{N}}\mathcal{H}_n$. Тогда $\mathcal{H}$ – класс всех полиномиальных классификаторов над $\mathbb{R}$. Легко проверить, что $\mathrm{VCdim}(\mathcal{H}) = \infty$, тогда как $\mathrm{VCdim}(\mathcal{H}_n) = n + 1$ (см. упражнение 7.12). Следовательно, $\mathcal{H}$ не является PAC-обучаемым, но, по теореме 7.3, он является неравномерно обучаемым.

7.2. Структурная минимизация риска

До сих пор мы представляли априорные знания, задавая класс гипотез $\mathcal{H}$, который, как мы полагаем, включает хороший предиктор для решаемой задачи обучения. Но есть и другой способ выразить априорные знания – задать предпочтения гипотез внутри $\mathcal{H}$. В парадигме структурной минимизации риска (SRM) мы сначала предполагаем, что $\mathcal{H}$ можно записать в виде $\mathcal{H} = \bigcup_{n\in\mathbb{N}}\mathcal{H}_n$, а затем задаем весовую функцию $w : \mathbb{N} \to [0, 1]$, которая назначает вес каждому классу гипотез $\mathcal{H}_n$, так что больший вес отражает большую предпочтительность класса. В этом разделе мы обсудим, как осуществить обучение с таким априорным знанием.

А в следующем опишем две важных схемы выбора весов, в т. ч. минимальную длину описания.

Итак, пусть класс гипотез $\mathcal{H}$ можно записать в виде $\mathcal{H} = \bigcup_{n\in\mathbb{N}}\mathcal{H}_n$. Например, $\mathcal{H}$ может быть классом всех полиномиальных классификаторов, а $\mathcal{H}_n$ – классом полиномиальных классификаторов степени n (см. пример 7.1). Предположим, что для любого n класс $\mathcal{H}_n$ обладает свойством равномерной сходимости (см. определение 4.3 в главе 4) с функцией выборочной сложности $m^{\mathrm{UC}}_{\mathcal{H}_n}(\epsilon, \delta)$. Определим также функцию $\epsilon_n : \mathbb{N}\times(0, 1) \to (0, 1)$ следующим образом:

$$\epsilon_n(m, \delta) = \min\{\epsilon\in(0, 1) : m^{\mathrm{UC}}_{\mathcal{H}_n}(\epsilon, \delta) \le m\}. \tag{7.1}$$

Словами это можно выразить так: мы имеем фиксированный размер выборки m и хотим найти наименьшую нижнюю границу разности между эмпирическим и истинным риском, достижимую при использовании выборок, содержащих m примеров.

Из определений равномерной сходимости и ϵ_n следует, что для любых m и δ с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ справедливо утверждение

$$\forall h\in\mathcal{H}_n|L_{\mathcal{D}}(h) - L_S(h)| \le \epsilon_n(m, \delta). \tag{7.2}$$

Обозначим $w : \mathbb{N} \to [0, 1]$ функцию такую, что $\sum_{n=1}^{\infty} w(n) \le 1$. Будем называть w *весовой функцией* над классами гипотез $\mathcal{H}_1, \mathcal{H}_2, \ldots$. Такая весовая функция может отражать важность, приписываемую алгоритмом обучения различным классам гипотез, или некоторую меру сложности классов гипотез. Если $\mathcal{H}$ – конечное объединение N классов гипотез, то можно просто приписать всем классам один и тот же вес $1/N$. Это будет означать, что априори мы не отдаем предпочтения никакому классу гипотез. Разумеется, если есть основания полагать (априорное знание), что некоторый класс содержит правильную целевую функцию с большей вероятностью, чем другие, то ему следует приписать больший вес. Если $\mathcal{H}$ – бесконечное (счетное) объединение классов гипотез, то равновесная схема уже невозможна, но есть много других схем. Например, можно выбрать $w(n) = 6/(\pi^2 n^2)$ или $w(n) = 2^{-n}$. Ниже в этой главе мы предложим еще один удобный способ определения весовых функций с помощью языков описания.

Правило SRM основано на «минимизации границы». Это означает, что цель парадигмы – найти гипотезу, которая минимизирует некоторую верхнюю границу истинного риска. Граница, которую стремится минимизировать правило SRM, описывается следующей теоремой.

Теорема 7.4. *Пусть $w : \mathbb{N} \to [0, 1]$ – такая функция, что $\sum_{n=1}^{\infty} w(n) \le 1$. Пусть $\mathcal{H}$ – класс гипотез, который можно записать в виде $\mathcal{H} = \bigcup_{n\in\mathbb{N}}\mathcal{H}_n$, где каждый класс $\mathcal{H}_n$ обладает свойством равномерной сходимости с функцией выборочной сложности $m^{\mathrm{UC}}_{\mathcal{H}_n}$. Пусть функция ϵ_n определена, как в (7.1). Тогда для любого $\delta\in(0, 1)$ и распределения $\mathcal{D}$ с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ имеет место следующее неравенство (одновременно) для всех $n\in\mathbb{N}$ и $h\in\mathcal{H}_n$:*

$$|L_{\mathcal{D}}(h) - L_S(h)| \le \epsilon_n(m, w(n)\cdot\delta).$$

Поэтому для любого $\delta\in(0, 1)$ и распределения $\mathcal{D}$ с вероятностью не менее $1 - \delta$ справедливо утверждение:

$$\forall h \in \mathcal{H}_n,\ L_{\mathcal{D}}(h) \le L_S(h) + \min_{n:h\in\mathcal{H}_n} \epsilon_n(m, w(n)\cdot\delta). \tag{7.3}$$

Доказательство. Для любого n определим $\delta_n = w(n)\delta$. Из предположения о том, что для всех n имеет место равномерная сходимость со скоростью, описанной неравенством (7.2), следует, что если заранее зафиксировать n, то с вероятностью не менее $1 - \delta_n$ для случайной выборки $S \sim \mathcal{D}^m$ справедливо утверждение:

$$\forall h \in \mathcal{H}_n,\ |L_{\mathcal{D}}(h) - L_S(h)| \le \epsilon_n(m, \delta_n).$$

Применяя лемму о границе объединения при $n = 1, 2, \ldots$, получаем, что с вероятностью не менее $1 - \sum_n \delta_n = 1 - \delta\sum_n w(n) \ge 1 - \delta$ предыдущее утверждение имеет место для всех n, что и завершает доказательство. □

Обозначим

$$n(h) = \min\{n : h \in \mathcal{H}_n\}. \tag{7.4}$$

Тогда из (7.3) следует, что

$$L_{\mathcal{D}}(h) \le L_S(h) + \epsilon_{n(h)}(m, w(n(h))\cdot\delta).$$

Парадигма SRM подразумевает поиск такой h, которая минимизирует эту границу. Это формализовано в следующем псевдокоде:

Структурная минимизация риска (SRM)

априорное знание:
$\mathcal{H} = \bigcup_n \mathcal{H}_n$, где $\mathcal{H}_n$ обладает свойством равномерной сходимости с $m^{\mathrm{UC}}_{\mathcal{H}_n}$
$w : \mathbb{N} \to [0, 1]$, где $\sum_n w(n) \le 1$
определение: ϵ_n, как в выражении (7.1); $n(h)$, как в выражении (7.4)
вход: обучающий набор $S \sim \mathcal{D}^m$, уверенность δ
выход: $h \in \operatorname{argmin}_{h\in\mathcal{H}}[Ls(h) + \epsilon_{n(h)}(m, w(n(h))\cdot\delta)]$

В отличие от парадигмы ERM, которую мы обсуждали в предыдущих главах, нас больше не интересует эмпирический риск $L_S(h)$, но в стремлении снизить ошибку оценивания мы хотим частично обменять смещение в сторону низкого эмпирического риска на смещение в сторону классов, для которых величина $\epsilon_{n(h)}(m, w(n(h))\cdot\delta)$ меньше.

Далее мы покажем, что парадигму SRM можно использовать для неравномерного обучения любого класса, являющегося счетным объединением равномерно сходящихся классов гипотез.

Теорема 7.5. *Пусть $\mathcal{H}$ – класс гипотез такой, что $\mathcal{H} = \bigcup_{n\in\mathbb{N}} \mathcal{H}_n$, где каждый класс $\mathcal{H}_n$ обладает свойством равномерной сходимости с функцией выборочной сложности $m^{\mathrm{UC}}_{\mathcal{H}_n}$. Пусть функция $w : \mathbb{N} \to [0, 1]$ определена следующим образом: $w(n) = 6/(\pi^2 n^2)$. Тогда $\mathcal{H}$ является неравномерно обучаемым по правилу SRM со скоростью*

$$m^{NUL}_{\mathcal{H}}(\epsilon, \delta, h) \le m^{UC}_{\mathcal{H}_{n(h)}}\left(\epsilon/2, \frac{6\delta}{(\pi n(h))^2}\right).$$

Доказательство. Пусть A – алгоритм SRM с весовой функцией w. Для любых $h \in \mathcal{H}$, ϵ и δ пусть $m \geq m^{\mathrm{UC}}_{\mathcal{H}_{n(h)}}(\epsilon, w(n(h))\delta)$. Используя тот факт, что $\sum_n w(n) = 1$, мы можем применить теорему 7.4 и получить, что с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ для любой $h' \in \mathcal{H}$ имеет место неравенство

$$L_{\mathcal{D}}(h') \leq L_S(h') + \epsilon_{n(h')}(m, w(n(h'))\delta).$$

Это справедливо, в частности, и для гипотезы $A(S)$, возвращенной правилом SRM. По определению SRM мы получаем, что

$$L_{\mathcal{D}}(A(S)) \leq \min_{h'}[L_S(h') + \epsilon_{n(h')}(m, w(n(h'))\delta)] \leq L_S(h) + \epsilon_{n(h)}(m, w(n(h))\delta).$$

Наконец, если $m \geq m^{\mathrm{UC}}_{\mathcal{H}_{n(h)}}(\epsilon/2, w(n(h))\delta)$, то, очевидно, $\epsilon_{n(h)}(m, w(n(h))\delta) \leq \epsilon/2$. Кроме того, из свойства равномерной сходимости каждого класса $\mathcal{H}_n$ мы имеем, что с вероятностью более $1 - \delta$,

$$L_S(h) \leq L_{\mathcal{D}}(h) + \epsilon/2.$$

Объединяя все вышесказанное, получаем, что $L_{\mathcal{D}}(A(S)) \leq L_{\mathcal{D}}(h) + \epsilon$, что и завершает доказательство. □

Отметим, что эта теорема также доказывает теорему 7.3.

Замечание 7.2 (отсутствие бесплатных завтраков для неравномерной обучаемости). Мы показали, что любое счетное объединение классов конечной VC-размерности допускает неравномерное обучение. Оказывается, что для любой бесконечной области образцов $\mathcal{X}$ класс всех бинарных функций на $\mathcal{X}$ не является счетным объединением классов конечной VC-размерности. Мы оставляем доказательство этого утверждения в качестве (нетривиального) упражнения (см. упражнение 7.5). Отсюда следует, что в некотором смысле теорема об отсутствии бесплатных завтраков имеет место и для неравномерного обучения: именно, если область определения бесконечна, то не существует алгоритма неравномерного обучения относительно класса всех детерминированных бинарных классификаторов (хотя для каждого такого классификатора существует тривиальный алгоритм его обучения – ERM относительно класса гипотез, содержащего только этот классификатор).

Интересно сравнить результат о неравномерной обучаемости, сформулированный в теореме 7.5, с задачей агностического PAC-обучения любого конкретного класса $\mathcal{H}_n$ по отдельности. Априорное знание, или смещение неравномерного обучаемого, для $\mathcal{H}$ слабее: он ищет модель во всем классе $\mathcal{H}$, а не ограничивается только одним конкретным $\mathcal{H}_n$. За такое ослабление априорного знания приходится расплачиваться увеличением выборочной сложности, необходимой для конкуренции с любой конкретной гипотезой $h \in \mathcal{H}_n$. Чтобы вычислить эту цену в конкретном случае, рассмотрим задачу бинарной классификации с бинарной функцией потери. Предположим, что для любого nVCdim$(\mathcal{H}_n) = n$. Тогда $m^{\mathrm{UC}}_{\mathcal{H}_n}(\epsilon, \delta) = C(n + \log(1/\delta))/\epsilon^2$ (где C – константа из теоремы 6.8), и прямое вычисление показывает, что

$$m^{NUL}_{\mathcal{H}}(\epsilon, \delta, h) - m^{UC}_{\mathcal{H}_n}(\epsilon/2, \delta) \leq 4C\frac{2\log(2n)}{\epsilon^2}.$$

Таким образом, стоимость ослабления априорного знания обучаемого с конкретного класса $\mathcal{H}_n$, который содержит целевую гипотезу h, до счетного объединения классов зависит от логарифма индекса первого класса, в котором находится h. Эта стоимость увеличивается с ростом индекса, что можно интерпретировать как ценность знания хорошего порядка гипотез в $\mathcal{H}$.

7.3. Минимальная длина описания и бритва Оккама

Пусть $\mathcal{H}$ – счетный класс гипотез. Тогда $\mathcal{H}$ можно записать в виде счетного объединения одноэлементных классов: $\mathcal{H} = \cup_{n\in\mathbb{N}}\{h_n\}$. В силу неравенства Хёфдинга (лемма 4.5) каждый одноэлементный класс обладает свойством равномерной сходимости со скоростью $m^{\mathrm{UC}}(\epsilon, \delta) = \dfrac{\log(2/\delta)}{2\epsilon^2}$. Следовательно, функция ϵ_n из формулы (7.1) принимает вид $\epsilon_n(m, \delta) = \sqrt{\dfrac{\log(2/\delta)}{2m}}$, а правило SRM превращается в

$$\operatorname*{arg\,min}_{h_n\in\mathcal{H}}\left[L_S(h) + \sqrt{\frac{-\log(w(n)) + \log(2/\delta)}{2m}}\right].$$

Можно вместо этого считать w функцией из $\mathcal{H}$ в [0, 1], и тогда правило SRM принимает вид

$$\operatorname*{arg\,min}_{h\in\mathcal{H}}\left[L_S(h) + \sqrt{\frac{-\log(w(n)) + \log(2/\delta)}{2m}}\right].$$

Следовательно, в этом случае априорное знание полностью определяется весами, назначенными каждой гипотезе. Мы назначаем большие веса гипотезам, которые, на наш взгляд, имеют больше шансов быть правильными, а алгоритм обучения предпочитает гипотезы с большими весами.

В этом разделе мы обсудим один удобный способ определения весовой функции для $\mathcal{H}$, который основан на длине описаний гипотез. Если имеется класс гипотез, то можно поставить вопрос о том, как описать, или представить каждую входящую в него гипотезу. Естественно, мы фиксируем некоторый язык описания. Это может быть английский язык, язык программирования или какой-то набор математических формул. На любом языке описание состоит из конечных строк символов (или знаков), взятых из некоторого фиксированного алфавита. Теперь формализуем эти идеи.

Пусть $\mathcal{H}$ – класс гипотез, которые мы хотим описать. Зафиксируем конечное множество символов (или «знаков»), которое будем называть алфавитом. Для конкретности положим $\Sigma = \{0, 1\}$. Строкой называется конечная последовательность символов из Σ; например, $\sigma = (0, 1, 1, 1, 0)$ – строка длины 5. Будем обозначать $|\sigma|$ длину строки. Множество всех строк конечной длины обозначается Σ^*. Языком описания для $\mathcal{H}$ называется функция $d : \mathcal{H} \to \Sigma^*$, отображающая каждую гипотезу h из $\mathcal{H}$ в строку $d(h)$. При этом строка $d(h)$ называется «описанием h», а ее длина обозначается $|h|$.

Потребуем, чтобы язык описания был беспрефиксным, т. е. для любых двух гипотез h и h' строка $d(h)$ не является префиксом $d(h')$. Это значит, что никакая строка $d(h)$ не может совпадать с первыми $|h|$ символами более длинной строки $d(h')$. Беспрефиксные наборы строк обладают следующим комбинаторным свойством:

Лемма 7.6 (неравенство Крафта). *Если $\mathcal{S} \subseteq \{0, 1\}^*$ – беспрефиксное множество строк, то*

$$\sum_{\sigma \in \mathcal{S}} \frac{1}{2^{|\sigma|}} \leq 1.$$

Доказательство. Определим распределение вероятностей на элементах $\mathcal{S}$ следующим образом: повторно подбрасываем симметричную монету, стороны которой помечены 0 и 1, до тех пор, пока последовательность исходов является элементом $\mathcal{S}$, после чего останавливаемся. Для любого $\boldsymbol{\sigma} \in \mathcal{S}$ обозначим $P(\boldsymbol{\sigma})$ вероятность того, что такой процесс породит строку $\boldsymbol{\sigma}$. Поскольку множество $\mathcal{S}$ беспрефиксное, то для любого $\boldsymbol{\sigma} \in \mathcal{S}$, если исходы подбрасывания монеты повторяют биты $\boldsymbol{\sigma}$, то процесс остановится только тогда, когда последовательность исходов совпадает с $\boldsymbol{\sigma}$. Поэтому для любого $\boldsymbol{\sigma} \in \mathcal{S}$ $P(\boldsymbol{\sigma}) = 1/2^{|\sigma|}$. Так как сумма вероятностей никогда не превосходит 1, то мы доказали требуемый результат. □

В свете неравенства Крафта любой беспрефиксный язык описания класса гипотез $\mathcal{H}$ порождает весовую функцию w над этим классом гипотез – мы просто полагаем $w(h) = 1/2^{|h|}$. Это наблюдение немедленно приводит к следующему результату.

Теорема 7.7. *Пусть $\mathcal{H}$ – класс гипотез, и $d : \mathcal{H} \to \{0, 1\}^*$ – беспрефиксный язык описания для $\mathcal{H}$. Тогда для любого размера выборки m, любого параметра уверенности $\delta > 0$ и любого распределения вероятностей $\mathcal{D}$ с вероятностью более $1 - \delta$ для случайной выборках $S \sim \mathcal{D}^m$ справедливо следующее утверждение*

$$\forall h \in \mathcal{H},\ L_{\mathcal{D}}(h) \leq L_S(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}},$$

где $|h|$ – длина $d(h)$.

Доказательство. Выберем $w(h) = 1/2^{|h|}$, применим теорему 7.4 с $\epsilon_n(m, \delta) = \sqrt{\frac{\ln(2/\delta)}{2m}}$ и заметим, что $\ln(2^{|h|}) = |h|\ln(2) \leq |h|$, □

Как и в случае с теоремой 7.4, этот результат ведет к парадигме обучения для $\mathcal{H}$ – при заданном обучающем наборе S искать гипотезу $h \in \mathcal{H}$, которая минимизирует верхнюю границу $L_S(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}}$. В частности, предлагается обменять эмпирический риск на сохранение длины описания. Таким образом, мы приходим к парадигме обучения «минимальная длина описания».

Минимальная длина описания (MDL)

априорное знание:

$\mathcal{H}$ – счетный класс гипотез

$\mathcal{H}$ описывается беспрефиксным языком над алфавитом {0, 1}

Для любого $h \in \mathcal{H}$ $|h|$ – длина представления h

вход: обучающий набор $S \sim \mathcal{D}^m$, уверенность δ

выход: $h \in \operatorname{argmin}_{h \in \mathcal{H}} \left[L_S(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}} \right]$

Пример 7.3. Пусть $\mathcal{H}$ – класс всех предикторов, которые можно реализовать на некотором языке программирования, например, C++. Представим каждую программу строкой нулей и единиц, получающейся в результате применения к программе команды gzip (это беспрефиксный язык описания над алфавитом {0, 1}). Тогда $|h|$ – просто длина (в битах) результата применения gzip к программе на C++, соответствующей h.

7.3.1. Бритва Оккама

Теорема 7.7 говорит, что при наличии двух гипотез с одинаковым эмпирическим риском истинный риск гипотезы с более коротким описанием можно ограничить меньшим значением. Можно считать, что этот результат несет следующее философское послание:

Чем короче объяснение (т. е. чем меньше длина описания гипотезы), тем больше шансов, что оно правильно.

Это хорошо известный принцип, названный бритвой Оккама в честь Уильяма Оккамского, английского философа XIV века, который, как считается, первым сформулировал его явно. В этом разделе мы приводим одно из его возможных обоснований. Неравенство в теореме 7.7 показывает, что, чем сложнее гипотеза (в смысле длины описания), тем больше размер выборки, которую она должна аппроксимировать, чтобы гарантировать небольшой истинный риск, $L_{\mathcal{D}}(h)$.

При более пристальном рассмотрении наша интерпретация бритвы Оккама может показаться несколько проблематичной. В том контексте, в котором принцип бритвы Оккама обычно применяется в науке, сложность измеряется в соответствии с естественным языком, мы же допускаем произвольный абстрактный язык описания. Предположим, что имеется две гипотезы такие, что $|h'|$ много меньше $|h|$. В силу доказанного выше результата, если обе гипотезы дают одинаковую ошибку на заданном обучающем наборе S, то истинная ошибка h может оказаться намного больше истинной ошибки h', поэтому следует предпочесть h', а не h. Однако мы могли бы выбрать другой язык описания, на котором гипотезе h будет соответствовать строка длины 3, а гипотезе h' – строка длины 100000. И вот вдруг оказывается, что предпочесть-то следует h. Но ведь это те же самые h и h', для которых мы двумя предложениями выше были уверены, что h' лучше. В чем же подвох?

На самом деле с точки зрения обобщаемости между гипотезами нет никакого внутреннего различия. Ключевой аспект здесь – зависимость между начальным выбором языка (или порядка предпочтения гипотез) и обучающим набором. Как мы знаем из неравенства Хёфдинга (формула 4.2), если зафиксировать любую гипотезу *до* наблюдения данных, то мы гарантированно получим относительно небольшую ошибку оценивания $L_{\mathcal{D}}(h) \leq L_S(h) + \sqrt{\frac{\ln(2/\delta)}{2m}}$. Выбор языка описания (или, эквивалентно, весов гипотез) – это слабая форма фиксации гипотезы. Вместо того, чтобы фиксировать единственную гипотезу, мы готовы принять сразу много. Коль скоро это делается независимо от обучающей выборки, оценка обобщаемости остается справедливой. Выбор той единственной гипотезы, которую нужно вычислять для выборки, может быть произвольным, и то же самое верно в отношении выбора языка описания.

7.4. Другие концепции обучаемости – согласованность

Понятие обучаемости можно еще ослабить, разрешив размеру потребной выборки зависеть не только от ϵ, δ и h, но и от истинного порождающего данные распределения $\mathcal{D}$ (из которого выбираются обучающие примеры и которое определяет риск). Такой тип гарантии качества отражен в понятии *согласованности* (consistency)[1] в качестве правила обучения.

Определение 7.8 (согласованность). Пусть Z – область определения, $\mathcal{P}$ – множество распределений вероятностей на Z, а $\mathcal{H}$ – класс гипотез. Правило обучения A называется *согласованным* с $\mathcal{H}$ и $\mathcal{P}$, если существует функция $m^{\text{CON}}_{\mathcal{H}} : (0, 1)^2 \times \mathcal{H} \times \mathcal{P} \to \mathbb{N}$ такая, что для любых $\epsilon, \delta \in (0, 1)$, $h \in \mathcal{H}$ и $\mathcal{D} \in \mathcal{P}$ справедливо утверждение: если $m \geq m^{\text{CON}}_{\mathcal{H}}(\epsilon, \delta, h, \mathcal{D})$, то с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ имеет место неравенство

$$L(A(S)) \leq L_{\mathcal{D}}(h) + \epsilon.$$

Если $\mathcal{P}$ – множество всех распределений[2], то говорят, что A *универсально согласован* с $\mathcal{H}$.

Конечно, понятие согласованности является ослаблением введенного ранее понятия неравномерной обучаемости. Очевидно, что если алгоритм неравномерно обучает класс $\mathcal{H}$, то он универсально согласован с этим классом. Ослабление строгое в том смысле, что существуют согласованные правила обучения, не являющиеся успешными неравномерными обучаемыми. Например, алгоритм

[1] В литературе согласованность часто определяют с помощью сходимости вероятности (соответствует слабой согласованности) или сходимости почти наверное (соответствует сильной согласованности).

[2] Формально мы предполагаем, что Z снабжено некоторой σ-алгеброй подмножеств Ω и под «всеми распределениями» понимаем все распределения вероятностей, для которых Ω содержится в ассоциированном семействе измеримых подмножеств.

`Memorize`, определенный в примере 7.4 ниже, универсально согласован с классом всех бинарных классификаторов над $\mathbb{N}$. Однако, как мы доказали выше, этот класс не является неравномерно обучаемым.

Пример 7.4. Рассмотрим алгоритм классификации `Memorize`, определенный следующим образом. Алгоритм запоминает обучающие примеры и, когда ему предъявляют тестовый пример x, он предсказывает метку, которая чаще всего встречается среди помеченных образцов x, встретившихся в обучающем наборе (и некоторую фиксированную метку по умолчанию, если x не встречается в обучающем наборе). Можно показать (см. упражнение 7.6), что этот алгоритм универсально согласован для любой счетной области определения $\mathcal{X}$ и конечного множества меток $\mathcal{Y}$ (относительно бинарной функции потери).

Интуитивно неочевидно, почему алгоритм `Memorize` следует рассматривать как *обучаемого*, поскольку в нем отсутствует идея обобщения, т. е. использования наблюденных данных для предсказания меток ранее не предъявлявшихся примеров. Поэтому тот факт, что `Memorize` – согласованный алгоритм для класса всех функций над любой счетной областью определения, ставит под сомнение полезность гарантий, которые дает согласованность. Более того, проницательный читатель, вероятно, заметил, что «плохой обучаемый», с которым мы познакомились в главе 2 и который приводил к переобучению, как раз и есть алгоритм `Memorize`. В следующем разделе мы обсудим значимость различных понятий обучаемости и вернемся к теореме об отсутствии бесплатных завтраков в свете различных определений этого понятия.

7.5. Обсуждение различных понятий обучаемости

Мы дали три определения обучаемости, а теперь обсудим их полезность. Как обычно, полезность математического определения зависит от того, для чего оно нужно. Поэтому перечислим несколько возможных целей, которых мы пытаемся достичь, определяя обучаемость, и обсудим полезность различных определений в свете этих целей.

Что такое риск обученной гипотезы?

Первая возможная цель предоставления гарантий качества алгоритма обучения – ограничение риска выходного предиктора. В этом отношении и PAC-обучение, и неравномерное обучение дают верхнюю границу истинного риска обученной гипотезы на основе ее эмпирического риска. Гарантии согласованности такой границы не дают. Однако всегда можно оценить риск выходного предиктора с помощью контрольного набора (как будет описано в главе 11).

Сколько примеров нужно, чтобы получить гипотезу не хуже наилучшей в классе $\mathcal{H}$?

Ища подходы к проблеме обучения, мы должны ответить на естественный вопрос: сколько примеров нужно набрать, чтобы обучить алгоритм? PAC-обучение

дает самый точный ответ. Но ни для неравномерного обучения, ни для согласованности заранее неизвестно, сколько примеров понадобится для обучения $\mathcal{H}$. В случае неравномерного обучения их количество зависит от наилучшей гипотезы в $\mathcal{H}$, а в случае согласованности – еще и от истинного распределения. В этом смысле PAC-обучение – единственное полезное определение обучаемости. С другой стороны, следует иметь в виду, что даже если ошибка оценивания обученного предиктора мала, его риск все равно может быть велик, если у $\mathcal{H}$ большая ошибка аппроксимации. Таким образом, на вопрос «сколько примеров необходимо, чтобы предиктор был не хуже оптимального байесовского?» даже гарантии PAC не дают твердого ответа. Это отражает тот факт, что полезность PAC-обучения зависит от качества априорных знаний.

Гарантии PAC также помогают понять, что делать дальше, если алгоритм обучения возвращает гипотезу с большим риском, т. к. мы можем ограничить сверху часть ошибки, связанную с оцениванием, и тем самым узнать, какую часть ошибки отнести на аппроксимацию. Если ошибка аппроксимации велика, то ясно, что следует взять другой класс гипотез. Аналогично, если неравномерный алгоритм терпит неудачу, то можно рассмотреть другую весовую функцию. Но если терпит неудачу согласованный алгоритм, то мы понятия не имеем, связано это с ошибкой аппроксимации или оценивания. Более того, даже если мы уверены, что проблема возникла из-за ошибки оценивания, все равно неизвестно, сколько примеров необходимо, чтобы уменьшить эту ошибку.

Как обучать? Как выразить априорные знания?

Быть может, самый полезный аспект теории обучения – ответ на вопрос «как обучать»? Определение PAC-обучения налагает ограничение на возможность обучения (в виде теоремы об отсутствии бесплатных завтраков) и привносит необходимость априорного знания. Оно дает точный способ кодирования априорного знания посредством выбора класса гипотез, а, когда такой выбор сделан, к нашим услугам общее правило обучения – ERM. Определение неравномерной обучаемости тоже дает точный способ закодировать априорное знание путем задания весов (подмножеств) гипотез из $\mathcal{H}$. После того как веса заданы, мы опять-таки имеем общее правило обучения – SRM. Правило SRM полезно также в задачах выбора модели, когда имеется частичное априорное знание. О выборе моделей мы будем подробно говорить в главе 11, а здесь приведем лишь краткий пример.

Рассмотрим задачу аппроксимации данных полиномом от одной переменной, т. е. наша цель – обучить функцию $h : \mathbb{R} \to \mathbb{R}$, а в качестве априорного знания мы рассматриваем класс полиномиальных гипотез. Однако мы не уверены, при какой степени d получатся наилучшие результаты для имеющегося набора данных: если степень мала, то может не получиться подогнать полином к данным (т. е. налицо большая ошибка аппроксимации), а если велика, то велик риск переобучения (т. е. большая ошибка оценивания). На рисунках ниже мы изобразили результат аппроксимации одного и того же обучающего набора полиномами степени 2, 3 и 10.

Легко видеть, что эмпирический риск уменьшается с увеличением степени. Следовательно, если взять в качестве $\mathcal{H}$ класс всех полиномов степени не выше 10, то правило ERM для этого класса вернет полином десятой степени, что приведет к переобучению. С другой стороны, если выбрать слишком малый класс гипотез, скажем, класс полиномов степени не выше 2, то правило ERM оказывается недообученным (велика ошибка аппроксимации). Но можно было бы использовать правило SRM на множестве всех полиномов, упорядочив подмножества $\mathcal{H}$ по степени. Это дало бы полином третьей степени, потому что для него комбинация эмпирического риска и верхней границы ошибки оценивания наименьшая. Другими словами, правило SRM позволяет выбрать хорошую модель на основе самих данных. За эту гибкость мы расплачиваемся тем, что заранее не знаем, сколько примеров понадобится, чтобы конкурировать с лучшей гипотезой в $\mathcal{H}$ (и плюс к тому небольшим увеличением ошибки оценивания по сравнению с PAC-обучением для полиномов оптимальной степени).

В отличие от понятий PAC-обучаемости и неравномерной обучаемости, определение согласованности не дает ни естественной парадигмы обучения, ни способа кодирования априорных знаний. На самом деле, во многих случаях априорные знания вообще не нужны. Например, мы видели, что алгоритм `Memorize`, который, как подсказывает интуиция, не стоило бы называть алгоритмом обучения, – это согласованный алгоритм для любого класса, определенного над счетной областью, и конечного множества меток. Можно сделать вывод, что согласованность – очень слабое требование.

Какой алгоритм обучения предпочесть?

Можно возразить, что хотя согласованность и слабое требование, желательно, чтобы алгоритм был согласован с множеством всех функций из $\mathcal{X}$ в $\mathcal{Y}$, т. к. это дает нам гарантию, что при достаточно большом количестве примеров мы всегда получим предиктор не хуже оптимального байесовского. Поэтому, если имеется два алгоритма, один из которых согласован, а другой нет, то следует предпочесть согласованный алгоритм. Однако в этом рассуждении два недостатка. Во-первых, может оказаться, что для наиболее «естественных» распределений, наблюдаемых на практике, выборочная сложность согласованного алгоритма настолько велика, что мы не сможем набрать достаточно примеров для получения гарантии. Во-вторых, не так уж трудно сделать любого PAC- или неравномерного обучаемого согласованным с классом всех функций из $\mathcal{X}$ в $\mathcal{Y}$. Конкретно, рассмотрим счетную область образцов $\mathcal{X}$, конечное множество меток $\mathcal{Y}$ и класс гипотез $\mathcal{H}$, состоящий из функций, отображающих $\mathcal{X}$ в $\mathcal{Y}$. Мы можем сделать любого неравномерного обучаемого для $\mathcal{H}$ совместимым с классом *всех* классификаторов

из $\mathcal{X}$ в $\mathcal{Y}$, применив следующий простой прием: получив обучающий набор, мы сначала прогоним неравномерного обучаемого на этом наборе, а затем получим верхнюю границу истинного риска обученного предиктора. Если эта граница достаточно мала, то все хорошо. В противном случае мы обращаемся к алгоритму `Memorize`. Эта нехитрая модификация делает алгоритм совместимым с множеством всех функций из $\mathcal{X}$ в $\mathcal{Y}$. Поскольку сделать любой алгоритм совместимым так просто, неразумно предпочитать один алгоритм другому только из соображений совместимости.

7.5.1. Еще раз о теореме об отсутствии бесплатных завтраков

Напомним, что теорема об отсутствии бесплатных завтраков (теорема 5.1) говорит, что никакой алгоритм не может обучить класс всех классификаторов над бесконечной областью определения. С другой стороны, в этой главе мы видели, что алгоритм `Memorize` совместим с классом всех классификаторов над счетно бесконечной областью. Чтобы понять, почему эти два утверждения не противоречат друг другу, сначала вспомним формулировку теоремы об отсутствии бесплатных завтраков.

Пусть $\mathcal{X}$ – счетная область образцов, и $\mathcal{Y} = \{\pm 1\}$. Теорема об отсутствии бесплатных завтраков утверждает следующее: для любого алгоритма A и размера обучающего набора m существуют распределение на $\mathcal{X}$ и функция $h^* : \mathcal{X} \to \mathcal{Y}$ такие, что если A получает выборку m независимых и одинаково распределенных примеров, помеченных функцией h^*, то A с высокой вероятностью вернет классификатор с большой ошибкой.

Из согласованности алгоритма `Memorize` вытекает следующее: для любого распределения на $\mathcal{X}$ и функции пометки $h^* : \mathcal{X} \to \mathcal{Y}$ существует размер обучающего набора m (зависящий от распределения и от h^*) такой, что если `Memorize` получает не менее m примеров, то он с высокой вероятностью вернет классификатор с малой ошибкой.

Мы видим, что в теореме об отсутствии бесплатных завтраков сначала фиксируется размер обучающего набора, а затем ищется распределение и функция пометки, плохие для этого размера. Напротив, в случае парадигмы согласованности мы сначала фиксируем распределение и функцию пометки, а только потом ищем размер обучающего набора, достаточный для обучения этого конкретного распределения и функции.

7.6. Резюме

Мы ввели понятие неравномерной обучаемости как ослабление PAC-обучаемости, а понятие согласованности – как ослабление неравномерной обучаемости. Это означает, что даже классы бесконечной VC-размерности могут быть обучаемыми в некотором более слабом смысле. Мы обсудили полезность различных определений обучаемости.

Для счетных классов гипотез можно применить идею минимальной длины описания (MDL), когда предпочтение отдается гипотезам с более короткими описаниями – в соответствии с принципом бритвы Оккама. Интересный пример

дает класс гипотез, содержащий все предикторы, которые можно реализовать на C++ (или любом другом языке программирования), – он допускает (неравномерное) обучение с помощью техники MDL.

Можно предположить, что класс всех предикторов, реализуемых на C++, – мощный класс функций, который, вероятно, содержит все, что можно надеяться обучить на практике. Способность обучить такой класс впечатляет и, казалось бы, на этой главе книгу можно было бы и закончить. Но это не так, потому что у обучения имеется и вычислительный аспект, т. е. время, необходимое для применения правила обучения. Так, чтобы реализовать парадигму MDL в применении к программам на C++, мы должны были бы выполнить исчерпывающий поиск в множестве всех таких программ, что займет вечность. Даже реализация парадигмы ERM для всех программ на C++ с длиной описания не более 1000 бит потребовала бы поиска среди 2^{1000} гипотез. Хотя выборочная сложность обучения этого класса составляет всего $(1000 + \log(2/\delta))\epsilon^2$, время работы превышает 2^{1000}. Это огромное число – гораздо больше числа атомов в видимой части Вселенной. В следующей главе мы формально определим вычислительную сложность обучения. А во второй части книги изучим классы гипотез, для которых можно эффективно реализовать схемы ERM и SRM.

7.7. Библиографические сведения

Связь нашего определения неравномерной обучаемости с бритвой Оккама исследована в работе Blumer, Ehrenfeucht, Haussler and Warmuth (1987). Концепция SRM введена в работах (Vapnik & Chervonenkis, 1974; Vapnik, 1995). Концепция MDL впервые описана в работах (Rissanen, 1978; Rissanen, 1983). Связь между SRM и MDL обсуждается в работе Vapnik (1995). Эти понятия тесно связаны с понятием *регуляризации* (см., например, Tikhonov, 1943). Мы будем подробно изучать регуляризацию во второй части книги.

Понятие согласованности оценок восходит к работе Fisher (1922). Наше изложение следует работе Steinwart and Christmann (2008), в которой также доказано несколько теорем об отсутствии бесплатных завтраков.

7.8. Упражнения

7.1. Докажите, что для любого конечного класса $\mathcal{H}$ и любого языка описания $d : \mathcal{H} \to \{0, 1\}^*$ VC-размерность $\mathcal{H}$ не превышает $2\sup\{|d(h)| : h \in \mathcal{H}\}$ – максимальной длины описания предиктора из $\mathcal{H}$. Более того, если d – беспрефиксное описание, то $\mathrm{VCdim}(\mathcal{H}) \le \sup\{|d(h)| : h \in \mathcal{H}\}$.

7.2. Пусть $\mathcal{H} = \{h_n : n \in \mathbb{N}\}$ – счетный класс гипотез для бинарной классификации. Покажите, что невозможно назначить веса гипотезам из $\mathcal{H}$, так чтобы:

- $\mathcal{H}$ можно было неравномерно обучить с такими весами. То есть весовая функция $w : \mathcal{H} \to [0, 1]$ должна удовлетворять условию $\sum_{h \in \mathcal{H}} w(h) \le 1$;
- веса были монотонно неубывающими. То есть если $i < j$, то $w(h_i) \le w(h_j)$.

7.3. Рассмотрим класс гипотез $\mathcal{H} = \bigcup_{n=1}^{\infty} \mathcal{H}_n$, где все $\mathcal{H}_n$ конечны. Найдите весовую функцию $w : \mathcal{H} \to [0, 1]$ такую, что $\sum_{h \in \mathcal{H}} w(h) \leq 1$, и для всех $h \in \mathcal{H} w(h)$ определяется величинами $n(h) = \min\{n : h \in \mathcal{H}_n\}$ и $|\mathcal{H}_{n(h)}|$.

- (*) Определите такую функцию w, когда для всех n классы $\mathcal{H}_n$ счетны (и, возможно, бесконечны).

7.4. Пусть $\mathcal{H}$ – некоторый класс гипотез. Для любого $h \in \mathcal{H}$ обозначим $|h|$ длину описания h в некотором фиксированном языке описания. Рассмотрим парадигму обучения MDL, в которой алгоритм возвращает гипотезу

$$h_S \in \operatorname{argmin}_{h \in \mathcal{H}} \left[L_S(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}} \right],$$

где S – выборка размера m. Для любого $B > 0$ положим $\mathcal{H}_B = \{h \in \mathcal{H} : |h| \leq B\}$ и определим

$$h_B^* = \arg\min_{h \in \mathcal{H}_B} L_{\mathcal{D}}(h).$$

Найдите верхнюю границу $L_{\mathcal{D}}(h_S) - L_{\mathcal{D}}(h_B^*)$, выраженную в терминах B, параметра уверенности δ и размера обучающего набора m.

- *Примечание*: в литературе такие границы называются оракульными неравенствами. Мы хотим оценить, насколько наш классификатор хорош по сравнению с эталонным (или «оракулом») h_B^*.

7.5. В этом упражнении мы хотим доказать теорему об отсутствии бесплатных завтраков для неравномерной обучаемости: для любой бесконечной области образцов класс всех функций не является обучаемым даже при ослабленной неравномерной парадигме обучения.

Напомним, что алгоритм A *неравномерно обучает* класс гипотез $\mathcal{H}$, если существует функция $m_{\mathcal{H}}^{\mathrm{NUL}} : (0, 1)^2 \times \mathcal{H} \to \mathbb{N}$ такая, что для любых $\epsilon, \delta \in (0, 1)$ и любой $h \in \mathcal{H}$ справедливо утверждение: если $m \geq m_{\mathcal{H}}^{\mathrm{NUL}}(\epsilon, \delta, h)$, то для любого распределения $\mathcal{D}$ с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ имеет место неравенство

$$L(A(S)) \leq L_{\mathcal{D}}(h) + \epsilon.$$

Если такой алгоритм существует, то говорят, что класс $\mathcal{H}$ является *неравномерно обучаемым*.

1. Пусть A – неравномерный алгоритм обучения для класса $\mathcal{H}$. Для любого $n \in \mathbb{N}$ определим $\mathcal{H}_n^A = \{h \in \mathcal{H} : m^{\mathrm{NUL}}(0{,}1,\ 0{,}1,\ h) \leq n\}$. Докажите, что VC-размерность каждого такого класса $\mathcal{H}_n$ конечна.
2. Докажите, что если класс $\mathcal{H}$ является неравномерно обучаемым, то существуют классы $\mathcal{H}_n$ такие, что $\mathcal{H} = \bigcup_{n \in \mathbb{N}} \mathcal{H}_n$ и для любого $n \in \mathbb{N}$ $\mathrm{VCdim}(\mathcal{H}_n)$ конечна.
3. Пусть $\mathcal{H}$ – класс, разбивающий бесконечное множество. Тогда для любой последовательности классов $(\mathcal{H}_n : n \in \mathbb{N})$ такой, что $\mathcal{H} = \bigcup_{n \in \mathbb{N}} \mathcal{H}_n$ существует такое n, для которого $\mathrm{VCdim}(\mathcal{H}_n) = \infty$.
   Указание: если дан класс $\mathcal{H}$, разбивающий некоторое бесконечное множество K, и последовательность классов $(\mathcal{H}_n : n \in \mathbb{N})$ конечной VC-размерности,

то начните с определения подмножеств $K_n \subseteq K$ таких, что для любого n $|K_n| > \mathrm{VCdim}(\mathcal{H}_n)$ и для любых $n \neq m$ $K_n \cap K_m = \emptyset$. Затем выберите для каждого такого K_n функцию $f_n : K_n \to \{0, 1\}$ такую, что никакая гипотеза $h \in \mathcal{H}_n$ не согласуется с f_n в области K_n. Наконец, определите $f : \mathcal{X} \to \{0, 1\}$, объединив все f_n, и докажите, что $f \in (\mathcal{H} \setminus \bigcup_{n \in \mathbb{N}} \mathcal{H}_n)$.

4. Постройте класс $\mathcal{H}_1$ функций, отображающих единичный отрезок [0, 1] в множество {0, 1}, который является неравномерно обучаемым, но не является PAC-обучаемым.
5. Постройте класс $\mathcal{H}_2$ функций, отображающих единичный отрезок [0, 1] в множество {0, 1}, который не является неравномерно обучаемым.

7.6. В этом упражнении мы хотим доказать, что алгоритм `Memorize` является согласованным обучаемым для любого класса (бинарных) функций над любой счетной областью образцов. Пусть $\mathcal{X}$ – счетная область образцов, а $\mathcal{D}$ – распределение вероятностей на $\mathcal{X}$.

1. Пусть $\{x_i : i \in \mathbb{N}\}$ – перечисление элементов $\mathcal{X}$ такое, что для любых $i \leq j$ $\mathcal{D}(\{x_i\}) \leq \mathcal{D}(\{x_j\})$. Докажите, что

$$\lim_{n \to \infty} \sum_{i \geq n} \mathcal{D}(\{x_i\}) = 0.$$

2. Докажите, что для любого заданного $\epsilon > 0$ существует $\epsilon_{\mathcal{D}} > 0$ такое, что

$$\mathcal{D}(\{x \in \mathcal{X} : \mathcal{D}(\{x\}) < \epsilon_{\mathcal{D}}\}) < \epsilon.$$

3. Докажите, что для любого $\eta > 0$ справедливо утверждение: если n таково, что $\mathcal{D}(\{x_i\}) < \eta$ для всех $i > n$, то для любого $m \in \mathbb{N}$

$$\mathbb{P}_{S \sim \mathcal{D}^m} [\exists x_i : (\mathcal{D}(\{x\}) > \eta \text{ и } x_i \notin S)] \leq n e^{-\eta m}.$$

4. Сделайте вывод, что если $\mathcal{X}$ счетна, то для любого распределения вероятностей $\mathcal{D}$ на $\mathcal{X}$ существует функция $m_{\mathcal{D}}$: $(0, 1) \times (0, 1) \to \mathbb{N}$ такая, что для любых $\epsilon, \delta > 0$ справедливо утверждение: если $m > m_{\mathcal{D}}(\epsilon, \delta)$, то

$$\mathbb{P}_{S \sim \mathcal{D}^m} [\mathcal{D}(\{x : x \notin S\}) > \epsilon] < \delta.$$

5. Докажите, что `Memorize` – согласованный алгоритм обучения для любого класса (бинарных) функций над любой счетной областью образцов.

Глава 8

ВРЕМЯ ОБУЧЕНИЯ

До сих пор в этой книге мы изучали обучение со статистической точки зрения, т. е. сколько примеров нужно для обучения. Иначе говоря, нас интересовал объем информации, необходимой для обучения. Но при рассмотрении автоматизированного обучения не меньшую роль в определении сложности задачи играют вычислительные ресурсы: какой объем *вычислений* необходим для решения задачи обучения. После того как обучаемому предъявлена достаточно большая обучающая выборка, наступает черед вычислений, призванных выделить подходящую гипотезу или определить метку тестового примера. Вычислительные ресурсы критически важны для любого применения машинного обучения на практике. Эти два типа ресурсов мы называем *выборочной сложностью* и *вычислительной сложностью*. В настоящей главе мы обратимся к вычислительной сложности обучения.

Вычислительную сложность обучения следует рассматривать в более широком контексте вычислительной сложности алгоритмов вообще. Эта область хорошо изучена; см., например, (Sipser, 2006). Следующие далее вводные замечания содержат сводку основных идей общей теории, имеющих прямое отношение к теме нашего обсуждения.

Фактическое время работы (в секундах) алгоритма зависит от конкретной машины, на которой этот алгоритм реализован (например, от тактовой частоты процессора). Чтобы избежать зависимости от конкретной машины, принято анализировать асимптотическое время работы алгоритмов. Например, мы говорим, что вычислительная сложность алгоритма сортировки списка n элементов слиянием равна $O(n\log(n))$. Это означает, что алгоритм можно реализовать на любой машине, которая удовлетворяет требованиям некоторой абстрактной модели вычислений, и фактическое время работы в секундах будет удовлетворять следующему условию: существуют константы c и n_0, которые могут зависеть от конкретной машины, такие, что для любого $n > n_0$ время сортировки любых n элементов не будет превышать $cn\log(n)$. Часто употребляют термин *практически осуществимый*, или *эффективно вычислимый* для задач, которые могут быть реализованы алгоритмом, время работы которого равно $O(p(n))$ для некоторой полиномиальной функции p. Следует отметить, что этот вид анализа зависит от определения того, что такое входной размер n любого объекта, к которому применяется алгоритм. Для «чисто алгоритмических» задач, обсуждаемых в литературе по вычислительной сложности, входной размер четко определен; алгоритм

получает входной объект, скажем, подлежащий сортировке список или подлежащую вычислению арифметическую операцию, четко определенного размера (скажем, количество бит в двоичном представлении). В задачах машинного обучения понятие входного размера не столь однозначно. Алгоритм стремится выявить паттерны в наборе данных и имеет доступ только к случайным выборкам из этих данных.

Мы начнем главу с обсуждения этого вопроса и определим вычислительную сложность обучения. Для хорошо подготовленных студентов мы приведем также формальное определение со всеми подробностями. Затем мы перейдем к вычислительной сложности реализации правила ERM. Сначала приведем несколько примеров классов гипотез, к которым правило ERM можно эффективно применить, а затем рассмотрим ряд случаев, в которых, несмотря на то, что класс эффективно обучаем, реализация ERM вычислительно трудна. Наконец, мы кратко обсудим, как можно доказать трудность данной задачи обучения, т. е. отсутствие алгоритма, который мог бы решить ее эффективно.

8.1. Вычислительная сложность обучения

Напомним, что алгоритм обучения имеет доступ к множеству примеров Z, классу гипотез $\mathcal{H}$, функции потерь ℓ и обучающему набору взятых из Z примеров, которые независимы и одинаково распределены с некоторым неизвестным распределением $\mathcal{D}$. При заданных параметрах ϵ, δ алгоритм должен вывести гипотезу h такую, что с вероятностью не менее $1 - \delta$

$$L_{\mathcal{D}}(h) \leq \arg\min_{h \in \mathcal{H}_B} L_{\mathcal{D}}(h).$$

Выше уже отмечалось, что фактическое время работы алгоритма в секундах зависит от конкретной машины. Для проведения машинно-независимого анализа мы применим стандартный подход теории вычислительной сложности. Во-первых, мы опираемся на понятие абстрактной машины, например машины Тьюринга (или машины Тьюринга на множестве вещественных чисел (Blum, Shub & Smale, 1989). Во-вторых, мы анализируем асимптотику времени работы, игнорируя постоянные множители; таким образом, конкретная машина несущественна при условии, что она реализует абстрактную. Обычно оценивается асимптотика относительно размера входных данных алгоритма. Например, в случае алгоритма сортировки слиянием время работы оценивается как функция от количества сортируемых элементов.

В контексте алгоритмов обучения четкого понятия «размера входных данных» не существует. Можно было бы определить размер входных данных как размер обучающего набора, подаваемого алгоритму, но особого смысла в этом нет. Если предъявить алгоритму очень большое количество примеров, гораздо больше, чем выборочная сложность задачи обучения, то алгоритм может просто игнорировать лишние примеры. Поэтому увеличение размера обучающего набора не делает проблему обучения более трудной, и, следовательно, время работы алгоритма не увеличивается с ростом этого размера. Но мы все же можем анализировать время работы как функцию от естественных параметров проблемы,

например, целевой верности, уверенности в достижении этой верности, размерности области образцов или некоторых мер сложности класса гипотез, из которого выбирается результат алгоритма.

Для иллюстрации рассмотрим алгоритм обучения осепараллельных прямоугольников. Конкретная проблема описывается параметрами ϵ, δ и размерностью пространства образцов. Мы можем определить последовательность проблем типа «обучение прямоугольника», зафиксировав ϵ, δ и меняя размерность: $d = 2, 3, 4, \ldots$. Можно также определить другую последовательность «обучение прямоугольника», зафиксировав d, δ и меняя целевую верность: $\epsilon = 1/2, 1/3, \ldots$. Разумеется, никто не мешает выбрать другие последовательности такого рода. Зафиксировав последовательность проблем, мы можем проанализировать асимптотическое время работы как функции от переменных данной последовательности.

Прежде чем переходить к формальному определению, нам нужно уделить внимание еще одной тонкости. Если исходить из всего вышесказанного, то алгоритм обучения может «обманывать», передавая груз вычислений выходной гипотезе. Например, алгоритм может просто определить выходную гипотезу как функцию, которая сохраняет обучающий набор в своей памяти, а, получив тестовый пример x, вычисляет гипотезу ERM на обучающем наборе и применяет ее к x. Отметим, что в этом случае выход нашего алгоритма фиксирован (это только что описанная функция) и время его работы постоянно. Но обучение-то все равно трудное – только его сложность теперь заключается в реализации выходного классификатора, предсказывающего метку. Чтобы воспрепятствовать такому «обману», мы введем дополнительное требование: выход алгоритма обучения должен предсказывать метку нового примера за время, не превосходящее времени обучения (т. е. вычисления выходного классификатора по входному обучающему набору). В следующем подразделе подготовленный читатель найдет формальное определение вычислительной сложности обучения.

8.1.1. Формальное определение*

Приведенное ниже определение опирается на понятие базовой абстрактной машины, которая является либо машиной Тьюринга, либо машиной Тьюринга на множестве вещественных чисел. Мы будем измерять вычислительную сложность алгоритма количеством выполняемых им «операций», предполагая, что для любой машины, которая реализует базовую абстрактную машину, существует константа c такая, что любая такая «операция» может быть выполнена на этой машине за c секунд.

Определение 8.1 (вычислительная сложность алгоритма обучения)**.** Мы определим вычислительную сложность обучения в два этапа. Сначала рассмотрим вычислительную сложность фиксированной проблемы обучения (определяемой тройкой $(Z, \mathcal{H}, \ell)$ – область примеров, эталонный класс гипотез и функция потерь). Затем мы рассмотрим скорость изменения этой сложности на последовательности таких задач.

1. Пусть дана функция $f : (0, 1)^2 \to \mathbb{N}$, задача обучения $(Z, \mathcal{H}, \ell)$ и алгоритм обучения A. Мы говорим, что A решает задачу обучения за время $O(f)$, если

существует постоянное число c такое, что для любого распределения вероятностей $\mathcal{D}$ на Z и входных параметров $\epsilon, \delta \in (0, 1)$ справедливо следующее утверждение: если A имеет доступ к примерам, независимо выбранным из распределения $\mathcal{D}$, то:

- A завершается, выполнив не более $cf(\epsilon, \delta)$ операций;
- выход A, обозначаемый h_A, можно применить для предсказания метки нового примера, и при этом будет выполнено не более $cf(\epsilon, \delta)$ операций;
- выход A вероятно почти корректен, т. е. с вероятностью не ниже $1 - \delta$ (для случайной выборки, которую получает A) $L_{\mathcal{D}}(h_A) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon$.

2. Рассмотрим последовательность проблем обучения $(Z_n, \mathcal{H}_n, \ell_n)_{n=1}^{\infty}$, где проблема n описывается областью примеров Z_n, классом гипотез $\mathcal{H}_n$ функцией потерь ℓ_n. Пусть A – алгоритм обучения, спроектированный для решения проблем обучения такого вида. Если задана функция $g : \mathbb{N} \times (0, 1)^2 \to \mathbb{N}$, то мы говорим, что время работы A на вышеупомянутой последовательности равно $O(g)$, если для всех n A решает проблему $(Z_n, \mathcal{H}_n, \ell_n)$ за время $O(f_n)$, где функция $f_n : (0, 1)^2 \to \mathbb{N}$ определена выражением $f_n(\epsilon, \delta) = g(n, \epsilon, \delta)$.

Мы говорим, что алгоритм A *эффективен* на последовательности $(Z_n, \mathcal{H}_n, \ell_n)$, если его время работы равно $O(p(n, 1/\epsilon, 1/\delta))$ для некоторого полинома p.

Из этого определения видно, что вопрос о том, можно ли эффективно решить общую проблему обучения, зависит от того, как ее представить в виде последовательности конкретных проблем обучения. Например, рассмотрим проблему обучения конечного класса гипотез. В предыдущих главах было показано, что правило ERM на $\mathcal{H}$ гарантированно выполняет (ϵ, δ)-обучение $\mathcal{H}$, если количество обучающих примеров имеет порядок $m_{\mathcal{H}}(\epsilon, \delta) = \log(|\mathcal{H}|/\delta)/\epsilon^2$. В предположении, что вычисление гипотезы для одного примера занимает постоянное время, можно реализовать правило ERM за время $O(|\mathcal{H}| m_{\mathcal{H}}(\epsilon, \delta))$, произведя исчерпывающий поиск в $\mathcal{H}$ с размером обучающего набора $m_{\mathcal{H}}(\epsilon, \delta)$. Для любого фиксированного конечного $\mathcal{H}$ алгоритм исчерпывающего поиска работает за полиномиальное время. Более того, если мы определим последовательность проблем, для которой $|\mathcal{H}_n| = n$, то исчерпывающий поиск все равно будет эффективен. Но если в последовательности проблем $|\mathcal{H}_n| = 2^n$, то выборочная сложность останется полиномиально зависящей от n, а вычислительная сложность алгоритма исчерпывающего поиска растет экспоненциально (и, значит, алгоритм неэффективен).

8.2. Реализация правила ERM

Если задан класс гипотез $\mathcal{H}$, то правило $\text{ERM}_{\mathcal{H}}$, пожалуй, является самой естественной парадигмой обучения. Более того, для проблем бинарной классификации мы видели, что если обучение вообще возможно, то оно возможно и по правилу ERM. В этом разделе мы обсудим вычислительную сложность реализации правила ERM для нескольких классов гипотез.

Если дан класс гипотез $\mathcal{H}$, область примеров Z и функция потерь ℓ, то соответствующее правило $\text{ERM}_{\mathcal{H}}$ можно определить следующим образом:

На конечной входной выборке $S \in Z^m$ вывести гипотезу $h \in \mathcal{H}$, которая минимизирует эмпирический риск $L_S(h) = (1/|S|) \sum_{z \in S} \ell(h, z)$.

В этом разделе мы изучим время реализации правила ERM для нескольких примеров задач обучения.

8.2.1. Конечные классы

Конечность класса гипотез можно считать довольно мягким ограничением. Например, $\mathcal{H}$ может быть множеством всех предикторов, которые можно реализовать в виде программы на C++, содержащей не более 10 000 бит кода. Есть и другие полезные конечные классы, скажем, любой класс гипотез, который можно параметризовать конечным числом параметров, причем мы будем удовлетворены, если в представлении каждого параметра используется конечное число бит. Например, таковым является класс осепараллельных прямоугольников в евклидовом пространстве $\mathbb{R}^d$, когда параметры, описывающие каждый прямоугольник, задаются с ограниченной точностью.

В предыдущих главах мы показали, что выборочная сложность обучения конечного класса ограничена сверху величиной $m_{\mathcal{H}}(\epsilon, \delta) = c\log(c|\mathcal{H}|/\delta)/\epsilon^c$, где $c = 1$ в реализуемом случае и $c = 2$ в нереализуемом. Поэтому выборочная сложность умеренно зависит от размера $\mathcal{H}$. В примере с программами на C++ количество гипотез равно $2^{10\ 000}$, но выборочная сложность составляет только $c(10\ 000 + \log(c/\delta))/\epsilon^c$.

Прямолинейный подход к реализации правила ERM для конечного класса гипотез заключается в том, чтобы произвести исчерпывающий поиск. То есть для каждой гипотезы $h \in \mathcal{H}$ мы вычисляем эмпирический риск $L_S(h)$ и возвращаем ту гипотезу, которая минимизирует эмпирический риск. В предположении, что вычисление $\ell(h, z)$ для одного примера занимает постоянное время k, такой исчерпывающий поиск потребует времени $k|\mathcal{H}|m$, где m – размер обучающего набора. Если положить m равным вышеупомянутой верхней границе выборочной сложности, то время работы будет равно $k|\mathcal{H}|c\log(c|\mathcal{H}|/\delta)/\epsilon^c$.

Линейная зависимость времени работы от размера $\mathcal{H}$ делает такой подход неэффективным (и нереализуемым на практике) для больших классов. Формально, если мы определим последовательность проблем $(Z_n, \mathcal{H}_n, \ell_n)_{n=1}^{\infty}$ такую, что $\log(|\mathcal{H}_n|) = n$, то исчерпывающий поиск займет экспоненциальное время. В примере с программами на C++, если $\mathcal{H}_n$ – множество функций, которые можно реализовать на C++ программой не длиннее n бит, то время работы растет экспоненциально с ростом n, а, значит, исчерпывающий поиск практически нереализуем. На самом деле, эта проблема и есть одна из причин, по которым мы работает с другими классами гипотез, например, классами линейных предикторов (см. следующую главу), а не зацикливаемся только на конечных классах.

Важно понимать, что неэффективность одного алгоритмического подхода (например, исчерпывающего поиска) еще не означает, что не существует эффективной реализации ERM. Мы приведем примеры, доказывающие, что правило ERM можно реализовать эффективно.

8.2.2. Осепараллельные прямоугольники

Обозначим $\mathcal{H}_n$ класс осепараллельных прямоугольников в $\mathbb{R}^n$:

$$\mathcal{H}_n = \{h_{(a_1,\ldots,a_n,b_1,\ldots,b_n)} : \forall i,\ a_i \le b_i\},$$

где

$$h_{(a_1,\ldots,a_n,b_1,\ldots,b_n)}(\mathbf{x},y)=\begin{cases}1, & \text{если } \forall i, x_i\in[a_i,b_i]\\ 0, & \text{в противном случае}\end{cases}.$$

Допускает эффективное обучение в реализуемом случае

Рассмотрим реализацию правила ERM в реализуемом случае. То есть дан обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ такой, что существует осепараллельный прямоугольник $h \in \mathcal{H}_n$, для которого $h(\mathbf{x}_i) = y_i$ для всех i. Наша цель – найти осепараллельный прямоугольник с нулевой ошибкой обучения, т. е. совместимый со всеми метками в S.

Покажем, что это можно сделать за время $O(nm)$. Действительно, для любого $i \in [n]$ положим $a_i = \min\{x_i : (\mathbf{x}, 1) \in S\}$ и $b_i = \max\{x_i : (\mathbf{x}, 1) \in S\}$. Проще говоря, a_i – минимальное, а b_i – максимальное значение i-й координаты положительных примеров из S. Легко проверить, что у такого прямоугольника ошибка обучения равна 0, а время нахождения каждой пары a_i и b_i составляет $O(m)$. Поэтому общее время работы алгоритма равно $O(nm)$.

Не допускает эффективного обучения в агностическом случае

В агностическом случае мы не предполагаем, что какая-то гипотеза h идеально предсказывает метки всех примеров из обучающего набора. Поэтому наша цель – найти h, минимизирующую количество примеров, для которых $y_i \neq h(\mathbf{x}_i)$. Оказывается, что для многих распространенных классов гипотез, в т. ч. рассматриваемого здесь класса осепараллельных прямоугольников, задача ERM является NP-трудной (и в большинстве случаев NP-трудной является даже задача нахождения какой-нибудь гипотезы $h \in \mathcal{H}$, для которой ошибка не превосходит произведения некоторой константы $c > 1$ на минимум эмпирического риска в $\mathcal{H}$). Это означает, что если P ≠ NP, то не существует алгоритма, который гарантированно находит ERM-гипотезу за время, полиномиально зависящее от m и n (Ben-David, Eiron & Long, 2003).

С другой стороны, стоит отметить, что если зафиксировать один конкретный класс гипотез, скажем, осепараллельные прямоугольники в пространстве некоторой фиксированной размерности n, то для этого класса существуют эффективные алгоритмы обучения. Иными словами, существуют успешные агностические PAC-обучаемые, время работы которых полиномиально зависит от $1/\epsilon$ и $1/\delta$ (но при этом зависимость от размерности n не полиномиальная).

Чтобы убедиться в этом, вспомним приведенную выше реализацию правила ERM для реализуемого случая, из которой следует, что осепараллельный прямоугольник определен самое большее $2n$ примерами. Поэтому имея обучающий набор размера m, мы можем произвести исчерпывающий поиск по всем подмножествам этого набора размера не более $2n$ и для каждого такого подмножества построить прямоугольник. А затем выбрать прямоугольник с минимальной ошибкой обучения. Гарантируется, что эта процедура найдет ERM-гипотезу, а время ее работы составляет $m^{O(n)}$. Следовательно, если n фиксировано, то время работы полиномиально зависит от размера выборки. Это не противоречит ци-

тированному результату о трудности задачи, поскольку мы утверждали, что если P ≠ NP, то невозможно найти алгоритм, время работы которого зависело бы полиномиально также от размерности n.

8.2.3. Булевы конъюнкции

Булевой конъюнкцией называется отображение $\mathcal{X} = \{0, 1\}^n$ в $\mathcal{Y} = \{0, 1\}$, которое можно выразить формулой исчисления высказываний вида $x_{i_1} \wedge \ldots \wedge x_{i_k} \wedge \neg x_{j_1} \wedge \ldots \wedge \neg x_{j_r}$ для некоторых индексов $i_1, \ldots, i_k, j_1, \ldots, j_r \in [n]$. Эта формула определяет такую функцию:

$$h(\mathbf{x}) = \begin{cases} 1, & \text{если } x_{i_1} = \cdots = x_{i_k} = 1 \text{ и } x_{j_1} = \cdots = x_{j_r} = 0 \\ 0, & \text{в противном случае} \end{cases}.$$

Пусть $\mathcal{H}_C^n$ – класс всех булевых конъюнкций над множеством $\{0, 1\}^n$. Размер $\mathcal{H}_C^n$ не превышает $3^n + 1$ (поскольку в формулу конъюнкции каждый элемент $\mathbf{x}$ либо входит, либо входит со знаком отрицания, либо не входит вовсе, и еще надо добавить формулу, не содержащую ни одного литерала). Таким образом, выборочная сложность при обучении $\mathcal{H}_C^n$ по правилу ERM не превышает $d\log(3/\delta)/\epsilon$.

Допускает эффективное обучение в реализуемом случае

Далее мы покажем, что проблему ERM можно решить для класса $\mathcal{H}_C^n$ за время, полиномиально зависящее от n и m. Идея в том, чтобы определить ERM-конъюнкцию, включив в гипотезу все литералы, не противоречащие ни одному примеру с положительной меткой. Пусть $\mathbf{v}_1, \ldots, \mathbf{v}_{m^+}$ – все положительно помеченные примеры во входной выборке S. Индукцией по $i \le m^+$ определим последовательность гипотез (или конъюнкций). Пусть h_0 – конъюнкция, содержащая все возможные литералы, т. е. $h_0 = x_1 \wedge \ldots \neg x_1 \wedge x_2 \wedge \ldots \wedge x_n \wedge \neg x_n$. Отметим, что h_0 назначает метку 0 всем элементам $\mathcal{X}$. Чтобы получить h_{i+1}, удалим из конъюнкции h_i все литералы, которым не удовлетворяет $\mathbf{v}_{i+1}$. Алгоритм выводит гипотезу h_{m^+}. Отметим, что h_{m^+} назначает положительную метку всем положительно помеченным примерам в S. Кроме того, для любого $i \le m^+ h_i$ – наиболее ограничительная конъюнкция из тех, что положительно помечают примеры $\mathbf{v}_1, \ldots, \mathbf{v}_i$. А поскольку мы рассматриваем обучение, считая выполненным предположение о реализуемости, то существует конъюнкция (гипотеза) $f \in \mathcal{H}_C^n$, согласованная со всеми примерами в S. Поскольку h_{m^+} – самая ограничительная конъюнкция, назначающая положительные метки всем положительно помеченным элементам S, то любому примеру, которому f сопоставила метку 0, h_{m^+} также сопоставляет 0. Отсюда следует, что h_{m^+} имеет нулевую ошибку обучения (относительно S) и потому является допустимой ERM-гипотезой. Время работы этого алгоритма составляет $O(mn)$.

Не допускает эффективного обучения в агностическом случае

Как и в случае осепараллельных прямоугольников, если P ≠ NP, то не существует алгоритма, который гарантированно находил бы ERM-гипотезу для класса булевых конъюнкций без предположения о реализуемости за время, полиномиально зависящее от m и n.

8.2.4. Обучение трехчленных ДНФ

Далее мы покажем, что если немного обобщить класс булевых конъюнкций, то решить проблему ERM станет практически невозможно даже в реализуемом случае. Рассмотрим класс трехчленных дизъюнктивных нормальных форм (ДНФ). Пространство образцов $\mathcal{X} = \{0, 1\}^n$, а каждая гипотеза представлена булевой формулой вида $h(\mathbf{x}) = A_1(\mathbf{x}) \vee A_2(\mathbf{x}) \vee A_3(\mathbf{x})$, где $A_i(\mathbf{x})$ – булева конъюнкция (определенная в предыдущем разделе). Выходом $h(\mathbf{x})$ является 1, если хотя бы одна из конъюнкций $A_1(\mathbf{x}), A_2(\mathbf{x}), A_3(\mathbf{x})$ вводит метку 1. Если все три конъюнкции выводят метку 0, то $h(\mathbf{x}) = 0$.

Обозначим $\mathcal{H}^n_{3DNF}$ класс гипотез, состоящий из всех трехчленных ДНФ. Размер $\mathcal{H}^n_{3DNF}$ не превосходит 3^{3n}. Поэтому выборочная сложность обучения $\mathcal{H}^n_{3DNF}$ по правилу ERM не превосходит $3n\log(3/\delta)/\epsilon$.

Однако с вычислительной точки зрения проблема обучения является трудной. Показано (см. Pitt & Valiant, 1988; Kearns, Schapire & Sellie, 1994), что если RP ≠ NP, то не существует алгоритма с полиномиальным временем работы, который осуществлял бы *собственное* обучение последовательности проблем обучения трехчленных ДНФ, в которой размерность n-й проблемы равна n. Говоря «собственное», мы имеем в виду, что алгоритм должен вывести гипотезу, являющуюся трехчленной ДНФ. В частности, поскольку правило $\mathrm{ERM}_{\mathcal{H}^n_{3DNF}}$ выводит трехчленную ДНФ, оно является собственным обучаемым, и потому реализовать соответствующий алгоритм трудно. В доказательстве используется сведение задачи о 3-раскраске графов к задаче PAC-обучения трехчленных ДНФ. Детали см. в упражнении 8.4. См. также Kearns and Vazirani, 1994, раздел 1.4.

8.3. Эффективно обучаемый, но не собственный алгоритм ERM

В предыдущем разделе мы видели, что невозможно эффективно реализовать правило ERM для класса $\mathcal{H}^n_{3DNF}$ трехчленных ДНФ. В этом разделе мы покажем, что этот класс можно обучить эффективно, если использовать ERM относительно большего класса.

Обучение, независимое от представления, не является трудной задачей

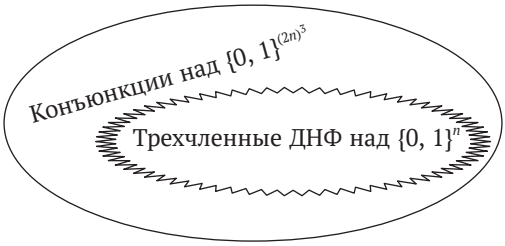
Далее мы покажем, что трехчленные ДНФ можно обучить эффективно. Это не противоречит вышеупомянутому результату о трудности, потому что теперь мы допускаем «независимое от представления» обучение. Это означает, что алгоритм обучения не обязан выводить гипотезу, являющуюся трехчленной ДНФ. Идея заключается в том, чтобы заменить исходный класс гипотез более широким, который допускает простое обучение. Алгоритм обучения может возвращать гипотезу, не принадлежащую исходному классу, отсюда и название «независимое от представления». Подчеркнем, что в большинстве случаев нас на самом деле интересует гипотеза с хорошей предсказательной способностью.

Для начала отметим, что, поскольку операция $\vee$ дистрибутивна относительно $\wedge$, любую трехчленную ДНФ можно переписать в виде

$$A_1 \vee A_2 \vee A_3 = \bigwedge_{u \in A_1,\, v \in A_2,\, w \in A_3} (u \vee v \vee w).$$

Далее определим отображение ψ: $\{0, 1\}^n \to \{0, 1\}^{(2n)^3}$ такое, что для любой тройки литералов u, v, w существует переменная в области значений ψ, показывающая, принимает ли выражение $u \vee v \vee w$ значение true или false. Таким образом, для любой трехчленной ДНФ над множеством $\{0, 1\}^n$ существует конъюнкция над множеством $\{0, 1\}^{(2n)^3}$ с точно такой же таблицей истинности. Поскольку мы предполагаем, что данные реализуемы, то можем решить проблему ERM относительно класса конъюнкций над $\{0, 1\}^{(2n)^3}$. При этом выборочная сложность обучения класса конъюнкций в этом пространстве более высокой размерности не превышает $n^3 \log(1/\delta)/\epsilon$. Следовательно, время обучения этого алгоритма полиномиально зависит от n.

На интуитивном уровне мы проделали вот что. Мы начали с класса гипотез, задача обучения которого трудна. Затем мы перешли к другому представлению, в котором класс гипотез больше исходного, но и лучше структурирован, что позволяет более эффективно производить поиск ERM. В новом представлении решить проблему ERM оказывается легко.



8.4. Трудность обучения*

Мы только что продемонстрировали, что вычислительная трудность реализации $\text{ERM}_{\mathcal{H}}$ не означает, что класс $\mathcal{H}$ не допускает обучения. Как доказать, что проблема обучения вычислительно трудная?

Один из подходов опирается на криптографические предположения. В некотором смысле криптография – противоположность обучению. Цель обучения – вскрыть некоторое правило, которому подчиняются наблюдаемые примеры, тогда как цель криптографии – сделать так, чтобы никто не мог раскрыть некоторый секрет, несмотря на наличие частичной информации о нем. При таком интуитивном понимании результаты, касающиеся криптографической безопасности системы, транслируются в результаты о необучаемости соответствующей задачи. К сожалению, в настоящее время не существует способа доказать невскрываемость криптографического протокола. Для этого недостаточно даже стандартного предположения $\text{P} \neq \text{NP}$ (хотя можно показать, что для большинства распространенных криптографических сценариев оно необходимо). Типичный

подход к доказательству безопасности криптографического протокола – начать с некоторых *криптографических предположений*. Чем чаще они используются в качестве основы криптографии, тем сильнее наша вера в их справедливость (или, по крайней мере, в то, что алгоритмы, опровергающие эти предположения, найти трудно).

Мы кратко опишем основную идею, стоящую за выводом трудности обучения из криптографических предположений. Многие криптографические системы базируются на предположении о существовании односторонней функции. Грубо говоря, односторонней называется функция $f : \{0, 1\}^n \to \{0, 1\}^n$ (точнее, последовательность функций для каждой размерности n), которую легко вычислить, но трудно обратить. Формально, f можно вычислить за время $\mathrm{poly}(n)$, но для любого рандомизированного алгоритма с полиномиальным временем работы A и для любого полинома $p(\cdot)$

$$\mathbb{P}[f(A(f(\mathbf{x}))) - f(\mathbf{x})] < \frac{1}{p(n)},$$

где вероятность вычисляется для случайной величины $\mathbf{x}$ с равномерным распределением на $\{0, 1\}^n$ и случайного A.

Односторонняя функция f называется функцией с лазейкой, если для некоторого полинома p и для любого n существует битовая строка s_n (называемая секретным ключом) длины $\le p(n)$ такая, что существует алгоритм с полиномиальным временем работы, обладающий следующим свойством: для любого n и любого $\mathbf{x} \in \{0, 1\}^n$ по входу $(f(\mathbf{x}), s_n)$ вырабатывается выход $\mathbf{x}$. Иными словами, хотя обратить f трудно, но при наличие доступа к секретному ключу это становится практически осуществимым. Такие функции параметризуются своим секретным ключом.

Пусть теперь F_n – семейство функций с лазейкой над множеством $\{0, 1\}^n$, которые можно вычислить с помощью некоторого алгоритма с полиномиальным временем работы. То есть мы фиксируем алгоритм, который, получив секретный ключ (представляющий одну функцию из F_n) и входной вектор, вычисляет значение функции, соответствующей секретному ключу, на входном векторе за полиномиальное время. Рассмотрим задачу обучения класса соответствующих обратных функций $\mathcal{H}_F^n = \{f^{-1} : f \in F_n\}$. Поскольку каждую функцию из этого класса можно обратить с помощью некоторого секретного ключа s_n размера, полиномиально зависящего от n, то класс $\mathcal{H}_F^n$ можно параметризовать этими ключами, и его размер не превышает $2^{p(n)}$. Поэтому его выборочная сложность полиномиально зависит от n. Мы утверждаем, что для этого класса не существует эффективного обучаемого. Если бы такой обучаемый L существовал, то, случайным образом выбрав полиномиальное количество строк в $\{0, 1\}^n$ с равномерным распределением и вычислив для них f, мы смогли бы сгенерировать помеченную обучающую выборку пар $(f(\mathbf{x}), \mathbf{x})$, которой было бы достаточно, чтобы наш обучаемый мог найти (ϵ, δ)-аппроксимацию f^{-1} (относительно равномерного распределения на множестве значений f), что явилось бы нарушением свойства односторонности f.

Более детальное рассмотрение, а заодно конкретный пример можно найти в книге Kearns and Vazirani (1994, гл. 6). Применяя сведение, они также показали,

что класс функций, вычислимых с помощью небольших булевых схем, не является эффективно обучаемым, даже в реализуемом случае.

8.5. Резюме

Асимптотическое время работы алгоритмов обучения анализируется как функция от различных параметров проблемы обучения, например, размера класса гипотез, меры верности и уверенности или размера области определения. Мы продемонстрировали, что в некоторых случаях правило ERM можно реализовать эффективно. Например, мы вывели эффективные алгоритмы решения проблемы ERM для класса булевых конъюнкций и для класса осепараллельных прямоугольников в предположении о реализуемости. Однако задача реализации ERM для тех же классов в агностическом случае является NP-трудной. Напомним, что со статистической точки зрения между реализуемым и агностическим случаем нет разницы (т. е. в обоих случаях класс допускает обучение тогда и только тогда, когда его VC-размерность конечна). Но, как мы убедились, с вычислительной точки зрения разница огромна. Мы также привели другой пример, класс трехчленных ДНФ, для которого реализация ERM трудна даже в предположении о реализуемости, хотя этот класс можно эффективно обучить другим алгоритмом.

Трудность реализации правила ERM для нескольких естественных классов гипотез стала стимулом для разработки альтернативных методов обучения, которые мы будем обсуждать в следующей части книги.

8.6. Библиографические сведения

В работе Valiant (1984) была введена модель эффективного PAC-обучения, в которой требовалось, чтобы время работы алгоритма полиномиально зависело от $1/\epsilon$, $1/\delta$ и размера представления гипотез, принадлежащих классу. Обсуждение деталей и подробные библиографические ссылки приведены в книге Kearns and Vazirani (1994).

8.7. Упражнения

8.1. Пусть $\mathcal{H}$ – класс отрезков на прямой (формально эквивалентный классу осепараллельных прямоугольников для размерности $n = 1$). Предложите реализацию правила $\text{ERM}_{\mathcal{H}}$ (в агностическом случае), которая работает за время $O(m^2)$ на обучающем наборе размера m.

Указание. Воспользуйтесь динамическим программированием.

8.2. Пусть $\mathcal{H}_1, \mathcal{H}_2, \ldots$ – последовательность классов гипотез для бинарной классификации. Предположим, что существует алгоритм обучения, который реализует правило ERM в реализуемом случае, так что выходная гипотеза для каждого класса $\mathcal{H}_n$ зависит только от $O(n)$ примеров из обучающего набора. Предположим далее, что такую гипотезу можно вычислить по этим $O(n)$ примерам за время

$O(n)$ и что эмпирический риск каждой такой гипотезы можно вычислить за время $O(mn)$. Например, если $\mathcal{H}_n$ – класс осепараллельных прямоугольников в $\mathbb{R}^n$, то, как мы видели, в реализуемом случае можно найти ERM-гипотезу, определяемую не более чем $2n$ примерами. Докажите, что тогда можно найти ERM-гипотезу для $\mathcal{H}_n$ в нереализуемом случае за время $O(mn\,m^{O(n)})$.

8.3. В этом упражнении мы представим несколько классов, для которых задача нахождения ERM-классификатора вычислительно трудна. Сначала рассмотрим класс n-мерных полупространств HS_n для области образцов $\mathcal{X} = \mathbb{R}^n$. Это класс всех функций вида $h_{\mathbf{w},b}(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \mathbf{x}\rangle + b)$, где векторы $\mathbf{w}, \mathbf{x} \in \mathbb{R}^n$, $\langle \mathbf{w}, \mathbf{x}\rangle$ обозначает их скалярное произведение, а $b \in \mathbb{R}$. Подробности см. в главе 9.

1. Покажите, что реализовать правило $\text{ERM}_{\mathcal{H}}$ над классом $\mathcal{H} = HS_n$ линейных предикторов вычислительно трудно. Точнее, рассмотрим последовательность проблем, в которой размерность n растет линейно, а количество примеров m – некоторая константа, умноженная на n.
   Указание. Для доказательства трудности можно свести эту задачу к следующей.

   > *Max FS.* Дана система линейных неравенств $A\mathbf{x} > \mathbf{b}$, где $A \in \mathbb{R}^{m\times n}$, $\mathbf{b} \in \mathbb{R}^m$ (т. е. система m линейных неравенств от n переменных, $\mathbf{x} = (x_1, \ldots, x_n)$). Требуется найти максимальную подсистему неравенств, имеющую решение (такая подсистема называется *допустимой*).

   Было доказано (Sankaran, 1993), что задача Max FS является NP-трудной. Покажите, что любой алгоритм, который находит гипотезу ERM_{HS_n} для любой обучающей выборки $S \in (\mathbb{R}^n \times \{+1, -1\})^m$, можно использовать для решения проблемы Max FS размера m, n.
   Указание. Определите отображение линейного неравенства от n переменных в помеченную точку в $\mathbb{R}^n$ и отображение векторов $\mathbb{R}^n$ в полупространства такие, что вектор $\mathbf{w}$ удовлетворяет неравенству q тогда и только тогда, когда помеченная точка, соответствующая q, правильно классифицируется полупространством, соответствующим $\mathbf{w}$. Сделайте вывод, что проблема минимизации эмпирического риска для полупространств также является NP-трудной (т. е. если она может быть решена за время, полиномиально зависящее от размера выборки m и размерности евклидова пространства n, то любая задача класса NP может быть решена за полиномиальное время).
2. Пусть $\mathcal{X} = \mathbb{R}^n$, и обозначим $\mathcal{H}_k^n$ класс всех пересечений k линейных подпространств в $\mathbb{R}^n$. В этом упражнении мы покажем, что реализация правила $\text{ERM}_{\mathcal{H}_k^n}$ вычислительно трудна для всех $k \geq 3$. Точнее, рассмотрим последовательность проблем, для которой $k \geq 3$ – константа, а n растет линейно. Размер обучающего набора m также растет линейно вместе с n. Теперь рассмотрим задачу о k-раскраске графа, которая формулируется следующим образом:

   > Дан граф $G = (V, E)$ и число k. Определить, существует ли функция $f : V \to \{1\ldots k\}$ такая, что для любых $(u, v) \in E$, $f(u) \neq f(v)$.

   Известно, что задача о k-раскраске является NP-трудной для всех $k \geq 3$ (Karp, 1972). Мы хотим свести задачу о k-раскраске к $\text{ERM}_{\mathcal{H}_k^n}$, т. е. доказать, что если существует алгоритм, решающий проблему $\text{ERM}_{\mathcal{H}_k^n}$ за время, по-

линомиально зависящее от k, n и размера выборки m, то существует полиномиальный алгоритм и для задачи о k-раскраске.

Пусть дан граф $G = (V, E)$, и $\{v_1 \ldots v_n\}$ – множество его вершин V. Построим выборку $S(G) \in (\mathbb{R}^n \times \{\pm 1\})^m$, где $m = |V| + |E|$, следующим образом:

- для любой вершины $vi \in V$ включим пример $\mathbf{e}_i$ с отрицательной меткой;
- для любого ребра $(v_i, v_j) \in E$ включим пример $(\mathbf{e}_i + \mathbf{e}_j)/2$ с положительной меткой.

1. Докажите, что если существует гипотеза $h \in \mathcal{H}_k^n$, имеющая нулевую ошибку на выборке $S(G)$, то G допускает k-раскраску.
   Указание. Пусть $h = \bigcap_{j=1}^{k} h_j$ – ERM-классификатор из класса $\mathcal{H}_k^n$ над S. Определим раскраску V, положив $f(v_i)$ равным минимальному числу j такому, что $h_j(\mathbf{e}_i) = -1$. Воспользовавшись тем фактом, что полупространства – выпуклые множества, покажите, что две вершины, соединенные ребром, не могут быть окрашены в одинаковый цвет.
2. Докажите, что если G допускает k-раскраску, то существует $h \in \mathcal{H}_k^n$ с нулевой ошибкой на $S(G)$.
   Указание. По данной раскраске f вершин G мы должны найти k гиперплоскостей $h_1, \ldots, h_k$, пересечение которых является идеальным классификатором для $S(G)$. Положим $b = 0{,}6$ для всех этих гиперплоскостей и для $t \le k$ пусть i-й вес t-й гиперплоскости, $w_{t,i}$, будет равен -1, если $f(v_i) = t$, и 0 в противном случае.
3. С учетом всего вышесказанного докажите, что для любого $k \ge 3$ задача $\mathrm{ERM}_{\mathcal{H}_k^n}$ является NP-трудной.

8.4. В этом упражнении мы покажем, что трудность проблемы ERM эквивалентна трудности собственного PAC-обучения. Напомним, что алгоритм называется «собственным», если он порождает гипотезу, принадлежащую классу гипотез. Для формализации этого утверждения нам понадобится следующее определение.

Определение 8.2. Классом вычислительной сложности RP (Randomized Polynomial – рандомизированные полиномы) называется класс всех проблем разрешимости (т. е. проблем, в которых для любого примера нужно выяснить, является ли ответом ДА или НЕТ), для которых существует вероятностный алгоритм (т. е. в процессе работы алгоритма разрешается подбрасывать монету), обладающий следующими свойствами:

- для любого входного примера время работы алгоритма полиномиально зависит от размера входа;
- если правильный ответ – НЕТ, то алгоритм должен вернуть НЕТ;
- если правильный ответ – ДА, то алгоритм возвращает ДА с вероятностью a $\ge 1/2$ и НЕТ с вероятностью $1 - a$[1].

Очевидно, что класс RP содержит класс P. Известно также, что RP содержится в классе NP. Неизвестно, совпадает ли RP с P или с NP, но многие полагают, что NP строго больше RP. В частности, считается, что NP-трудные задачи невоз-

[1] Вместо константы 1/2 в этом определении можно использовать любое число из интервала (0, 1).

можно решить с помощью рандомизированного алгоритма с полиномиальным временем работы.

- Покажите, что если класс $\mathcal{H}$ допускает *собственное* PAC-обучение алгоритмом с полиномиальным временем работы, то проблема $\text{ERM}_{\mathcal{H}}$ принадлежит классу сложности RP. В частности, отсюда следует, что если проблема $\text{ERM}_{\mathcal{H}}$ является NP-трудной (например, в случае класса пересечений полупространств из предыдущего упражнения), то в предположении, что NP ≠ RP, не существует собственного алгоритма PAC-обучения для $\mathcal{H}$ с полиномиальным временем работы.

  Указание. Предположим, что существует алгоритм A, который реализует собственное PAC-обучение класса $\mathcal{H}$ за время, полиномиально зависящее от некоторого параметра класса n, а также от $1/\epsilon$ и $1/\delta$. Ваша цель – воспользоваться этим алгоритмом как подпрограммой рандомизированного алгоритма B, решающего проблему $\text{ERM}_{\mathcal{H}}$ за полиномиальное время. Имея обучающий набор $S \in (\mathcal{X} \times \{\pm 1\}^m)$ и некоторую гипотезу $h \in \mathcal{H}$ с нулевой ошибкой на S, примените этот алгоритм PAC-обучения к равномерному распределению на S и выполните его таким образом, чтобы с вероятностью ≥ 0,3 он находил функцию $h \in \mathcal{H}$ с ошибкой, меньшей $\epsilon = 1/|S|$ (относительно этого равномерного распределения). Покажите, что описанный алгоритм действительно является RP-решателем для $\text{ERM}_{\mathcal{H}}$.

Часть *II*

ОТ ТЕОРИИ К АЛГОРИТМАМ

ЛАНЬ®

Глава 9

ЛИНЕЙНЫЕ ПРЕДИКТОРЫ

В этой главе мы будем изучать семейство линейных предикторов, одно из самых полезных семейств классов гипотез. Многие широко используемые на практике алгоритмы опираются на линейные предикторы прежде всего из-за способности эффективно обучаться во многих случаях. Кроме того, линейные предикторы интуитивно понятны, их результаты легко интерпретировать, и они хорошо аппроксимируют данные во многих естественно возникающих задачах обучения.

Мы введем несколько классов гипотез, принадлежащих этому семейству: полупространства, линейно-регрессионные предикторы и логистически-регрессионные предикторы, а также представим соответствующие алгоритмы обучения: линейное программирование и алгоритм перцептрона для класса полупространств и алгоритм наименьших квадратов для линейной регрессии. В этой главе нас будет интересовать обучение линейных предикторов на основе подхода ERM, но в последующих главах мы рассмотрим и другие парадигмы обучения этих классов гипотез.

Для начала определим класс аффинных функций

$$L_d = \{h_{\mathbf{w},b} : \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}\},$$

где

$$h_{\mathbf{w},b}(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = \left(\sum_{i=1}^{d} w_i x_i \right) + b.$$

Будет удобно использовать также нотацию

$$L_d = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle + b : \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}\},$$

которая читается так: L_d – множество функций, в котором каждая функция параметризована вектором $\mathbf{w} \in \mathbb{R}^d$ и скаляром $b \in \mathbb{R}$, принимает на входе вектор $\mathbf{x}$ и возвращает скаляр $\langle \mathbf{w}, \mathbf{x} \rangle + b$.

Различные классы гипотез, являющихся линейными предикторами, представлены композициями с функцией $\varphi : \mathbb{R} \to \mathcal{Y}$ на L_d. Например, для бинарной классификации можно взять в качестве φ функцию sign, а для проблем регрессии, в которых $\mathcal{Y} = \mathbb{R}$, φ – просто тождественная функция.

Иногда удобнее включать параметр b, называемый *смещением*, в состав $\mathbf{w}$ как дополнительную координату и добавить еще одну координату, всегда равную 1,

во все векторы $\mathbf{x} \in \mathcal{X}$; точнее, положим $\mathbf{w}' = (b, w_1, w_2, \ldots w_d) \in \mathbb{R}^{d+1}$, $\mathbf{x}' = (1, x_1, x_2, \ldots, x_d) \in \mathbb{R}^{d+1}$. Тогда

$$h_{\mathbf{w},b}(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = \langle \mathbf{w}', \mathbf{x}' \rangle.$$

Отсюда следует, что любую аффинную функцию в $\mathbb{R}^d$ можно переписать в виде однородной линейной функции в $\mathbb{R}^{d+1}$, применяемой к результату преобразования, которое дописывает постоянную координату 1 к каждому входному вектору. Поэтому в тех случаях, когда это упрощает изложение, мы будем опускать смещение и рассматривать L_d как класс однородных линейных функций вида $h_{\mathbf{w}}(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$.

В этой книге мы часто используем общий термин «линейные функции» для обозначения как аффинных, так и (однородных) линейных функций.

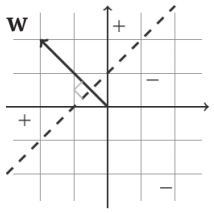
9.1. Полупространства

Первым мы рассмотрим класс полупространств, предназначенный для решения проблем бинарной классификации, когда $\mathcal{X} = \mathbb{R}^d$ и $\mathcal{Y} = \{-1, +1\}$. Класс полупространств определяется следующим образом:

$$HS_d = \text{sign} \circ L_d = \{\mathbf{x} \mapsto \text{sign}(h_{\mathbf{w},b}(\mathbf{x})) : h_{\mathbf{w},b} \in L_d\}.$$

Иными словами, каждое полупространство в HS_d параметризуется вектором $\mathbf{w} \in \mathbb{R}^d$ и скаляром $b \in \mathbb{R}$, и, получив вектор $\mathbf{x}$, гипотеза возвращает в качестве метки $\text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$.

Для геометрической иллюстрации этого класса гипотез поучительно рассмотреть случай $d = 2$. Каждой гипотезе соответствует полуплоскость, перпендикулярная вектору $\mathbf{w}$ и пересекающая вертикальную ось в точке $(0, -b/w_2)$. Примеры, лежащие «выше» этой гиперплоскости, т. е. образующие острый угол с $\mathbf{w}$, помечаются положительной меткой, а лежащие «ниже» нее, т. е. образующие тупой угол с $\mathbf{w}$, – отрицательной.



В разделе 9.1.3 мы покажем, что $\text{VCdim}(HS_d) = d + 1$. Отсюда следует, что класс полупространств можно обучить с помощью парадигмы ERM при условии, что размер выборки составляет $\Omega((d + \log(1/\delta))/\epsilon)$. Поэтому мы сейчас обсудим, как реализовать процедуру ERM для полупространств.

Далее мы представим два способа найти ERM-полупространство в реализуемом случае. В контексте полупространств реализуемый случай часто называют «разделимым», поскольку имеется возможность разделить гиперплоскостью положительные и отрицательные примеры. Известно, что реализация правила

ERM в неразделимом (т. е. агностическом) случае – вычислительно трудная задача (Ben-David and Simon, 2001). Существует несколько подходов к обучению на неразделимых данных. Самый популярный – использование *суррогатной функции потерь*, т. е. нахождение полупространства, которое минимизирует эмпирический риск, но относительно не бинарной, а какой-то другой функции потерь. Например, в разделе 9.3 мы опишем подход на основе логистической регрессии, который можно эффективно реализовать даже в неразделимом случае. Более подробно суррогатные функции потерь изучаются в главе 12.

9.1.1. Линейное программирование для класса полупространств

Линейной программой (ЛП) называется проблема, которую можно сформулировать как задачу максимизации линейной функции при соблюдении линейных неравенств:

$$\begin{array}{ll} \min\limits_{\mathbf{w}\in\mathbb{R}^d} & \langle \mathbf{u}, \mathbf{w}\rangle \\ \text{при условии} & A\mathbf{w} \geq \mathbf{v} \end{array},$$

где $\mathbf{w} \in \mathbb{R}^d$ – искомый вектор переменных, A – матрица $m \times d$, а $\mathbf{v} \in \mathbb{R}^m$, $\mathbf{u} \in \mathbb{R}^d$ – векторы. Линейные программы допускают эффективное решение[1], и даже имеются открытые реализации ЛП-решателей.

Мы покажем, что проблему ERM для полупространств в реализуемом случае можно выразить как линейную программу. Для простоты будем рассматривать однородный случай. Пусть $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ – обучающий набор размера m. Поскольку мы считаем, что выполняется предположение о реализуемости, то ERM-предиктор должен давать нулевые ошибки на примерах из обучающего набора. То есть мы ищем вектор $\mathbf{w} \in \mathbb{R}^d$, для которого

$$\text{sign}(\langle \mathbf{w}, \mathbf{x}_i\rangle) = y_i, \qquad \forall i = 1, \ldots, m.$$

Эквивалентно можно сказать, что мы ищем вектор $\mathbf{w}$, для которого

$$y_i\langle \mathbf{w}, \mathbf{x}_i\rangle > 0, \qquad \forall i = 1, \ldots, m.$$

Обозначим $\mathbf{w}^*$ вектор, удовлетворяющий этому условию (он должен существовать в силу предположения о реализуемости). Определим $\gamma = \min_i(y_i\langle \mathbf{w}^*, \mathbf{x}_i\rangle)$ и положим $\bar{\mathbf{w}} = \mathbf{w}^*/\gamma$. Тогда для любого i имеем

$$y_i\langle \bar{\mathbf{w}}, x_i\rangle = \frac{1}{\gamma} y_i\langle \mathbf{w}^*, \mathbf{x}_i\rangle \geq 1.$$

Таким образом, мы показали, что существует вектор, удовлетворяющий условию

$$y_i\langle \mathbf{w}, \mathbf{x}_i\rangle \geq 1, \qquad \forall i = 1, \ldots, m. \tag{9.1}$$

Очевидно, что такой вектор является ERM-предиктором.

---

[1] Точнее, за время, полимиально зависящее от m, d и размера представления вещественного числа.

Чтобы найти вектор, удовлетворяющий условию (9.1), мы можем следующим образом воспользоваться ЛП-решателем. Пусть A – матрица $m \times d$, строками которой являются примеры, умноженные на y_i, т. е. $A_{i,j} = y_i x_{i,j}$, где $x_{i,j}$ – j-й элемент вектора $\mathbf{x}_i$. Обозначим $\mathbf{v}$ вектор $(1, \ldots, 1) \in \mathbb{R}^m$. Тогда условие (9.1) можно переписать в виде

$$A\mathbf{w} \geq \mathbf{v}.$$

В форме ЛП требуется целевая функция, подлежащая максимизации, однако же все $\mathbf{w}$, удовлетворяющие этим ограничениям, в равной мере претендуют на роль выходной гипотезы. Поэтому мы зададим фиктивную целевую функцию $\mathbf{u} = (0, \ldots, 0) \in \mathbb{R}^d$.

9.1.2. Перцептрон для полупространств

Другую реализацию правила ERM дает алгоритм перцептрона Розенблатта (Rosenblatt, 1958). Это итеративный алгоритм, который строит последовательность векторов $\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \ldots$. В начальный момент $\mathbf{w}^{(1)}$ – вектор, содержащий одни нули. На t-й итерации перцептрон находит пример i, неправильно помеченный вектором $\mathbf{w}^{(t)}$, т. е. такой, для которого $\text{sign}(\langle \mathbf{w}^{(t)}, \mathbf{x}_i \rangle) \neq y_i$. Затем перцептрон обновляет $\mathbf{w}^{(t)}$, прибавляя к нему экземпляр $\mathbf{x}_i$, умноженный на метку y_i: $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + y_i \mathbf{x}_i$. Напомним, что наша цель – добиться выполнения условий $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle > 0$ для всех i, и заметим, что

$$y_i \langle \mathbf{w}^{(t+1)}, \mathbf{x}_i \rangle = y_i \langle \mathbf{w}^{(t)} + y_i \mathbf{x}_i, \mathbf{x}_i \rangle = y_i \langle \mathbf{w}^{(t)}, \mathbf{x}_i \rangle + \|\mathbf{x}_i\|^2.$$

Таким образом, после обновления, произведенного перцептроном, решение становится «более правильным» на i-м примере.

Пакетный перцептрон

вход: обучающий набор $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
инициализация: $\mathbf{w}^{(1)} = (0, \ldots, 0)$
  for $t = 1, 2, \ldots$
    if ($\exists i$ такое, что $y_i \langle \mathbf{w}^{(t)}, \mathbf{x}_i \rangle \leq 0$) **then**
      $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + y_i \mathbf{x}_i$
    else
      вывести $\mathbf{w}^{(t)}$

Следующая теорема гарантирует, что в реализуемом случае алгоритм останавливается, и в этот момент все примеры классифицированы правильно.

Теорема 9.1. *Предположим, что примеры* $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ *разделимы, обозначим* $B = \min\{\|\mathbf{w}\| : \forall i \in [m], y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1\}$ *и положим* $R = \max_i \|\mathbf{x}_i\|$. *Тогда перцептрон останавливается, совершив не более* $(RB)^2$ *итераций, и в момент остановки выполняются условия* $\forall i \in [m] y_i \langle \mathbf{w}^{(t)}, \mathbf{x}_i \rangle > 0$.

Доказательство. По определению условия остановки, если перцептрон остановился, то все примеры должны быть разделены. Мы покажем, что если перцеп-

трон проработал T итераций, то должно выполняться неравенство $T \le (RB)^2$, откуда следует, что перцептрон должен остановиться после не более $(RB)^2$ итераций.

Обозначим $\mathbf{w}^*$ вектор, на котором достигается минимум из определения B. Это значит, что $yi\langle \mathbf{w}^*, x_i\rangle \ge 1$ для всех i и среди векторов, удовлетворяющих этим ограничениям, $\mathbf{w}^*$ имеет минимальную норму.

Идея доказательства состоит в том, чтобы показать, что после выполнения T итераций косинус угла между $\mathbf{w}^*$ и $\mathbf{w}^{(T+1)}$ не меньше $\sqrt{T}/\mathrm{RB}$. Иначе говоря, мы покажем, что

$$\frac{\langle \mathbf{w}^*, \mathbf{w}^{(T+1)}\rangle}{\|\mathbf{w}^*\|\|\mathbf{w}^{(T+1)}\|} \ge \frac{\sqrt{T}}{RB}. \tag{9.2}$$

В силу неравенства Коши–Буняковского левая часть (9.2) не превосходит 1. Поэтому из (9.2) следует, что

$$1 \ge \frac{\sqrt{T}}{RB} \quad \Rightarrow \quad T \le (RB)^2,$$

что и требовалось доказать.

Чтобы доказать справедливость неравенства (9.2), мы сначала покажем, что $\langle \mathbf{w}, \mathbf{w}^{(T+1)}\rangle \ge T$. Действительно, на первой итерации $\mathbf{w}^{(1)} = (0, \ldots, 0)$, так что $\langle \mathbf{w}, \mathbf{w}^{(1)}\rangle = 0$, тогда как на t-й итерации, если обновление проведено для примера $(\mathbf{x}_i, y_i)$, то

$$\begin{aligned}\langle \mathbf{w}^*, \mathbf{w}^{(t+1)}\rangle - \langle \mathbf{w}^*, \mathbf{w}^{(t)}\rangle &= \langle \mathbf{w}^*, \mathbf{w}^{(t+1)} - \mathbf{w}^{(t)}\rangle \\ &= \langle \mathbf{w}^*, y_i\mathbf{w}^{(t+1)}\rangle = y_i\langle \mathbf{w}^*, \mathbf{x}_i\rangle \\ &\ge 1.\end{aligned}$$

Следовательно, после T итераций мы имеем

$$\langle \mathbf{w}^*, \mathbf{w}^{(T+1)}\rangle = \sum_{t=1}^{T}(\langle \mathbf{w}^*, \mathbf{w}^{(t+1)}\rangle - \langle \mathbf{w}^*, \mathbf{w}^{(t)}\rangle) \ge T, \tag{9.3}$$

что и требуется.

Далее оценим $\|\mathbf{w}^{(T+1)}\|$ сверху. Для t-й итерации имеем

$$\begin{aligned}\|\mathbf{w}^{(t+1)}\|^2 &= \|\mathbf{w}^{(t+1)} + y_i\mathbf{x}_i\|^2 \\ &= \|\mathbf{w}^{(t)}\|^2 + 2y_i\langle \mathbf{w}^{(t)}, \mathbf{x}_i\rangle + y_i^2\|\mathbf{x}_i\|^2 \\ &\le \|\mathbf{w}^{(t)}\|^2 + R^2,\end{aligned} \tag{9.4}$$

причем последнее неравенство имеет место, потому что пример i обязательно выбран так, что $y_i\langle \mathbf{w}^{(t)}, \mathbf{x}_i\rangle \le 0$, а норма $\mathbf{x}_i$ не превосходит R. Теперь, поскольку $\|\mathbf{w}^{(1)}\|^2 = 0$, если рекурсивно воспользоваться неравенством (9.4) для T итераций, то получится, что

$$\|\mathbf{w}^{(T+1)}\|^2 \le TR^2 \quad \Rightarrow \quad \|\mathbf{w}^{(T+1)}\|^2 \le \sqrt{T}R. \tag{9.5}$$

Объединяя (9.3) с (9.5) и воспользовавшись тем фактом, что $\|\mathbf{w}^*\| = B$, получаем

$$\frac{\langle \mathbf{w}^{(T+1)}, \mathbf{w}^*\rangle}{\|\mathbf{w}^*\|\|\mathbf{w}^{(T+1)}\|} \ge \frac{T}{B\sqrt{T}R} = \frac{\sqrt{T}}{BR}.$$

Таким образом, мы показали что неравенство (9.2) справедливо, и на этом доказательство завершается. □

Замечание 9.1. Алгоритм перцептрона легко реализуется и гарантированно сходится. Однако скорость сходимости зависит от параметра B, который в некоторых ситуациях может экспоненциально зависеть от d. В таких случаях проблему ERM лучше реализовать, решая линейную программу, как описано в предыдущем разделе. Тем не менее для многих естественно возникающих наборов данных величина B не слишком велика и перцептрон сходится очень быстро.

9.1.3. VC-размерность класса полупространств

ЛАНЬ®

Для вычисления VC-размерности класса полупространств начнем с однородного случая.

Теорема 9.2. *VC-размерность класса однородных полупространств в* $\mathbb{R}^d$ *равна* d.

Доказательство. Сначала рассмотрим множество векторов $\mathbf{e}_1, \ldots, \mathbf{e}_d$ таких, что i-я координата вектора $\mathbf{e}_i$ равна 1, а остальные равны 0. Это множество разбивается классом однородных полупространств. В самом деле, для любой пометки $y_1,\ldots, y_d$ положим $\mathbf{w} = (y_1, \ldots, y_d)$, тогда $\langle \mathbf{w}, \mathbf{e}_i \rangle = y_i$ для всех i.

Далее пусть $\mathbf{x}_1, \ldots, \mathbf{x}_{d+1}$ – множество $d + 1$ векторов в $\mathbb{R}^d$. Тогда должны существовать вещественные числа $a_1, \ldots, a_{d+1}$, не все равные нулю, такие, что $\sum_{i=1}^{d+1} a_i \mathbf{x}_i = \mathbf{0}$. Обозначим $I = \{i : a_i > 0\}$ и $J = \{j : a_j < 0\}$. Тогда либо I, либо J не пусто. Сначала предположим, что непусты оба множества. Тогда

$$\sum_{i \in I} a_i \mathbf{x}_i = \sum_{j \in J} |a_j| \mathbf{x}_j.$$

ЛАНЬ®

Теперь предположим, что множество $\mathbf{x}_1, \ldots, \mathbf{x}_{d+1}$ разбивается классом однородных полупространств. Тогда должен существовать вектор $\mathbf{w}$ такой, что $\langle \mathbf{w}, \mathbf{x}_i \rangle > 0$ для всех $i \in I$, тогда как $\langle \mathbf{w}, \mathbf{x}_j \rangle < 0$ для всех $j \in J$. Отсюда следует, что

$$0 < \sum_{i \in I} a_i \langle \mathbf{x}_i, \mathbf{w} \rangle = \left\langle \sum_{i \in I} a_i \mathbf{x}_i, \mathbf{w} \right\rangle = \left\langle \sum_{j \in J} |a_j| \mathbf{x}_j, \mathbf{w} \right\rangle = \sum_{j \in J} |a_j| \langle \mathbf{x}_j, \mathbf{w} \rangle < 0,$$

т. е. мы получили противоречие. Наконец, если J (соответственно, I) пусто, то правое (соответственно, левое) неравенство следует заменить равенством, что все равно приводит к противоречию. □

Теорема 9.3. *VC-размерность класса неоднородных полупространств в* $\mathbb{R}^d$ *равна* $d + 1$.

Доказательство. Во-первых, как и при доказательстве теоремы 9.2, легко проверить, что множество векторов $\mathbf{0}, \mathbf{e}_1, \ldots, \mathbf{e}_d$ разбивается классом неоднородных полупространств. Во-вторых, предположим, что множество векторов $\mathbf{x}_1, \ldots, \mathbf{x}_{d+2}$ разбивается классом неоднородных полупространств. Применив переход к однородным координатам, продемонстрированный в начале главы, мы получаем, что существует $d + 2$ векторов в $\mathbb{R}^{d+1}$, которые разделяются классом однородных полупространств. Но это противоречит теореме 9.2. □

9.2. Линейная регрессия

Линейная регрессия – стандартный статистический инструмент для моделирования связи между некоторыми «объясняющими» переменными и наблюдаемым вещественным результатом. В терминах проблемы обучения область образцов $\mathcal{X}$ является подмножеством $\mathbb{R}^d$ для некоторого d, а область меток $\mathcal{Y}$ – это множество вещественных чисел. Мы хотим обучить линейную функцию $h : \mathbb{R}^d \to \mathbb{R}$, которая оптимально аппроксимирует связь между нашими переменными (например, предсказывает вес ребенка как функцию возраста и веса при рождении). На рис. 9.1 показан пример линейно-регрессионного предиктора для $d = 1$.

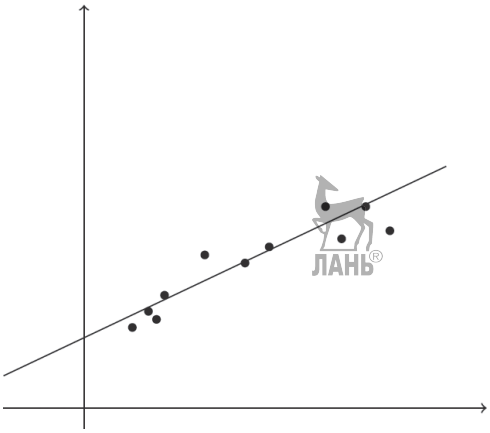


Рис. 9.1. Линейная регрессия в случае $d = 1$.
По оси x может, например, откладываться возраст ребенка,
а по оси y – его вес

Класс гипотез для линейно-регрессионных предикторов – это просто множество линейных функций

$$\mathcal{H}_{reg} = L_d = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle + b : \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}\}.$$

Далее мы должны определить функцию потерь для регрессии. Если в случае классификации определение потери очевидно, т. к. $\ell(h, (\mathbf{x}, y))$ просто показывает, правильно ли $h(\mathbf{x})$ предсказывает y, то в случае регрессии оба предсказания 3,00001 кг и 4 кг для веса 3 кг «неверны», но мы, без сомнения, предпочли бы первое. Следовательно, нужно определить, как «штрафуется» расхождение между $h(\mathbf{x})$ и y. Один из общеупотребительных способов – воспользоваться квадратичной функцией потерь

$$\ell(h, (\mathbf{x}, y)) = (h(\mathbf{x}) - y)^2.$$

Для такой функции потерь эмпирический риск называется среднеквадратичной ошибкой:

$$L_S(h) = \frac{1}{m} \sum_{i=1}^{m} (h(\mathbf{x}_i) - y_i)^2.$$

В следующем подразделе мы увидим, как реализовать правило ERM для линейной регрессии с квадратичной функцией потерь. Разумеется, существует много других функций потерь, например, потеря, равная абсолютной величина, $\ell(h, (\mathbf{x}, y)) = |h(\mathbf{x}) - y|$. Правило ERM для такой функции потерь можно реализовать с помощью линейного программирования (см. упражнение 9.1).

Отметим, что поскольку линейная регрессия не является задачей бинарной классификации, проанализировать ее выборочную сложность с помощью VC-размерности не удастся. Один из возможных способов анализа выборочной сложности линейной регрессии состоит в том, чтобы применить дискретизацию (см. замечание 4.1 в главе 4). То есть, если мы готовы удовольствоваться представлением каждого элемента вектора $\mathbf{w}$ и смещения b конечным числом битов (скажем, 64-разрядным представлением с плавающей точкой), то класс гипотез становится конечным и его размер не превышает $2^{64(d+1)}$. Мы можем положиться на границы выборочной сложности конечных классов гипотез, приведенные в главе 4. Отметим, однако, что для применения границ выборочной сложности из главы 4 необходимо, чтобы и функция потерь была ограничена. Ниже мы опишем более строгие средства анализа выборочной сложности проблем регрессии.

9.2.1. Метод наименьших квадратов

Метод наименьших квадратов – это алгоритм решения проблемы ERM для класса линейно-регрессионных предикторов относительно квадратичной функцией потерь. При заданном обучающем наборе S и при использовании однородного варианта L_d проблема ERM для этого класса состоит в нахождении величины

$$\arg\min_{\mathbf{w}} L_S(h_{\mathbf{w}}) = \arg\min_{\mathbf{w}} \frac{1}{m}\sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2.$$

Чтобы решить эту проблему, мы вычисляем градиент целевой функции и приравниваем его к нулю. То есть нужно решить уравнение

$$\frac{2}{m}\sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)\mathbf{x}_i = 0.$$

Эту задачу можно представить в виде $A\mathbf{w} = \mathbf{b}$, где

$$A = \left(\sum_{i=1}^{m}\mathbf{x}_i\mathbf{x}_i^{\top}\right) \quad \text{и} \quad \mathbf{b} = \sum_{i=1}^{m} y_i\mathbf{x}_i. \tag{9.6}$$

Или в матричной форме

$$A = \begin{pmatrix} \vdots & & \vdots \\ \mathbf{x}_1 & \dots & \mathbf{x}_m \\ \vdots & & \vdots \end{pmatrix}\begin{pmatrix} \vdots & & \vdots \\ \mathbf{x}_1 & \dots & \mathbf{x}_m \\ \vdots & & \vdots \end{pmatrix}^{\top}, \tag{9.7}$$

$$\mathbf{b} = \begin{pmatrix} \vdots & & \vdots \\ \mathbf{x}_1 & \dots & \mathbf{x}_m \\ \vdots & & \vdots \end{pmatrix}\begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}. \tag{9.8}$$

Если матрица A обратима, то решение проблемы ERM имеет вид

$$\mathbf{w} = A^{-1}\mathbf{b}.$$

Если же A необратима, то понадобятся некоторые стандартные средства линейной алгебры, приведенные в приложении С. Легко показать, что если линейное пространство, натянутое на обучающие образцы, не совпадает с $\mathbb{R}^d$, то A необратима. Тем не менее, мы всегда можем найти решение системы $A\mathbf{w} = \mathbf{b}$, поскольку $\mathbf{b}$ принадлежит образу матрицы A. Действительно, поскольку A симметрична, для нее существует спектральное разложение $A = VDV^\top$, где D – диагональная, а V – ортогональная матрица (т. е. $V^\top V$ совпадает с единичной матрицей размера $d \times d$). Определим D^+ как диагональную матрицу такую, что $D^+_{i,i} = 0$, если $D_{i,i} = 0$, а в противном случае $D^+_{i,i} = 1/D_{i,i}$. Теперь определим

$$A^+ = VD^+V \quad \text{и} \quad \hat{\mathbf{w}} = A + \mathbf{b}.$$

Обозначим $\mathbf{v}_i$ i-й столбец V. Тогда

$$A\hat{\mathbf{w}} = AA^+\mathbf{b} = VDV^\top VD^+V^\top\mathbf{b} = VDD^+V^\top\mathbf{b} = \sum_{i:D_{i,i}\neq 0}\mathbf{v}_i\mathbf{v}_i\mathbf{b}.$$

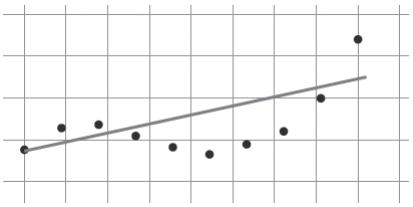
Таким образом, $A\hat{\mathbf{w}}$ – проекция $\mathbf{b}$ на линейную оболочку тех векторов $\mathbf{v}_i$, для которых $D_{i,i} \neq 0$. Поскольку линейная оболочка векторов $\mathbf{x}_1, \ldots, \mathbf{x}_m$ совпадает с линейной оболочкой таких $\mathbf{v}_i$, и $\mathbf{b}$ принадлежит линейной оболочке $\mathbf{x}_i$, то получается, что $A\hat{\mathbf{w}} = \mathbf{b}$, чем и завершается наше доказательство.

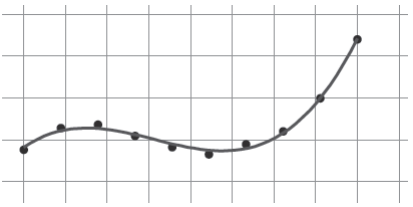
9.2.2. Линейная регрессия для задач полиномиальной регрессии

В некоторых задачах обучения возникает нужда в нелинейных, например полиномиальных, предикторах. Возьмем, к примеру, полином степени n от одной переменной:

$$p(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n,$$

где $(a_0, \ldots, a_n)$ – вектор коэффициентов размера $n + 1$. Ниже показан обучающий набор, который лучше аппроксимируется полиномом третьей степени, чем линейной функцией.





Мы будем рассматривать класс одномерных полиномиально-регрессионных предикторов:

$$\mathcal{H}^n_{poly} = \{x \mapsto p(x)\},$$

где p – одномерный полином степени n, параметризованный вектором коэффициентов $(a_0, \ldots, a_n)$. Отметим, что $\mathcal{X} = \mathbb{R}$, поскольку это полином от одной переменной, а $\mathcal{Y} = \mathbb{R}$, поскольку это проблема регрессии.

Один из способов обучить этот класс – свести задачу к проблеме линейной регрессии, которую мы уже умеем решать. Для этого определим отображение $\psi : \mathbb{R} \to \mathbb{R}^{n+1}$ такое, что $\psi(x) = (1, x, x^2, \ldots, x^n)$. Тогда

$$p(\psi(x)) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n = \langle \mathbf{a}, \psi(x) \rangle,$$

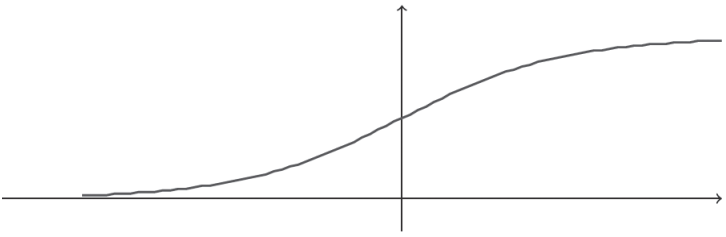
и мы можем найти оптимальный вектор коэффициентов $\mathbf{a}$, применив метод наименьших квадратов, как было показано выше.

9.3. Логистическая регрессия

В случае логистической регрессии мы обучаем семейство функций h из $\mathbb{R}^d$ в отрезок [0, 1]. Однако логистическая регрессия используется для задач классификации: мы можем интерпретировать $h(\mathbf{x})$ как *вероятность* того, что метка $\mathbf{x}$ равна 1. Класс гипотез, ассоциированный с логистической регрессией, – это композиция сигмоиды $\varphi_{\text{sig}} : \mathbb{R} \to [0, 1]$ с классом линейных функций L_d. В качестве сигмоиды обычно используется *логистическая функция*:

$$\varphi_{\text{sig}}(z) = \frac{1}{1 + \exp(-z)}. \tag{9.9}$$

Слово «сигмоида» означает «S-образная», и это понятно, поскольку график этой функции выглядит, как показано на рисунке ниже.



Таким образом, класс гипотез (для простоты мы используем однородные линейные функции) имеет вид:

$$H_{\text{sig}} = \varphi_{\text{sig}} \circ L_d = \{\mathbf{x} \mapsto \varphi_{\text{sig}}(\langle \mathbf{w}, \mathbf{x} \rangle) : \mathbf{w} \in \mathbb{R}^d\}.$$

Отметим, что если $\langle \mathbf{w}, \mathbf{x} \rangle$ очень велико, то $\varphi_{\text{sig}}(\langle \mathbf{w}, \mathbf{x} \rangle)$ близко к 1, а если $\langle \mathbf{w}, \mathbf{x} \rangle$ очень мало, то $\varphi_{\text{sig}}(\langle \mathbf{w}, \mathbf{x} \rangle)$ близко к 0. Напомним, что для полупространства, соответствующего вектору $\mathbf{w}$, предсказанием является $\text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$. Поэтому предсказания полупространственной и логистической гипотез очень похожи, когда $|\langle \mathbf{w}, \mathbf{x} \rangle|$ велико. Но если $|\langle \mathbf{w}, \mathbf{x} \rangle|$ близко к 0, то $\varphi_{\text{sig}}(\langle \mathbf{w}, \mathbf{x} \rangle) \approx 1/2$. Интуитивно понятно, что логистическая гипотеза не уверена в значении метки, поэтому предполагает, что метка равна $\text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$ с вероятностью, чуть больше 50%. Напротив,

полупространственная гипотеза всегда выдает детерминированное предсказание – 1 или –1 – даже если $|\langle \mathbf{w}, \mathbf{x} \rangle|$ очень близко к 0.

Далее нужно задать функцию потерь. То есть мы должны определить, насколько плохо предсказывать некоторое значение $h_{\mathbf{w}}(\mathbf{x}) \in [0, 1]$ при условии, что истинная метка $y \in \{\pm 1\}$. Очевидно, что нам хотелось бы, чтобы $h_{\mathbf{w}}(\mathbf{x})$ было велико, если $y = 1$, и чтобы $1 - h_{\mathbf{w}}(\mathbf{x})$ (вероятность предсказания –1) было велико, если $y = -1$. Заметим, что

$$1 - h_{\mathbf{w}}(\mathbf{x}) = 1 - \frac{1}{1 + \exp(-\langle \mathbf{w}, \mathbf{x} \rangle)} = \frac{\exp(-\langle \mathbf{w}, \mathbf{x} \rangle)}{1 + \exp(-\langle \mathbf{w}, \mathbf{x} \rangle)} = \frac{1}{1 + \exp(\langle \mathbf{w}, \mathbf{x} \rangle)}.$$

Поэтому любая разумная функция потерь должна монотонно возрастать вместе с $1/(1 + \exp(y\langle \mathbf{w}, \mathbf{x} \rangle))$ или, что эквивалентно, монотонно возрастать вместе с $1 + \exp(-y\langle \mathbf{w}, \mathbf{x} \rangle)$. Логистическая функция потерь, используемая в логистической регрессии, штрафует $h_{\mathbf{w}}$ в зависимости от величины логарифма $1 + \exp(-y\langle \mathbf{w}, \mathbf{x} \rangle)$ (напомним, что логарифм – монотонная функция). Это означает, что

$$\ell(h_{\mathbf{w}}, (\mathbf{x}, y)) = \log(1 + \exp(-y\langle \mathbf{w}, \mathbf{x} \rangle)).$$

Следовательно, при заданном обучающем наборе $S = (\mathbf{x}_1, 1), \ldots, (\mathbf{x}_m, y_m)$ проблема ERM, ассоциированная с логистической регрессией, имеет вид

$$\operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 + \exp(-\langle \mathbf{w}, \mathbf{x} \rangle)\right). \tag{9.10}$$

Преимущество логистической функции потерь заключается в том, что она *выпукла* относительно $\mathbf{w}$, поэтому проблему ERM можно эффективно решить стандартными методами. В последующих главах мы изучим, как обучать выпуклые функции и, в частности, сформулируем простой алгоритм минимизации выпуклых функций.

Проблема ERM, ассоциированная с логистической регрессией (выражение (9.10)), совпадает с проблемой нахождения оценки максимального правдоподобия – хорошо известным в статистике подходом к отысканию параметров, максимизирующих совместную вероятность заданного набора данных в предположении определенной параметрической функции вероятности. Мы будем изучать этот подход в главе 24.

9.4. Резюме

Семейство линейных предикторов – один из самых полезных классов гипотез, и многие широко используемые на практике алгоритмы обучения опираются на линейные предикторы. Мы продемонстрировали эффективные алгоритмы обучения линейных предикторов с бинарной функцией потерь в разделимом случае и с квадратичной и логистической функцией потерь в нереализуемом случае. В последующих главах мы расскажем о свойствах функции потерь, обеспечивающих эффективность обучения.

Естественно, линейные предикторы эффективны только в том случае, когда мы в качестве априорного знания предполагаем, что существует линейный пре-

диктор с малым риском относительно истинного распределения. В следующей главе мы покажем, как построить нелинейные предикторы с помощью композиции линейных предикторов с простыми классами. Это позволит применять линейные предикторы для широкого круга априорных предположений.

9.5. Библиографические сведения

Алгоритм перцептрона восходит к работе Rosenblatt (1958). Доказательство его сходимости дано в работах Agmon (1954), Novikoff (1962). Регрессия методом наименьших квадратов описана в работах Gauss (1795), Legendre (1805) и Adrain (1808).

9.6. Упражнения

9.1. Покажите, как свести проблему ERM линейной регрессии с абсолютной величиной в качестве функции потерь $\ell(h,(\mathbf{x}, y)) = |h(\mathbf{x}) - y|$ к линейной программе, т. е. покажите, как записать проблему

$$\min_{\mathbf{w}} \sum_{i=1}^{m} |\langle \mathbf{w}, \mathbf{x} \rangle - y_i|$$

в виде линейной программы.

Указание. Сначала докажите, что для любого $c \in \mathbb{R}$

$$|c| = \min_{a \geq 0} a \quad \text{s.t.} \quad c \leq a \text{ и } c \geq -a.$$

9.2. Покажите, что матрица A, определенная в формуле (9.6), обратима тогда и только тогда, когда линейной оболочкой векторов $\mathbf{x}_1, \ldots, \mathbf{x}_m$ является все пространство $\mathbb{R}^d$.

9.3. Покажите, что теорема 9.1 точна в следующем смысле: для любого целого положительного числа m существует вектор $\mathbf{w}^* \in \mathbb{R}^d$ (для некоторого d) и последовательность примеров $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ такая, что выполняются следующие утверждения:

- $R = \max_i \|\mathbf{x}_i\| \leq 1$;
- $\|\mathbf{w}^*\|^2 = m$ и для всех $i \leq m$, $y_i \langle \mathbf{x}_i, \mathbf{w}^* \rangle \geq 1$. Заметим, что в обозначениях теоремы 9.1 мы получаем отсюда, что

$$B = \min\{\|\mathbf{w}\| : \forall i \in [m], y_i \langle \mathbf{w}, x_i \rangle \geq 1\} \leq \sqrt{m}.$$

  Таким образом, $(BR)^2 \leq m$;
- при выполнении алгоритма перцептрона на этой последовательности примеров до сходимости необходимо произвести m обновлений.

Указание. Выберите $d = m$ и для каждого i выберите $\mathbf{x}_i = \mathbf{e}_i$.

9.4. (*) Для произвольного заданного числа m приведите пример последовательности помеченных образцов $((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)) \in (\mathbb{R}^3 \times \{-1, +1\})^m$, для которой верхняя граница из теоремы 9.1 равна m и перцептрон неизбежно допускает m ошибок.

Указание. Пусть $\mathbf{x}_i$ – трехмерный вектор вида (a, b, y_i), где $a^2 + b^2 = R^2 - 1$. Пусть $\mathbf{w}^*$ – вектор $(0, 0, 1)$. Теперь внимательно проанализируйте доказательство верхней границы перцептрона (теорема 9.1), посмотрите, где мы использовали неравенства (≤) вместо равенств (=) и придумайте сценарии, когда неравенство на самом деле обращается в равенство.

9.5. Рассмотрим такую модификацию алгоритма перцептрона: на шаге обновления вместо присваивания $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + y_i\mathbf{x}_i$ всякий раз, как допущена ошибка, мы производим присваивание $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \eta y_i\mathbf{x}_i$ для некоторого $\eta > 0$. Докажите, что модифицированный перцептрон выполняет столько же итераций, сколько исходный, и сходится к вектору, указывающему то же направление, что и в исходном алгоритме.

9.6. В этом упражнении мы получим границы VC-размерности класса (замкнутых) шаров в $\mathbb{R}^d$, т. е.

$$B_d = \{B_{\mathbf{v},r} : \mathbf{v} \in \mathbb{R}^d, r > 0\},$$

где

$$B_{\mathbf{v},r}(\mathbf{x}) = \begin{cases} 1, & \text{если } \|\mathbf{x} - \mathbf{v}\| \le r \\ 0, & \text{в противном случае} \end{cases}.$$

1. Рассмотрим отображение $\varphi: \mathbb{R}^d \to \mathbb{R}^{d+1}$, определенное следующим образом: $\varphi(\mathbf{x}) = (\mathbf{x}, \|\mathbf{x}\|^2)$. Покажите, что если множество $\mathbf{x}_1, \ldots, \mathbf{x}_m$ разбивается классом B_d, то $\varphi(\mathbf{x}_1), \ldots, \varphi(\mathbf{x}_m)$ разбивается классом полупространств в $\mathbb{R}^{d+1}$(в этом упражнении мы предполагаем, что sign(0) = 1). Что это говорит нам о величине $\mathrm{VCdim}(B_d)$?
2. (*) Найдите множество $d + 1$ точек в R^d, которое разбивается классом B_d. Выведите отсюда, $d + 1 \le \mathrm{VCdim}(B_d) \le d + 2$.

ЛАНЬ®

Глава 10

УСИЛЕНИЕ

Усиление (boosting) – алгоритмический подход, который, выросши из одного теоретического вопроса, стал весьма распространенным на практике инструментом машинного обучения. В нем используется обобщение линейных предикторов для решения двух главных проблем, поднятых в этой книге ранее. Первая – компромисс между смещением и сложностью. Мы видели (в главе 5), что ошибку алгоритма обучения ERM можно представить в виде суммы *ошибки аппроксимации* и *ошибки оценивания*. Чем выразительнее класс гипотез, в котором осуществляет поиск обучаемый, тем меньше ошибка аппроксимации, но ошибка оценивания при этом растет. Таким образом, перед обучаемым стоит проблема выбора компромисса между двумя крайностями. Идея усиления позволяет более точно управлять этим выбором. Обучаемый начинает с простого класса (возможно, с большой ошибкой аппроксимации), а по ходу обучения класс, которому может принадлежать предиктор, становится все шире.

Вторая проблема, решаемая усилением, – вычислительная сложность обучения. В главе 8 мы видели, что для многих интересных классов концептов задача нахождения ERM-гипотезы может оказаться вычислительно неосуществимой. Алгоритм усиления повышает верность *слабых обучаемых*. На интуитивном уровне слабого обучаемого можно представлять себе как алгоритм, в котором простая эвристика используется для вывода гипотезы, принадлежащей легкому для обучения классу и лишь немного превосходящей случайную догадку. Если слабого обучаемого можно реализовать эффективно, то усиление дает способ агрегирования таких слабых гипотез с целью аппроксимировать постепенно улучшающиеся предикторы для более обширных и трудных для обучения классов.

В этой главе мы опишем и проанализируем практически полезный алгоритм усиления AdaBoost (аббревиатура Adaptive Boosting – адаптивное усиление). Этот алгоритм выводит гипотезу, являющуюся линейной комбинацией простых гипотез. Иными словами, AdaBoost опирается на семейство классов гипотез, полученных путем композиции линейного предиктора с простыми классами. Мы покажем, что AdaBoost позволяет управлять выбором компромисса между ошибками аппроксимации и оценивания посредством варьирования всего лишь одного параметра.

AdaBoost – пример общей темы, которая не раз встретится в этой книге: повышение выразительности линейных предикторов путем их композиции с другими функциями. Мы вернемся к ней в разделе 10.3.

Истоки AdaBoost следует искать в теоретическом вопросе о том, можно ли «усилить» эффективного слабого обучаемого, превратив его в эффективного сильного обучаемого. Этот вопрос был поднят в работе Кернса и Валианта в 1988 г. и разрешен в 1990 г. Робертом Шапире, тогда аспирантом Массачусетского технологического института. Однако предложенный им механизм был не слишком практичен. В 1995 г. Роберт Шапире и Йоав Фройнд (Yoav Freund) предложили алгоритм AdaBoost, который стал первой практически полезной реализацией усиления. Этот простой и элегантный алгоритм приобрел огромную популярность, а работа Фройнда и Шапире была отмечена многочисленными наградами.

Ко всему прочему, усиление – прекрасный пример практической пользы теории обучения. Возникнув как чисто теоретическая проблема, идея усиления привела к разработке популярных и широко используемых алгоритмов. Как мы продемонстрируем ниже в этой главе, AdaBoost успешно применялся для обучения распознаванию лиц в изображениях.

10.1. Слабая обучаемость

Напомним определение PAC-обучения из главы 3: класс гипотез $\mathcal{H}$ является PAC-обучаемым, если существуют функция $m_{\mathcal{H}} : (0, 1)^2 \to \mathbb{N}$ и алгоритм обучения, обладающие следующим свойством: для любых $\epsilon, \delta \in (0, 1)$, для любого распределения $\mathcal{D}$ на $\mathcal{X}$ и для любой функции пометки $f : \mathcal{X} \to \{\pm 1\}$ справедливо следующее утверждение: если для $\mathcal{H}$, $\mathcal{D}$ и f имеет место предположение о реализуемости, то при обучении на $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ примерах, независимо выбранных из распределения $\mathcal{D}$ и помеченных функцией f, алгоритм возвращает гипотезу h такую, что с вероятностью не менее $1 - \delta$ выполняется неравенство $L_{(D,f)}(h) \leq \epsilon$.

Далее, фундаментальная теорема теории обучения (теорема 6.8 из главы 6) характеризует семейство обучаемых классов и утверждает, что любой PAC-обучаемый класс можно обучить с помощью алгоритма ERM. Однако в определении PAC-обучения и в фундаментальной теореме игнорируется вычислительный аспект обучения. Действительно, как было показано в главе 8, существуют случаи, когда реализация правила ERM оказывается вычислительно трудной (даже при выполнении предположения о реализуемости).

Но, быть может, нам удастся справиться с вычислительной трудностью, пожертвовав требованием к верности? Быть может, для данного распределения $\mathcal{D}$ и целевой функции обучения f существует эффективно вычислимый алгоритм обучения, ошибка которого лишь чуть-чуть лучше, чем у случайной догадки? Попытка ответить на эти вопросы приводит к такому определению.

Определение 10.1 (γ–слабая обучаемость).

- Говорят, что A является γ-слабым алгоритмом обучения для класса $\mathcal{H}$, если существует функция $m_{\mathcal{H}} : (0, 1)^2 \to \mathbb{N}$ такая, что для любого $\delta \in (0, 1)$, для любого распределения $\mathcal{D}$ на $\mathcal{X}$ и для любой функции пометки $f : \mathcal{X} \to \{\pm 1\}$ справедливо следующее утверждение: если для $\mathcal{H}$, $\mathcal{D}$ и f имеет место предположение о реализуемости, то при обучении на $m \geq m_{\mathcal{H}}(\delta)$ примерах, независимо выбранных из распределения $\mathcal{D}$ и помеченных функцией f, алгоритм возвращает гипотезу h такую, что с вероятностью не менее $1 - \delta$ выполняется неравенство $L_{(\mathcal{D},f)}(h) \leq 1/2 - \gamma$.

- Класс гипотез $\mathcal{H}$ называется γ-слабо обучаемым, если для него существует γ-слабый алгоритм обучения.

Это определение почти совпадает с определением PAC-обучения, которое мы здесь будем называть *сильным обучением*, но есть одно существенное отличие: сильная обучаемость подразумевает возможность найти сколь угодно хороший классификатор (с ошибкой обучения не более ϵ для сколь угодно малого $\epsilon > 0$). А в случае слабой обучаемости нам нужно только вывести гипотезу, ошибка которой не превышает $1/2 - \gamma$, т. е. лишь немногим лучше ошибки, которую мы получили бы при случайной пометке. Мы надеемся, что найти эффективных слабых обучаемых будет проще, чем эффективных (настоящих) PAC-обучаемых.

Фундаментальная теорема об обучении (теорема 6.8) говорит, что если класс гипотез $\mathcal{H}$ имеет VC-размерность d, то выборочная сложность PAC-обучения $\mathcal{H}$ удовлетворяет неравенству $m_{\mathcal{H}}(\epsilon, \delta) \geq C_1(d + \log(1/\delta))$, где C_1 – константа. Подставив сюда $\epsilon = 1/2 - \gamma$, мы сразу же получаем, что при $d = \infty$ класс $\mathcal{H}$ не является γ-слабо обучаемым. Отсюда следует, что со статистической точки зрения (если не обращать внимания на вычислительную сложность) слабая обучаемость также характеризуется VC-размерностью $\mathcal{H}$ и потому оказывается столь же трудной задачей, как PAC-обучение (сильное). Но если принять в расчет вычислительную сложность, то потенциальное преимущество слабого обучения заключается в том, что может существовать алгоритм, удовлетворяющий требованиям слабого обучения и допускающий эффективную реализацию.

Один из возможных подходов состоит в том, чтобы взять «простой» класс гипотез B и применить ERM к B в качестве слабого алгоритма обучения. Для этого B должен удовлетворять двум требованиям:

- ERM_B допускает эффективную реализацию;
- для любой выборки, помеченной некоторой гипотезой из класса $\mathcal{H}$, любая ERM_B-гипотеза будет иметь ошибку, не превышающую $1/2 - \gamma$.

Тогда возникает вопрос: можем ли мы усилить *эффективного* слабого обучаемого, превратив его в *эффективного* сильного обучаемого. В следующем разделе мы покажем, что это действительно возможно, но сначала приведем пример, в котором эффективная слабая обучаемость класса $\mathcal{H}$ возможна с применением базового класса гипотез B.

Пример 10.1 (слабая обучаемость трехчастных классификаторов с помощью решающих пней). Пусть $\mathcal{X} = \mathbb{R}$, и $\mathcal{H}$ – класс трехчастных классификаторов, т. е. $\mathcal{H} = \{h_{\theta_1,\theta_2,b} : \theta_1, \theta_2 \in \mathbb{R}, \theta_1 < \theta_2, b \in \{\pm 1\}\}$, и для любого x,

$$h_{\theta_1,\theta_2,b}(x) = \begin{cases} +b, & \text{если } x < \theta_1 \text{ или } x > \theta_2 \\ -b, & \text{если } \theta_1 \leq x \leq \theta_2 \end{cases}.$$

На рисунке ниже изображен пример гипотезы (для $b = 1$):

Пусть B – класс решающих пней, т. е. $B = \{x \mapsto \text{sign}(x - \theta) \cdot b : \theta \in \mathbb{R}, b \in \{\pm 1\}\}$. Далее мы покажем, что ERM_B – γ-слабый обучаемый для $\mathcal{H}$ при $\gamma = 1/12$. Чтобы

убедиться в этом, мы сначала покажем, что для любого распределения, согласованного с $\mathcal{H}$, существует решающий пень с $L_{\mathcal{D}}(h) \le 1/3$. В самом деле, просто заметим, что любой классификатор, принадлежащий $\mathcal{H}$, состоит из трех участков (два неограниченных луча и средний интервал) с чередующимися метками. Для любой пары таких участков существует решающий пень, согласованный с пометкой этих двух компонент. Заметим, что для любого распределения $\mathcal{D}$ на $\mathbb{R}$ и любого разбиения прямой на три таких участка один из участков будет иметь $\mathcal{D}$-вес не более 1/3. Пусть $h \in \mathcal{H}$ – гипотеза с нулевой ошибкой. Решающий пень, не согласованный с h, только на таком участке будет иметь ошибку, и она не превысит 1/3.

Наконец, VC-размерность класса решающих пней равна 2, а это значит, что если размер выборки больше $\Omega(\log(1/\delta)/\epsilon^2)$, то с вероятностью не менее $1 - \delta$ правило ERM_B возвращает гипотезу с ошибкой, не превышающей $1/3 + \epsilon$. Положив $\epsilon = 1/12$, мы получим, что ошибка ERM_B не превышает $1/3 + 1/12 = 1/2 - 1/12$.

Как видим, ERM_B является γ-слабым алгоритмом обучения $\mathcal{H}$. Далее мы покажем, как эффективно реализовать правило ERM для класса решающих пней.

10.1.1. Эффективная реализация ERM для класса решающих пней

Пусть $\mathcal{X} = \mathbb{R}^d$, и рассмотрим базовый класс гипотез, состоящий из решающих пней над $\mathbb{R}^d$:

$$\mathcal{H}_{\mathrm{DS}} = \{\mathbf{x} \mapsto \mathrm{sign}(\theta - x_i) \cdot b : \theta \in \mathbb{R}, i \in [d], b \in \{\pm 1\}\}.$$

Для простоты предположим, что $b = 1$, т. е. нас интересуют все гипотезы класса $\mathcal{H}_{\mathrm{DS}}$ вида $\mathrm{sign}(\theta - x_i)$. Пусть $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$ – обучающий набор. Мы покажем, как реализовать правило ERM, т. е. как найти решающий пень, который минимизирует функцию $L_S(h)$. А поскольку в следующем разделе мы покажем, что для применения алгоритма AdaBoost нужно найти гипотезу с небольшим риском относительно некоторого распределения выборок S, то здесь будет показано, как минимизировать такие функции риска. Конкретно, пусть $\mathbf{D}$ – вектор вероятностей в $\mathbb{R}^m$ (т. е. все элементы $\mathbf{D}$ неотрицательны и $\sum_i D_i = 1$). Слабый обучаемый, который мы опишем ниже, получает $\mathbf{D}$ и S и выводит решающий пень $h : \mathcal{X} \to \mathcal{Y}$, доставляющий минимум риску относительно $\mathbf{D}$:

$$L_{\mathbf{D}}(h) = \sum_{i=1}^{m} D_i \mathbb{1}_{[h(\mathbf{x}_i) \neq y_i]}.$$

Заметим, что если $\mathbf{D} = (1/m, \ldots, 1/m)$, то $L_{\mathbf{D}}(h) = L_S(h)$.

Напомним, что всякий решающий пень параметризован индексом $j \in [d]$ и порогом θ. Поэтому минимизация $L_{\mathbf{D}}(h)$ сводится к решению задачи

$$\min_{j \in [d]} \min_{\theta \in \mathbb{R}} \left(\sum_{i: y_i = 1} D_i \mathbb{1}_{[x_{i,j} > \theta]} + \sum_{i: y_i = -1} D_i \mathbb{1}_{[x_{i,j} \le \theta]} \right). \tag{10.1}$$

Зафиксируем $j \in [d]$ и отсортируем примеры, так чтобы $x_{1,j} \le x_{2,j} \le \ldots \le x_{m,j}$. Определим $\Theta_j = \{(x_{i,j} + x_{i+1,j})/2 : i \in [m-1]\} \cup \{(x_{1,j} - 1), (x_{m,j} + 1)\}$. Заметим, что для любого порога $\theta \in \mathbb{R}$ существует $\theta' \in \Theta_j$, дающий такие же предсказания для выборки

S, что и θ. Поэтому вместо минимизации по $\theta \in \mathbb{R}$ мы можем минимизировать по $\theta \in \Theta_j$.

Тем самым мы получаем эффективную процедуру: выбрать такие $j \in [d]$ и $\theta \in \Theta_j$, которые минимизируют целевую величину (10.1). Для любых j и $\theta \in \Theta_j$ мы должны вычислить сумму по m примерам; следовательно, время работы этого подхода составляет $O(dm^2)$. Далее мы продемонстрируем прием, который позволит минимизировать целевую величину за время $O(dm)$.

Наблюдение состоит в следующем. Предположим, что мы вычислили целевую величину для $\theta \in (x_{i-1,j}, x_{i,j})$. Пусть $F(\theta)$ – ее вычисленное значение. Тогда, если взять $\theta' \in (x_{i,j}, x_{i+1,j})$, то будем иметь

$$F(\theta') = F(\theta) - D_i \mathbb{1}_{[y_i=1]} + D_i \mathbb{1}_{[y_i=-1]} = F(\theta) - y_i D_i.$$

Следовательно, мы можем вычислить целевую величину в точке θ' за постоянное время, если известна целевая величина в предыдущей точке θ. Отсюда следует, что после шага предобработки, на котором мы сортируем примеры по каждой координате, задачу минимизации можно будет решить за время $O(dm)$. Таким образом, мы получаем следующий псевдокод.

ERM для решающих пней

вход:
  обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
  вектор вероятностей $\mathbf{D}$
цель: найти j^*, θ^*, доставляющие минимум выражению (10.1)
инициализация: $F^* = \infty$
for $j = 1, \ldots, d$
  отсортируем S по j-й координате и обозначим
    $x_{1,j} \le x_{2,j} \le \cdots \le x_{m,j} \le x_{m+1,j} \stackrel{\text{def}}{=} x_{m,j+1}$
  $F = \sum_{i:y_i=1} D_i$
  if $F < F^*$
    $F^* = F$, $\theta^* = x_{1,j-1}$, $j^* = j$
  for $i = 1, \ldots, m$
    $F = F - y_i D_i$
    if $F < F^*$ и $x_{i,j} \neq x_{i+1,j}$
      $F^* = F$, $\theta^* = \frac{1}{2}(x_{i,j} + x_{i+1,j})$, $j^* = j$
вывести j^*, θ^*

10.2. Алгоритм AdaBoost

AdaBoost (аббревиатура Adaptive Boosting) – алгоритм, который имеет доступ к слабому обучаемому и находит гипотезу с низким эмпирическим риском. Он принимает на входе обучающий набор примеров $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, где для любого $iy_i = f(\mathbf{x}_i)$ при некоторой обучающей функции f. Процесс усиления представляет собой последовательность раундов. На t-м раунде усилитель сначала определяет распределение на примерах из S, обозначаемое $\mathbf{D}^{(t)}$. Это означает, что $\mathbf{D}^{(t)} \in \mathbb{R}_+^m$ и $\sum_{i=1}^m D_i^{(t)} = 1$. Затем усилитель передает распределение $\mathbf{D}^{(t)}$ и выборку

S слабому обучаемому. Таким образом, слабый обучаемый может построить независимые и одинаково распределенные примеры согласно $\mathbf{D}^{(t)}$ и f. Предполагается, что слабый обучаемый возвращает «слабую» гипотезу h_t, ошибка которой

$$\epsilon_t = L_{\mathbf{D}^{(t)}}(h_t) = \sum_{i=1}^{m} D_i^{(t)} \mathbb{1}_{[h_t(\mathbf{x}_i) \neq y_i]}$$

не превышает $1/2 - \gamma$ (разумеется, существует вероятность, не превышающая δ, что слабый обучаемый потерпит неудачу). Затем AdaBoost назначает вес гипотезе h_t следующим образом: $w_t = \frac{1}{2}\log(1/\epsilon_t - 1)$. Значит, вес h_t обратно пропорционален ошибке h_t. В конце раунда AdaBoost обновляет распределение, так чтобы примеры, на которых h_t ошибается, получили большую массу вероятности, тогда как примеры, на которых h_t дает правильный результат, – меньшую. Интуитивно понятно, что это заставит слабого обучаемого на следующем раунде сконцентрироваться на проблематичных примерах. На выходе алгоритма AdaBoost получается «сильный» классификатор, основанный на взвешенной сумме всех слабых гипотез. Ниже представлен псевдокод AdaBoost.

ЛАНЬ®

AdaBoost

вход:
обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
слабый обучаемый WL
количество раундов T
инициализация $\mathbf{D}^{(1)} = (1/m, \ldots, 1/m)$.
for $t = 1, \ldots, T$:
вызвать слабого обучаемого $h_t = \mathrm{WL}(\mathbf{D}^{(t)}, S)$
вычислить $\epsilon_t = \sum_{i=1}^{m} D_i^{(t)} \mathbb{1}_{[y_i \neq h_t(\mathbf{x}_i)]}$
положить $w_t = \frac{1}{2}\log(1/\epsilon_t - 1)$
обновить $D_i^{(t+1)} = \dfrac{D_i^{(t)} \exp(-w_t y_i h_t(\mathbf{x}_i))}{\sum_{j=1}^{m} D_j^{(t)} \exp(-w_t y_j h_t(\mathbf{x}_j))}$ для всех $i = 1, \ldots, m$
вывести гипотезу $h_s(\mathbf{x}) = \mathrm{sign}(\sum_{t=1}^{T} w_t h_t(\mathbf{x}))$

ЛАНЬ®

Следующая теорема показывает, что ошибка обучения выходной гипотезы экспоненциально убывает вместе с количеством раундов усиления.

Теорема 10.2. *Пусть S – обучающий набор, и предположим, что на каждой итерации алгоритма AdaBoost слабый обучаемый возвращает гипотезу, для которой $\epsilon_t \leq 1/2 - \gamma$. Тогда ошибка обучения выходной гипотезы AdaBoost не превышает*

$$L_S(h_s) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{[h_s(\mathbf{x}_i) \neq y_i]} \leq \exp(-2\gamma^2 T).$$

Доказательство. Для каждого t обозначим $f_t = \sum_{p \leq t} w_p h_p$. Тогда выходом AdaBoost является f_T. Кроме того, обозначим

$$Z_t = \frac{1}{m} \sum_{i=1}^{m} e^{-y_i f_t(x_i)}.$$

Заметим, что для любой гипотезы имеем $\mathbb{1}_{[h(x)\neq y]} \leq e^{-yh(x)}$. Поэтому $L_S(f_T) \leq Z_T$, так что достаточно доказать, что $Z_T \leq e^{-2\gamma^2 T}$. Чтобы ограничить величину Z_T сверху, перепишем ее в виде

$$Z_T = \frac{Z_T}{Z_0} = \frac{Z_T}{Z_{T-1}} \cdot \frac{Z_{T-1}}{Z_{T-2}} \cdots \frac{Z_2}{Z_1} \cdot \frac{Z_1}{Z_0}, \tag{10.2}$$

где мы воспользовались тем фактом, что $Z_0 = 1$, поскольку $f_0 \equiv 0$. Следовательно, достаточно показать, что на каждом раунде t

$$\frac{Z_{t+1}}{Z_t} \leq e^{-2\gamma^2}. \tag{10.3}$$

Для этого мы сначала заметим, что, рассуждая по индукции, легко убедиться в том, что для всех t и i имеет место равенство

$$D_i^{(t+1)} = \frac{e^{-y_i f_t(x_i)}}{\sum_{j=1}^m e^{-y_j f_t(x_j)}}.$$

Отсюда

$$\begin{aligned}
\frac{Z_{t+1}}{Z_t} &= \frac{\sum_{i=1}^m e^{-y_i f_{t+1}(x_i)}}{\sum_{j=1}^m e^{-y_j f_t(x_j)}} \\
&= \frac{\sum_{i=1}^m e^{-y_i f_t(x_i)} e^{-y_i w_{t+1} h_{t+1}(x_i)}}{\sum_{j=1}^m e^{-y_j f_t(x_j)}} \\
&= \sum_{i=1}^m D_i^{(t+1)} e^{-y_i w_{t+1} h_{t+1}(x_i)} \\
&= e^{-w_{t+1}} \sum_{i: y_i h_{t+1}(x_i)=1} D_i^{(t+1)} + e^{w_{t+1}} \sum_{i: y_i h_{t+1}(x_i)=-1} D_i^{(t+1)} \\
&= e^{-w_{t+1}}(1-\epsilon_{t+1}) + e^{w_{t+1}} \epsilon_{t+1} \\
&= \frac{1}{\sqrt{1/\epsilon_{t+1} - 1}}(1-\epsilon_{t+1}) + \sqrt{1/\epsilon_{t+1} - 1}\, \epsilon_{t+1} \\
&= \sqrt{\frac{\epsilon_{t+1}}{1-\epsilon_{t+1}}}\,(1-\epsilon_{t+1}) + \sqrt{\frac{1-\epsilon_{t+1}}{\epsilon_{t+1}}}\,\epsilon_{t+1} \\
&= 2\sqrt{\epsilon_{t+1}(1-\epsilon_{t+1})}.
\end{aligned}$$

Согласно нашему предположению, $\epsilon_{t+1} \leq 1/2 - \gamma$. Так как функция $g(a) = a(1 - a)$ монотонно возрастает на отрезке $[0, 1/2]$, то получаем

$$2\sqrt{\epsilon_{t+1}(1-\epsilon_{t+1})} \leq 2\sqrt{\left(\frac{1}{2}-\gamma\right)\left(\frac{1}{2}+\gamma\right)} = \sqrt{1-4\gamma^2}.$$

Наконец, воспользовавшись неравенством $1 - a \le e^{-a}$, получаем, что $\sqrt{1-4\gamma^2} \le e^{-4\gamma^2/2} = e^{-2\gamma^2}$. Это доказывает справедливость неравенства (10.3), а вместе с тем и теоремы. □

На каждой итерации AdaBoost производится $O(m)$ операций и одно обращение к слабому обучаемому. Поэтому, если слабый обучаемый может быть реализован эффективно (как в случае ERM и класса решающих пней), то и весь процесс обучения будет эффективен.

Замечание 10.2. В теореме 10.2 предполагается, что на каждой итерации AdaBoost слабый обучаемый возвращает гипотезу, для которой взвешенная ошибка выборки не превосходит $1/2 - \gamma$. По определению, слабый обучаемый может потерпеть неудачу с вероятностью δ. Согласно лемме о границе объединения, вероятность, что слабый обучаемый ни разу не ошибется на всех итерациях, составляет не менее $1 - \delta T$. В упражнении 10.1 мы покажем, что зависимость выборочной сложности от δ всегда может быть сделана логарифмической по $1/\delta$, и, следовательно, вызов слабого обучаемого с очень малым значением δ – не проблема. Стало быть, мы можем предположить, что δT тоже мало. Далее, поскольку слабый обучаемый применяется только совместно с распределениями на обучающем наборе, во многих случаях его можно реализовать так, чтобы вероятность неудачи была нулевой ($\delta = 0$). Так, например, обстоит дело с описанным в предыдущем разделе слабым обучаемым, который находит минимальное значение $L_D(h)$ для решающих пней.

Теорема 10.2 говорит, что эмпирический риск гипотезы, построенной алгоритмом AdaBoost, стремится к нулю с ростом T. Однако нас-то интересует истинный риск выходной гипотезы. Чтобы иметь возможность рассуждать об истинном риске, заметим, что на самом деле выходом AdaBoost является композиция полупространства с предсказаниями T слабых гипотез, построенных слабым обучаемым. В следующем разделе мы покажем, что если слабые гипотезы берутся из базового класса гипотез низкой VC-размерности, то ошибка оценивания AdaBoost будет малой, т. е. истинный риск выходной гипотезы AdaBoost не слишком сильно отличается от ее эмпирического риска.

10.3. Линейные комбинации базовых гипотез

Как уже отмечалось, популярный подход к построению слабого обучаемого состоит в том, чтобы применить правило ERM к базовому классу гипотез (например, ERM для класса решающих пней). Мы также видели, что выходом алгоритма усиления является композиция полупространства с предсказаниями слабых гипотез. Поэтому, если дан базовый класс гипотез B (например, решающие пни), то выходом AdaBoost будет член следующего класса:

$$L(B,T) = \left\{ x \mapsto \operatorname{sign}\left(\sum_{t=1}^{T} w_t h_t(x) \right) : w \in \mathbb{R}^T,\ \forall t,\quad h_t \in B \right\}. \tag{10.4}$$

Это значит, что каждая гипотеза $h \in L(B, T)$ параметризована T базовыми гипотезами из B и вектором $\mathbf{w} \in \mathbb{R}^T$. Предсказание такой гипотезы h для образца

x получается следующим образом: сначала применить T базовых гипотез для построения вектора $\psi(x) = (h_1(x), \ldots, h_T(x)) \in \mathbb{R}^T$, а затем применить (однородное) полупространство, определенное вектором $\mathbf{w}$ к $\psi(x)$.

В этом разделе мы проанализируем ошибку оценивания $L(B, T)$, выразив верхнюю границу VC-размерности $L(B, T)$ в терминах VC-размерности B и T. Мы покажем, что с точностью до логарифмических множителей VC-размерность $L(B, T)$ ограничена произведением T на VC-размерность B. Отсюда следует, что ошибка оценивания AdaBoost растет линейно с ростом T. С другой стороны, эмпирический риск AdaBoost уменьшается с ростом T. На самом деле, как мы покажем ниже, T можно использовать, чтобы уменьшить ошибку аппроксимации $L(B, T)$. Следовательно, параметр T алгоритма AdaBoost позволяет управлять компромиссом между смещением и сложностью.

Чтобы продемонстрировать, как выразительная способность $L(B, T)$ увеличивается с ростом T, рассмотрим простой пример, когда $\mathcal{X} = \mathbb{R}$, а базовым классом является класс решающих пней

$$\mathcal{H}_{\mathrm{DS1}} = \{x \mapsto \operatorname{sign}(x - \theta) \cdot b \colon \theta \in \mathbb{R}, b \in \{\pm 1\}\}.$$

Отметим, что в этом одномерном случае $\mathcal{H}_{\mathrm{DS1}}$ эквивалентен (неоднородным) полупространствам $\mathbb{R}$.

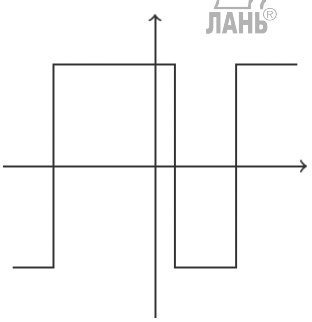
Пусть теперь $\mathcal{H}$ – более сложный класс (по сравнению с полупространствами на прямой) кусочно-постоянных функций. Обозначим g_r кусочно-постоянную функцию не более чем с r участками постоянства, т. е. такую, что существуют пороги $-\infty = \theta_0 < \theta_1 < \theta_2 < \cdots < \theta_r = \infty$, для которых

$$g_r(x) = \sum_{i=1}^{r} \alpha_i \mathbb{1}_{[x \in (\theta_{i-1}, \theta_i]]} \quad \forall i, \quad \alpha_i \in \{\pm 1\}.$$

Обозначим $\mathcal{G}_r$ класс всех таких кусочно-постоянных классификаторов, состоящих не более чем из r участков постоянства.

Далее мы покажем, что $\mathcal{G}_T \subseteq L(\mathcal{H}_{\mathrm{DS1}}, T)$, т. е. класс полупространств над T решающими пнями дает все кусочно-постоянные классификаторы, содержащие не более T участков постоянства.

Действительно, без ограничения общности рассмотрим любое $g \in \mathcal{G}_T$ с $\alpha_t = (-1)^t$. Это означает, что если x принадлежит полуинтервалу $(\theta_{t-1}, \theta_t]$, то $g(x) = (-1)^t$. Например:



Функция

$$h(x) = \text{sign}\left(\sum_{t=1}^{T} w_t \text{sign}(x - \theta_{t-1})\right), \tag{10.5}$$

где $w_1 = 0{,}5$ и $w_t = (-1)^t$ для $t > 1$, принадлежит $L(\mathcal{H}_{\text{DS1}}, T)$ и равна g (см. упражнение 10.2).

Из этого примера вытекает, что $L(\mathcal{H}_{\text{DS1}}, T)$ может разбивать любое множество $T + 1$ образцов в $\mathbb{R}$, поэтому VC-размерность $L(\mathcal{H}_{\text{DS1}}, T)$ не меньше $T + 1$. Следовательно, T – параметр, который может управлять компромиссом между смещением и сложностью: увеличение T дает более выразительный класс гипотез, но может привести к увеличению ошибки оценивания. В следующем подразделе мы формально ограничим сверху VC-размерность $L(B, T)$ для любого базового класса B.

10.3.1. VC-размерность L(B, T)

Следующая лемма говорит, что VC-размерность $L(B, T)$ ограничена сверху величиной $\tilde{O}(\text{VCdim}(B)T)$ (нотация $\tilde{O}$ означает, что логарифмические множители игнорируются).

Лемма 10.3. *Пусть B – базовый класс, и $L(B, T)$ определено, как в (10.4). Предположим, что T и $\text{VCdim}(B)$ не меньше 3. Тогда*

$$\text{VCdim}(L(B, T)) \leq T(\text{VCdim}(B) + 1)(3\log(T(\text{VCdim}(B) + 1)) + 2).$$

Доказательство. Обозначим $d = \text{VCdim}(B)$. Пусть $C = \{x_1, \ldots, x_m\}$ – множество, разбиваемое классом $L(B, T)$. Любая пометка C гипотезой $h \in L(B, T)$ получается, если сначала выбрать $h_1, \ldots, h_T \in B$, а затем применить полупространственную гипотезу к вектору $(h_1(x), \ldots, h_T(x))$. По лемме Зауэра, существует не более $(em/d)^d$ различных дихотомий (т. е. пометок), индуцированных B на C. Следовательно, мы должны выбрать T из максимум $(em/d)^d$ различных гипотез. Существует не более $(em/d)^{dT}$ способов сделать это. Далее к каждому такому выбору применим линейный предиктор, что дает не более $(em/T)^T$ дихотомий. Следовательно, общее число дихотомий, которые мы можем построить, ограничено сверху величиной

$$(em/d)^{dT}(em/T)^T \leq m^{(d+1)T},$$

где мы воспользовались предположением, что d и T не меньше 3. Поскольку мы предполагаем, что C разбито, эта величина должна быть равна как минимум 2^m, откуда получаем:

$$2m \leq m^{(d+1)T}.$$

Следовательно,

$$m \leq \log(m)\frac{(d+1)T}{\log(2)}.$$

Лемма А.1 из приложения А говорит, что необходимым условием для выполнения этого неравенства является

$$m \leq 2\frac{(d+1)T}{\log(2)}\log\frac{(d+1)T}{\log(2)} \leq (d+1)T(3\log((d+1)T)+2),$$

что и требовалось доказать. □

В упражнении 10.4 мы покажем, что для некоторых базовых классов B имеет место также неравенство $\mathrm{VCdim}(L(B, T)) \geq \Omega(\mathrm{VCdim}(B)\ T)$.

10.4. Применение AdaBoost для распознавания лиц

Обратимся теперь к базовой гипотезе, предложенной Виолой и Джонсом для задачи о распознавании лиц. В этой задачи пространством образцов является множество изображений, представленных матрицами полутоновых пикселей. Для определенности будем рассматривать изображения размера 24×24 пикселя, так что пространство образцов – это множество вещественных матриц размера 24×24. Наша цель – обучить классификатор $h : \mathcal{X} \to \{\pm 1\}$, который, получив на входе изображение, сообщает, есть на нем человеческое лицо или нет.

Любая гипотеза из базового класса имеет вид $h(x) = f(g(x))$, где f – гипотеза решающего пня, а $g : \mathbb{R}^{24\times 24} \to \mathbb{R}$ – функция, отображающая изображение в скаляр. Каждая функция g параметризована:

- осепараллельным прямоугольником R. Так как размер изображения 24×24, то существует не более 24^4 осепараллельных прямоугольников;
- типом $t \in \{A, B, C, D\}$. Каждому типу соответствует маска, как показано на рис. 10.1.

Чтобы вычислить g, мы растягиваем маску t на прямоугольник R, а затем вычисляем сумму пикселей (т. е. сумму их полутоновых значений), оказавшихся внутри внешних прямоугольников и вычитаем ее из суммы пикселей во внутренних прямоугольниках.

Поскольку количество таких функций g не превосходит $24^4 \cdot 4$, мы можем реализовать слабого обучаемого для базового класса гипотез, вычислив сначала все возможные результаты g на каждом изображении, а затем применив слабого обучаемого для класса решающих пней, который был описан в предыдущем разделе. Первый шаг можно выполнить очень эффективно, если произвести предобработку, в ходе которой вычисляется интегральное изображение для каждого изображения в обучающем наборе. Детали см. в упражнении 10.5.

На рис. 10.2 изображены первые два признака, выбранные алгоритмом AdaBoost, примененным к базовым признакам Виолы и Джонса.

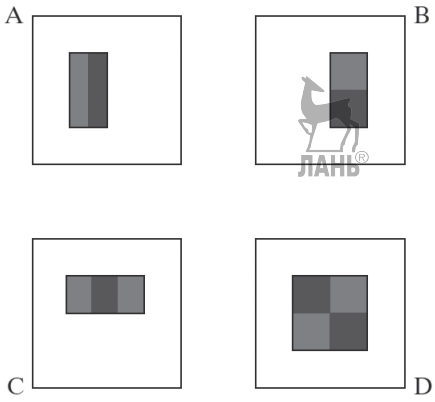


Рис. 10.1. Четыре типа функций g, используемые в базовых гипотезах для распознавания лиц. Значение g для типов A и B равно разности между суммой пикселей внутри двух прямоугольных областей. Форма и размер этих областей одинаковы, а друг к другу они примыкают по горизонтали или по вертикали. Для типа C значение g равно сумме пикселей в двух внешних прямоугольниках минус сумма пикселей в среднем прямоугольнике. Для типа D мы вычисляем разность сумм пикселей в диагональных парах прямоугольников

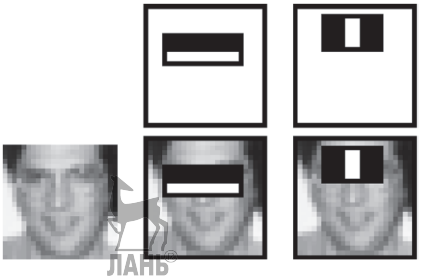


Рис. 10.2. Первый и второй признаки, выбранные алгоритмом AdaBoost, реализованным Виолой и Джонсом. Оба признака показаны в верхнем ряду, а затем наложены на типичное обучающее лицо в нижнем ряду. Первый признак измеряет разность яркости в области глаз и в области верхней части щек. Он основан на том наблюдении, что область глаз часто темнее щек. Второй признак сравнивает яркости в области глаз и в области переносицы

10.5. Резюме

Метод усиления повышает верность слабых обучаемых. В этой главе мы описали алгоритм AdaBoost. Мы показали, что после T итераций этот алгоритм возвращает гипотезу из класса $L(B, T)$, полученную композицией линейного классификатора с T гипотезами из базового класса B. Мы продемонстрировали, что параметр T управляет компромиссом между ошибками аппроксимации и оценивания. В следующей главе мы изучим, как настраивать такие параметры, как T, исходя из данных.

10.6. Библиографические сведения

Как уже отмечалось, идея усиления возникла из теоретического вопроса о том, можно ли «усилить» эффективного слабого обучаемого, превратив его в эффективного сильного обучаемого (Kearns & Valiant, 1988), решенного в работе Schapire (1990). Алгоритм AdaBoost был предложен в работе Freund and Schapire (1995).

Усиление можно рассматривать с разных точек зрения. Чисто теоретически алгоритм AdaBoost можно интерпретировать как отрицательный результат: если сильное обучение класса гипотез – вычислительно трудная задача, то таковой же является и задача слабого обучения. Этот результат может быть полезен, чтобы доказать трудность агностического PAC-обучения класса B, исходя из трудности PAC-обучения какого-то другого класса $\mathcal{H}$ при условии, что $\mathcal{H}$ допускает слабое обучение с помощью B. Например, в работе Klivans and Sherstov (2006) показано, что PAC-обучить класс пересечений полупространств трудно (даже в реализуемом случае). Этим результатом можно воспользоваться, чтобы показать, что агностическое PAC-обучение одного полупространства – тоже трудная задача (Shalev-Shwartz, Shamir & Sridharan, 2010). Идея в том, чтобы показать, то агностический PAC-обучаемый для одного полупространства может породить слабого обучаемого для класса пересечений полупространств, а т. к. слабого обучаемого можно усилить, то мы получим сильного обучаемого для класса пересечений.

AdaBoost доказывает также эквивалентность существования слабого обучаемого и разделимости данных с помощью линейного классификатора, опирающегося на предсказания слабых гипотез. Этот результат тесно связан с минимаксной теоремой фон Неймана (von Neumann, 1928) – фундаментальным результатом теории игр.

AdaBoost также связан с понятием зазора, которое мы будем изучать в главе 15. Кроме того, его можно рассматривать как алгоритм опережающего жадного выбора, это тема главы 25. В недавно вышедшей книге Schapire and Freund (2012) усиление рассматривается со всех точек зрения, открывая читателю доступ к богатству исследований в этой области.

10.7. Упражнения

10.1. Усиление уверенности. Пусть A – алгоритм, гарантирующий следующее условие: существуют константа $\delta_0 \in (0, 1)$ и функция $m_{\mathcal{H}} : (0, 1) \to \mathbb{N}$ такие, что для любого $\epsilon \in (0, 1)$, если $m \geq m_{\mathcal{H}}(\epsilon)$, то для любого распределения $\mathcal{D}$ с вероятностью не менее $1 - \delta_0$ имеет место неравенство $L_{\mathcal{D}}(A(S)) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$.

Предложите процедуру, которая, опираясь на A, обучает $\mathcal{H}$, следуя обычной модели агностического PAC-обучения, и имеет выборочную сложность

$$m_{\mathcal{H}}(\epsilon, \delta) \leq k\, m_{\mathcal{H}}(\epsilon) + \left\lceil \frac{2\log(4k/\delta)}{\epsilon^2} \right\rceil,$$

где

$$k = \lceil \log(\delta) / \log(\delta_0) \rceil.$$

Указание. Разделите данные на $k + 1$ порций, так чтобы в каждой из первых k порций было $m_{\mathcal{H}}(\epsilon)$ примеров. Обучите A на первых k порциях. Докажите, что вероятность того, что для каждой из этих порций $L_{\mathcal{D}}(A(S)) > \min_{h \in \mathcal{H}} L\mathcal{D}(h) + \epsilon$, составляет не более $\delta_0^k \le \delta/2$. Наконец, воспользуйтесь последней порцией, чтобы выбрать одну из k гипотез, сгенерированных A по первым k порциям (примените следствие 4.6).

10.2. Докажите, что функция h из формулы (10.5) совпадает с кусочно-постоянной функцией, определенной с такими же порогами, что и h.

10.3. Мы неформально отметили, что в алгоритме AdaBoost механизм весов используется для того, чтобы «заставить» слабого обучаемого уделять больше внимания проблематичным примерам на следующей итерации. В этом упражнении мы приведем строгое обоснование этого утверждения.

Покажите, что ошибка h_t относительно распределения $\mathbf{D}^{(t+1)}$ равна в точности $1/2$. То есть покажите, что для любого $t \in [T]$

$$\sum_{i=1}^{m} D_i^{(t+1)} \mathbb{1}_{[y_i \neq h_t(\mathbf{x}_i)]} = 1/2.$$

10.4. В этом упражнении мы обсудим VC-размерность классов вида $L(B, T)$. Мы доказали существование верхней границы $O(dT \log(dT))$, где $d = \text{VCdim}(B)$. А сейчас мы хотим доказать существование очень близкой нижней границы. Однако это справедливо не для всех классов B.

1. Отметим, что для любого класса B и любого числа $T \ge 1$ имеет место неравенство $\text{VCdim}(B) \le \text{VCdim}(L(B, T))$. Найдите класс B, для которого $\text{VCdim}(B) = \text{VCdim}(L(B, T))$ при любом $T \ge 1$.
   Указание. Возьмите в качестве $\mathcal{X}$ конечное множество.
2. Пусть B_d – класс решающих пней над $\mathbb{R}^d$. Докажите, что $\log(d) \le \text{VCdim}(B_d) \le 5 + 2\log(d)$.
   Указания:
   - для доказательства верхней границы воспользуйтесь упражнением 10.11;
   - для доказательства нижней границы возьмем $d = 2^k$. Пусть A – матрица размера $k \times d$, столбцами которой являются все d двоичных векторов из $\{\pm 1\}^k$. Строки A образуют множество k векторов в $\mathbb{R}^d$. Покажите, что это множество разбивается классом решающих пней над $\mathbb{R}^d$.
3. Пусть $T \ge 1$ – произвольное целое число. Докажите, что $\text{VCdim}(L(B_d, T)) \ge 0{,}5T\log(d)$.
   Указание. Постройте множество, содержащее $(T/2)k$ примеров, взяв строки матрицы A из предыдущего упражнения и строки матриц $2A, 3A, 4A, \ldots, (T/2)A$. Покажите, что результирующее множество разбивается классом $L(B_d, T)$.

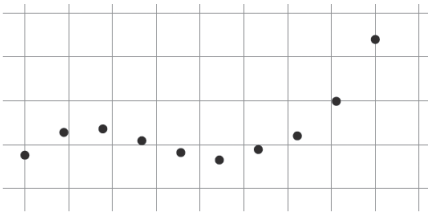
10.5. Эффективное вычисление признаков Виолы–Джонса с помощью интегрального изображения. Пусть A – матрица 24×24, представляющая изображение. Интегральным изображением A, обозначаемым $I(A)$, называется матрица B такая, что $B_{i,j} = \sum_{i' \le i, j' \le j} A_{i,j}$.

- Покажите, что $I(A)$ можно вычислить за время, линейно зависящее от размера A.
- Покажите, как можно вычислить признак Виолы–Джонса по $I(A)$ за постоянное время (это значит, что время работы не зависит от размера прямоугольника, определяющего признак).

ВЫБОР И КОНТРОЛЬ МОДЕЛИ

В предыдущей главе мы описали алгоритм AdaBoost и показали, что его параметр T управляет компромиссом между смещением и сложностью. Но как выбрать значение T на практике? Вообще, приступая к некоторой практической проблеме, мы обычно имеем на выбор несколько алгоритмов, которые могут дать хорошее решение, и у каждого есть набор параметров. Как выбрать наилучший алгоритм для данной проблемы? И как задать его параметры? Эту задачу часто называют *выбором модели*.

Для иллюстрации рассмотрим проблему обучения функции одномерной регрессии, $h : \mathbb{R} \to \mathbb{R}$. Предположим, что у нас имеется обучающий набор, показанный на рисунке ниже.



Мы можем рассмотреть аппроксимацию данных полиномом, как описано в главе 9. Однако нам неизвестно, при какой степени полинома d получатся наилучшие результаты: если степень мала, то данные плохо лягут на кривую (будет велика ошибка аппроксимации), а если велика, то возникает риск переобучения (большая ошибка оценивания). Ниже показан результат аппроксимации полиномами степени 2, 3 и 10. Легко видеть, что эмпирический риск уменьшается с увеличением степени. Но, глядя на графики, мы приходим к интуитивному выводу, что степень 3 лучше, чем 10. Значит, одного эмпирического риска недостаточно для выбора модели.

В этой главе мы представим два подхода к выбору модели. Первый основан на парадигме структурной минимизации риска (SRM), которую мы описали и проанализировали в разделе 7.2. SRM особенно полезна, когда алгоритм обучения зависит от параметра, управляющего компромиссом между смещением и сложностью (как, например, степень аппроксимирующего полинома в примере выше или параметр T в алгоритме AdaBoost). Второй подход опирается на концепцию *контроля* (validation). Основная идея состоит в том, чтобы разбить обучающий набор на две части: одна используется для обучения моделей-кандидатов, а вторая – чтобы выбрать ту, которая дает наилучшие результаты.

В задаче выбора модели мы пытаемся найти баланс между ошибками аппроксимации и оценивания. Вообще, если алгоритм обучения не может найти предиктор с малым риском, то важно понять, в чем причина: в переобучении или в недообучении. В разделе 11.3 мы обсудим, как можно ответить на этот вопрос.

11.1. Выбор модели с помощью SRM

Парадигма SRM была описана и проанализирована в разделе 7.2. Здесь мы покажем, как ее можно использовать для выбора компромисса между смещением и сложностью, не отдавая заранее предпочтения никакой конкретной гипотезе. Рассмотрим счетную последовательность классов гипотез $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \ldots$. Например, в проблеме полиномиальной регрессии $\mathcal{H}_d$ может быть множеством полиномов степени не больше d. Другой пример: $\mathcal{H}_d$ – класс $L(B, d)$ в алгоритме AdaBoost, описанном в предыдущей главе.

Предположим, что для каждого d класс $\mathcal{H}_d$ обладает свойством равномерной сходимости (см. определение 4.3 в главе 4) с функцией выборочной сложности вида

$$m^{\mathrm{UC}}_{\mathcal{H}_d}(\epsilon, \delta) \leq \frac{g(d)\log(1/\delta)}{\epsilon^2}, \tag{11.1}$$

где $g : \mathbb{N} \to \mathbb{R}$ – монотонно возрастающая функция. Например, в проблемах бинарной классификации мы можем в качестве $g(d)$ взять произведение VC-размерности класса $\mathcal{H}_d$ на универсальную константу (появляющуюся в фундаментальной теореме обучения, см. теорему 6.8). Для классов $L(B, d)$ из алгоритма AdaBoost функция g будет просто расти вместе с d.

Напомним, что правило SRM следует подходу «минимизации границы», и в нашем случае граница такова: с вероятностью не менее $1 - \delta$ для любых $d \in \mathbb{N}$ и $h \in \mathcal{H}_d$

$$L_{\mathcal{D}}(h) \leq L_S(h) + \sqrt{\frac{g(d)\log(1/\delta) + 2\log(d) + \log(\pi^2/6))}{m}}. \tag{11.2}$$

Эта граница, вытекающая непосредственно из теоремы 7.4, показывает, что для любых d и $h \in \mathcal{H}_d$ истинный риск ограничен двумя членами: эмпирическим риском $L_S(h)$ и сложностью, зависящей от d. Правило SRM ищет такие d и $h \in \mathcal{H}_d$, которые минимизируют правую часть неравенства (11.2).

Вернемся к примеру полиномиальной регрессии; хотя эмпирический риск полинома десятой степени меньше, чем у полинома третьей степени, мы все-таки

предпочитаем последний, т. к. его сложность (отразившаяся в значении функции $g(d)$) гораздо меньше.

Хотя подход на основе SRM полезен во многих ситуациях, на практике верхняя граница в формуле (11.2) часто оказывается излишне пессимистичной. В следующем разделе мы опишем более практический подход.

11.2. Контроль

Нам часто хотелось бы получить более точную оценку истинного риска предиктора, возвращенного алгоритмом обучения. Выведенные до сих пор границы ошибки оценивания говорили нам, что для *всех* гипотез из данного класса истинный риск не слишком сильно отличается от эмпирического. Однако эти границы могут быть неточными и пессимистичными, поскольку справедливы для всех гипотез и всех возможных распределений данных. Более точную оценку истинного риска можно получить, если использовать часть обучающего набора как контрольный набор, на котором оценивается успех выходного предиктора. Эта процедура называется *контролем*.

Естественно, более точная оценка истинного риска полезна при выборе модели; мы поговорим об этом в разделе 11.2.2.

11.2.1. Зарезервированный набор

Самый простой способ оценить истинную ошибку предиктора h – взять дополнительную выборку примеров, независимую от обучающего набора, и использовать эмпирическую ошибку на этом контрольном наборе в качестве оценки. Формально, пусть $V = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{m_v}, y_{m_v})$ – новый набор m_v примеров, выбранных из распределения $\mathcal{D}$ (независимо от m примеров, вошедших в обучающий набор S). Из неравенства Хёфдинга (лемма 4.5) вытекает следующая

Теорема 11.1. *Пусть h – некоторый предиктор, а функция потерь принимает значения из отрезка* $[0, 1]$. *Тогда для любого $\delta \in (0, 1)$ с вероятностью не менее $1 - \delta$ для случайного контрольного набора V размера m_v имеет место неравенство*

$$\left|L_V(h) - L_{\mathcal{D}}(h)\right| \leq \sqrt{\frac{\log(2/\delta)}{2m_v}}.$$

Граница в теореме 11.1 не зависит ни от алгоритма, ни от обучающего набора, использованного для построения h, и при этом точнее встречавшихся ранее границ. Причина в том, что эта граница выражена в терминах оценки на отдельном контрольном наборе, независимом от способа генерации h. Чтобы проиллюстрировать эту мысль, предположим, что h была получена применением ERM-предиктора относительно класса гипотез VC-размерности d к обучающему набору из m примеров. Тогда из фундаментальной теоремы обучения (теорема 6.8) мы получаем границу

$$L_{\mathcal{D}}(h) \leq L_S(h) + \sqrt{C\frac{d + \log(1/\delta)}{m}},$$

где C – константа из теоремы 6.8. С другой стороны, теорема 11.1 дает границу

$$L_{\mathcal{D}}(h) \leq L_V(h) + \sqrt{\frac{\log(2/\delta)}{2m_v}}.$$

Поэтому, если взять m_v того же порядка, что и m, то мы получим более точную оценку, а во сколько именно раз, зависит от VC-размерности. Но за это приходится расплачиваться дополнительными примерами сверх тех, что используются для обучения.

Выборка обучающего набора, а затем независимого от него контрольного эквивалентна случайному разбиению множества примеров на две части: для обучения и для контроля. Поэтому контрольный набор часто называют *зарезервированным*.

11.2.2. Контроль при выборе модели

Контроль можно естественно использовать для выбора модели следующим образом. Сначала мы обучаем различные алгоритмы (или один и тот же алгоритм с разными параметрами) на заданном обучающем наборе. Пусть $\mathcal{H} = \{h_1, \ldots, h_r\}$ – множество выходных предикторов, порожденных разными алгоритмами. Например, при обучении полиномиальных функций регрессии h_r будет соответствовать полиномам степени r. Теперь, чтобы выбрать из $\mathcal{H}$ один предиктор, мы воспользуемся отдельным контрольным набором и возьмем предиктор с минимальной ошибкой на этом наборе. Иными словами, мы применяем правило $\mathrm{ERM}_{\mathcal{H}}$ к контрольному набору.

Этот процесс очень похож на обучение конечного класса гипотез. Единственное различие состоит в том, что $\mathcal{H}$ не фиксируется заранее, а зависит от обучающего набора. Но поскольку контрольный набор не зависит от обучающего, он не зависит также и от $\mathcal{H}$, поэтому техника, примененная при выводе границ для конечных классов гипотез, применима и здесь. В частности, теорема 11.1 в сочетании с леммой о границе объединения дает следующий результат.

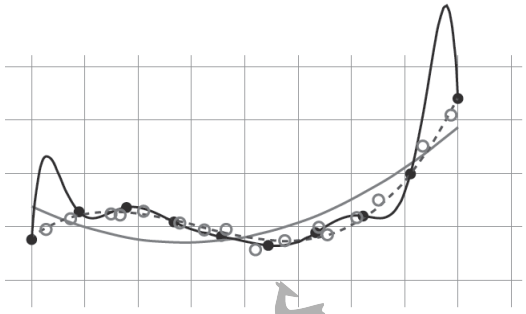
Теорема 11.2. *Пусть $\mathcal{H} = \{h_1, \ldots, h_r\}$ – произвольное множество предикторов, и предположим, что функция потерь принимает значения из отрезка* $[0, 1]$. *Предположим, что контрольный набор V размера m_v выбирается независимо от $\mathcal{H}$. Тогда с вероятностью не менее* $1 - \delta$ *для случайного набора V имеет место неравенство*

$$\forall h \in \mathcal{H}, \ \left|L_{\mathcal{D}}(h) - L_V(h)\right| \leq \sqrt{\frac{\log(2|\mathcal{H}|/\delta)}{2m_v}}.$$

Эта теорема утверждает, что ошибка на контрольном наборе аппроксимирует истинную ошибку, если только класс $\mathcal{H}$ не слишком велик. Но если мы будем пробовать слишком много методов (так что $|\mathcal{H}|$ велико по сравнению с размером контрольного набора), то рискуем получить переобучение.

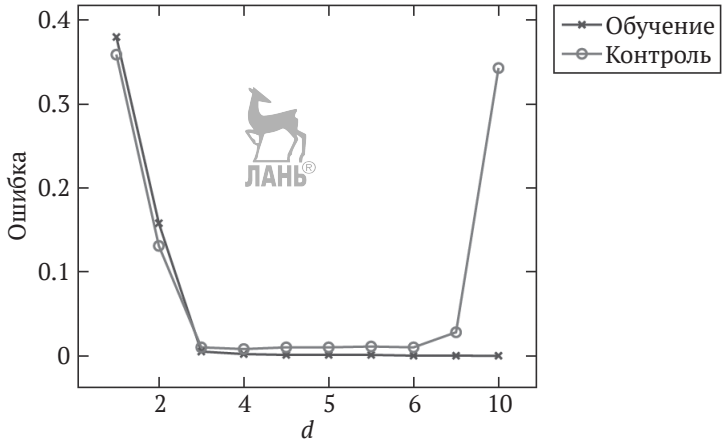
Чтобы проиллюстрировать полезность контроля для выбора модели, снова рассмотрим пример аппроксимации полиномом от одной переменной, приведенный в начале этой главы. На рисунке ниже изображен тот же самый обучающий набор и ERM-полиномы степени 2, 3 и 10, но на этот раз мы добавили еще

и контрольный набор (обозначенный незаполненными окружностями). Полином степени 10 дает минимальную ошибку обучения, а полином степени 3 – минимальную ошибку на контрольном наборе, поэтому он и выбран в качестве лучшей модели.



11.2.3. Кривая выбора модели

Кривая выбора модели показывает ошибки обучения и контроля в виде функции от сложности рассматриваемой модели. Например, для проблемы полиномиальной аппроксимации кривая выглядит так:



Можно показать, что ошибка обучения монотонно убывает при увеличении степени полинома (в нашем случае это и есть сложность модели). С другой стороны, ошибка контроля сначала убывает, а затем начинает возрастать, что свидетельствует о наступлении переобучения.

Рисование таких кривых помогает понять, правильно ли мы обследуем пространство параметров. Часто необходимо настроить не один, а несколько параметров, и количество возможных значений каждого параметра велико. Например, в главе 13 мы описываем концепцию *регуляризации*, в которой параметром алгоритма обучения является вещественное число. В таких случаях следует начинать с грубой сетки значений параметров и рисовать соответствующие кривые

выбора модели. На основе этой кривой мы подбираемся к правильному режиму обследования и дальше используем для поиска более мелкую сетку. Например, в случае полиномиальной аппроксимации, если начать поиск с множества значений {1, 10, 20} и не прибегать к более мелкой сетке, основанной на получающейся кривой, то мы придем к совершенно неудовлетворительной модели.

11.2.4. k-групповая перекрестная проверка

Описанная выше процедура предполагает, что данных в изобилии и выбрать отдельный контрольный набор не составляет труда. Но в некоторых приложениях данных едва хватает, и мы не хотим «транжирить» дефицитные данные на контроль. Техника *k*-групповой перекрестной проверки позволяет получить точную оценку истинной ошибки, не тратя слишком много данных.

Идея в том, чтобы разбить исходный обучающий набор на k подмножеств (групп) размера m/k (для простоты будем считать, что m/k – целое число). Для каждой группы алгоритм обучается на объединении остальных групп, а затем ошибка оценивается на этой группе. В качестве оценки истинной ошибки принимается среднее всех полученных ошибок. Частный случай, когда $k = m$, где m – количество примеров, называется «*исключением по одному*» (leave-one-out – LOO).

k-групповая перекрестная проверка часто применяется для выбора модели (или настройки параметров). После того как оптимальные параметры выбраны, алгоритм заново обучается на всем обучающем наборе с этими параметрами. Ниже приведен псевдокод *k*-групповой перекрестной проверки. Процедура получает на входе обучающий набор S, множество возможных значений параметра Θ, целое число k, равное количеству групп, и алгоритм обучения A, который принимает обучающий набор и параметр $\theta \in \Theta$. На выходе получается наилучший параметр, а также гипотеза, обученная с этим параметром на всем обучающем наборе.

Применение *k*-групповой перекрестной проверки к выбору модели

вход:
  обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
  множество значений параметра Θ
  алгоритм обучения A
  целое число k
разбить S на $S_1, S_2, \ldots, S_k$
foreach $\theta \in \Theta$
  for $i = 1 \ldots k$
    $h_{i,\theta} = A(S \backslash S_i; \theta)$
  $\text{error}(\theta) = (1/k)\sum_{i=1}^{k} L_{S_i}(h_{i,\theta})$
выход
 $\theta^* = \text{argmin}_\theta[\text{error}(\theta)]$
 $h_{\theta^*} = A(S; \theta^*)$

На практике метод перекрестной проверки часто дает очень хорошие результаты. Но иногда он может потерпеть неудачу, как показывает искусственный пример, приведенный в упражнении 11.1. Строгое понимание точного поведения перекрестной проверки до сих пор остается нерешенной проблемой. В работе Роджерса и Вагнера (Rogers & Wagner, 1978) показано, что для k локальных правил (например, метода k ближайших соседей, см. главу 19) процедура перекрестной проверки дает очень хорошую оценку истинной ошибки. В других работах показано, что перекрестная проверка работает для устойчивых алгоритмов (вопрос устойчивости и ее связи с обучаемостью рассматривается в главе 13).

11.2.5. Обучение–контроль–тестирование

В большинстве практических приложений мы разбиваем множество доступных примеров на три части. Первая часть используется для обучения алгоритма, вторая – в качестве контрольного набора при выборе модели. Выбрав наилучшую модель, мы проверяем качество выходного предиктора на третьем наборе, который часто называют тестовым. Полученное в результате число служит оценкой истинной ошибки обученного предиктора.

11.3. Что делать, если обучить не удается

Рассмотрим следующую ситуацию. Дана задача обучения, и вы пытаетесь решить ее, выбрав класс гипотез и алгоритм обучения с параметрами. Вы настроили параметры с помощью контрольного набора и протестировали обученный предиктор на тестовом наборе. Увы, результаты оказались неудовлетворительными. Кто виноват и что делать дальше?

Есть много элементов, которые можно «поправить». Основные предложения перечислены ниже:

- взять выборку побольше;
- изменить класс гипотез:
  - увеличить его;
  - уменьшить его;
  - полностью заменить;
  - изменить параметры;
- взять другое представление признаков данных;
- изменить алгоритм оптимизации, используемый в процессе применения правила обучения.

Чтобы найти лучшее лекарство, важно, прежде всего, понять причину плохого качества. Напомним, что в главе 5 мы разложили истинную ошибку обученного предиктора на ошибку аппроксимации и оценивания. Ошибка аппроксимации определена как $L_{\mathcal{D}}(h^*)$ для некоторой гипотезы $h^* \in \operatorname{argmin}_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$, а ошибка оценивания – как $L_{\mathcal{D}}(h_S) - L_{\mathcal{D}}(h^*)$, где h_S – обученный предиктор (на обучающем наборе S).

Ошибка аппроксимации класса не зависит ни от размера выборки, ни от используемого алгоритма. Она зависит только от распределения $\mathcal{D}$ и от класса

гипотез $\mathcal{H}$. Поэтому, если ошибка аппроксимации велика, увеличивать размер обучающего набора бесполезно и уменьшать класс гипотез тоже не имеет смысла. Помочь может увеличение класса гипотез или его полная замена (если у нас есть в запасе альтернативное априорное знание в виде другого класса гипотез). Можно также подумать о применении того же класса гипотез, но с другим представлением данных (см. главу 25).

Ошибка оценивания класса зависит от размера выборки. Поэтому, если ошибка оценивания велика, то стоит попытаться получить дополнительные обучающие примеры. Можно также попробовать уменьшить класс гипотез. Но увеличивать класс гипотез в этом случае бессмысленно.

Разложение ошибки с помощью контроля

Мы видим, что понимание того, чем вызвана проблема – ошибкой аппроксимации или ошибкой оценивания, – очень полезно для нахождения подходящего лекарства. В предыдущем разделе мы видели, как оценить $L_{\mathcal{D}}(h_S)$ с помощью эмпирического риска на контрольном наборе. Оценить же ошибку аппроксимации класса труднее. Вместо этого мы разложим ошибку по-другому, так чтобы ее можно было оценить, зная обучающий и контрольный набор.

$$L(h_S) = (L_{\mathcal{D}}(h_S) - L_V(h_S)) + (L_V(h_S) - L_S(h_S)) + L_S(h_S).$$

Первый член, $(L_{\mathcal{D}}(h_S) - L_V(h_S))$, можно очень точно ограничить сверху, воспользовавшись теоремой 11.1. Интуитивно понятно, что если второй член, $(L_V(h_S) - L_S(h_S))$, велик, то мы говорим, что алгоритм страдает от «переобучения», а если велик эмпирический риск, $L_S(h_S)$, то алгоритм страдает от «недообучения». Отметим, что оба эти члена являются недостаточно хорошими оценками ошибок оценивания и аппроксимации. Чтобы убедиться в этом, рассмотрим случай, когда $\mathcal{H}$ – класс VC-размерности d, а $\mathcal{D}$ – такое распределение, что ошибка аппроксимации $\mathcal{H}$ относительно $\mathcal{D}$ равна 1/4. При условии, что размер обучающего набора меньше d, мы будем иметь $L_S(h_S) = 0$ для любой ERM-гипотезы. Следовательно, риск обучения $L_S(h_S)$ и ошибка аппроксимации $L_{\mathcal{D}}(h^*)$ могут сильно различаться. Тем не менее, как мы покажем ниже, значения $L_S(h_S)$ и $(L_V(h_S) - L_S(h_S))$ все же дают полезную информацию.

Рассмотрим сначала случай, когда $L_S(h_S)$ велико. Мы можем написать

$$L_S(h_S) = (L_S(h_S) - L_S(h^*)) + (L_S(h^*) - L_{\mathcal{D}}(h^*)) + L_{\mathcal{D}}(h^*).$$

Когда h_S является $\mathrm{ERM}_{\mathcal{H}}$-гипотезой, $L_S(h_S) - L_S(h^*) \leq 0$. Кроме того, поскольку h^* не зависит от S, член $L_S(h^*) - L_{\mathcal{D}}(h^*)$ можно ограничить сверху очень точно (как в теореме 11.1). Последний член – это ошибка аппроксимации. Отсюда следует, что если $L_S(h_S)$ велико, то ошибка аппроксимации тоже велика, и лекарство для нашего алгоритма следует подобрать соответствующее (как обсуждалось выше).

Замечание 11.1. Может случиться так, что ошибка аппроксимации нашего класса мала, а значение $L_S(h_S)$ все-таки велико. Например, в нашей реализации ERM может быть ошибка, и алгоритм возвращает гипотезу h_S, не являющуюся ERM-гипотезой. Возможно также, что найти ERM-гипотезу вычислительно трудно, и наш алгоритм применяет некоторую эвристику, чтобы найти приближение

к ней. В таких случаях трудно понять, насколько h_S хороша по сравнению с ERM-гипотезой. Но иногда можно по крайней мере узнать, существуют ли лучшие гипотезы. Так, в следующей главе мы будем изучать выпуклые проблемы обучения, в которых существуют условия оптимальности, которые можно проверить и узнать, сошелся ли алгоритм оптимизации к ERM-решению. В других случаях решение может зависеть от случайной инициализации алгоритма, поэтому можно попробовать выбрать другие начальные значения в надежде получить лучшее решение.

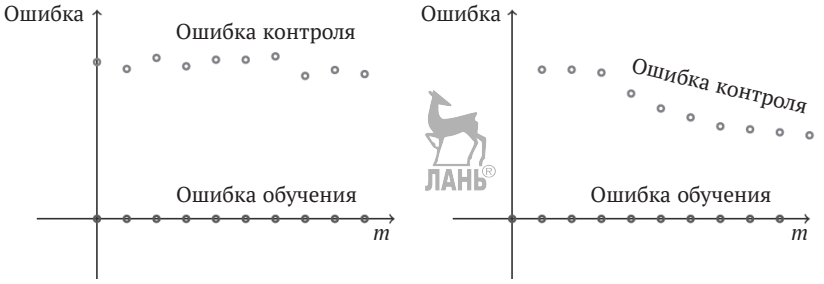


Рис. 11.1. Примеры кривых обучения. Слева: эта кривая соответствует случаю, когда количество примеров всегда меньше VC-размерности класса. Справа: эта кривая соответствует случаю, когда ошибка аппроксимации равна нулю, а количество примеров больше VC-размерности класса

Далее рассмотрим случай, когда $L_S(h_S)$ мало. Как мы уже выяснили, это необязательно означает, что ошибка аппроксимации мала. Действительно, рассмотрим два случая, когда мы пытаемся обучить класс гипотез VC-размерности d, применяя правило ERM. В первом случае имеется обучающий набор, насчитывающий $m < d$ примеров, и ошибка аппроксимации класса велика, а во втором – обучающий набор содержит $m > 2d$ примеров, и ошибка аппроксимации равна нулю. В обоих случаях $L_S(h_S) = 0$. Как же их различить?

Кривые обучения

Один из способов различить эти два случая – нарисовать *кривые обучения*. Чтобы построить такую кривую, мы обучаем алгоритм на префиксных выборках из набора данных увеличивающегося размера. Например, сначала можно обучить алгоритм на первых 10% примеров, потом на 20% и т. д. Для каждой такой префиксной выборки мы вычисляем ошибку обучения (на той выборке, на которой был обучен алгоритм) и ошибку контроля (на предопределенном контрольном наборе). Такие кривые обучения помогают различить два вышеупомянутых случая. В первом случае мы ожидаем, что ошибка контроля будет приблизительно равна 1/2 для всех префиксов, потому что в действительности мы ничему не научились. Во втором случае ошибка контроля сначала будет постоянной, но затем должна уменьшаться (после того, как размер обучающего набора превысит VC-размерность). Оба случая показаны на рис. 11.1.

Вообще говоря, коль скоро ошибка аппроксимации больше нуля, мы ожидаем, что ошибка обучения будет расти вместе с размером выборки, поскольку, чем

больше точек, тем труднее объяснить их все. С другой стороны, ошибка контроля должна уменьшаться с ростом размера выборки. Если VC-размерность конечна, то при стремлении размера выборки к бесконечности ошибки обучения и контроля сходятся к ошибке аппроксимации. Поэтому, экстраполируя кривые обучения и контроля, мы можем попробовать угадать, чему равна ошибка аппроксимации, или по крайней мере получить грубую оценку интервала, в который она попадает.

Но вернемся к проблеме нахождения лучшего средства починить алгоритм. Если мы наблюдаем, что $L_S(h_S)$ мало, а ошибка контроля велика, то в любом случае можем сказать, что размер обучающего набора недостаточен для обучения класса $\mathcal{H}$. Тогда мы можем нарисовать кривую обучения. Если мы видим, что ошибка контроля начинает убывать, то лучше всего будет увеличить количество примеров (если есть возможность добыть дополнительные данные). Другое разумное решение – уменьшить сложность класса гипотез. С другой стороны, если ошибка контроля колеблется вокруг значения 1/2, то нет никаких свидетельств в пользу того, что ошибка аппроксимации $\mathcal{H}$ приемлема. Быть может увеличение размера обучающего набора вообще не поможет. Но все же получение дополнительных данных может принести пользу, т. к. в какой-то момент мы, возможно, увидим, что происходит: начинается уменьшаться ошибка контроля или расти ошибка обучения. Если же дополнительные данные получить сложно, то лучше сначала попробовать уменьшить сложность класса гипотез.

Подводя итоги, мы можем порекомендовать следующие действия.

1. Если обучение включает настройку параметров, то нарисуйте кривую выбора модели, чтобы удостовериться в правильности настройки (см. раздел 11.2.3).
2. Если ошибка обучения чрезмерно велика, попробуйте увеличить класс гипотез, полностью заменить его или изменить представление признаков данных.
3. Если ошибка обучения мала, нарисуйте кривые обучения и попытайтесь, глядя на них, понять, связана проблема с ошибкой оценивания или ошибкой аппроксимации.
4. Если ошибка аппроксимации кажется достаточно малой, попробуйте раздобыть дополнительные данные. Если это невозможно, попробуйте уменьшить сложность класса гипотез.
5. Если ошибка аппроксимации кажется большой, попробуйте полностью изменить класс гипотез или представление признаков данных.

11.4. Резюме

Выбор модели – это задача подбора модели, подходящей для проблемы обучения, решаемая на основе самих данных. Мы показали, как это можно сделать с применением парадигмы обучения SRM, а также с помощью более практичного подхода на основе идеи контроля. Если алгоритм обучения не дает желаемых результатов, то следует провести декомпозицию ошибки алгоритма, воспользовавшись кривыми обучения, чтобы найти наилучшее лекарство.

11.5. Упражнения

11.1. Неудача k-групповой перекрестной проверки. Предположим, что метка выбирается случайным образом в соответствии с распределением вероятностей $\mathbb{P}[y = 1] = \mathbb{P}[y = 0] = 1/2$. Рассмотрим алгоритм, который выводит постоянный предиктор $h(\mathbf{x}) = 1$, если в обучающем наборе нечетное количество единичных меток, а в противном случае выводит постоянный предиктор $h(\mathbf{x}) = 0$. Докажите, что в этом случае разность между оценкой с исключением по одному и истинной ошибкой всегда равна 1/2.

11.2. Пусть $\mathcal{H}_1, \ldots, \mathcal{H}_k$ – k классов гипотез. Предположим, что имеется m независимых и одинаково распределенных обучающих примеров, и мы хотим обучить класс $\mathcal{H} = \bigcup_{i=1}^{k} \mathcal{H}_i$. Рассмотрим два альтернативных подхода:

- обучить $\mathcal{H}$ на m примерах по правилу ERM;
- разделить m примеров на обучающий набор размера $(1 - \alpha)m$ и контрольный набор размера αm для некоторого $\alpha \in (0, 1)$. Затем выбрать модель с применением контроля. Это значит, что сначала мы обучим каждый класс $\mathcal{H}_i$ на $(1 - \alpha)m$ обучающих примерах по правилу ERM относительно $\mathcal{H}_i$, и пусть $\hat{h}_1, \ldots, \hat{h}_k$ – получившиеся гипотезы. Затем применим правило ERM относительно конечного класса $\{\hat{h}_1, \ldots, \hat{h}_k\}$ на αm контрольных примерах.

Опишите, при каких условиях первый метод лучше второго и наоборот.

ЛАНЬ®

Глава 12

ВЫПУКЛЫЕ ПРОБЛЕМЫ ОБУЧЕНИЯ

В этой главе мы познакомимся с *выпуклыми проблемами обучения*. Это важное семейство, главным образом потому, что в него входит большинство проблем, которые поддаются эффективному обучению. Мы уже встречали линейную регрессию с квадратичной потерей и логистическую регрессию – это выпуклые проблемы, и их действительно можно обучить эффективно. Встречались нам и невыпуклые проблемы, например, полупространства с бинарной функцией потерь. Известно, что в нереализуемом случае такие проблемы вычислительно трудны.

Вообще, выпуклой проблемой обучения называется проблема, для которой класс гипотез – выпуклое множество, а функция потерь выпукла для каждого примера. Мы начнем эту главу с некоторых определений, относящихся к выпуклости. Помимо выпуклости, мы определим понятие липшицевости и гладкости – дополнительные свойства функции потерь, обеспечивающие успешное обучение. Далее мы обратимся к определению выпуклых проблем обучения и продемонстрируем необходимость дальнейших ограничений, например, ограниченности и липшицевости или гладкости. Мы определим эти более узкие семейства проблем обучения и сформулируем утверждение о том, что выпуклые проблемы, дополнительно удовлетворяющие условиям гладкости и ограниченности или липшицевости и ограниченности, являются обучаемыми. Это утверждение будет доказано в следующих двух главах, где мы представим две парадигмы обучения, которые позволяют успешно обучить все такие проблемы.

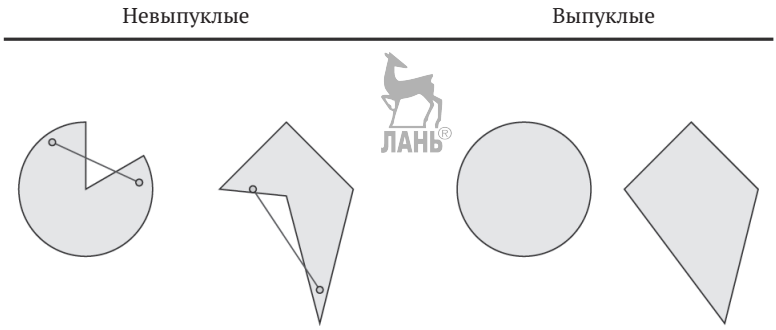
Наконец, в разделе 12.3 мы покажем, как можно подойти к невыпуклым проблемам путем минимизации «суррогатной» выпуклой функции потерь (вместо исходной невыпуклой). Суррогатные выпуклые функции потерь дают эффективные решения, но могут увеличивать риск обученного предиктора.

12.1. Выпуклость, липшицевость и гладкость

12.1.1. Выпуклость

Определение 12.1 (выпуклое множество)**.** Множество C в векторном пространстве называется выпуклым, если для любых двух векторов $\mathbf{u}$, $\mathbf{v}$, принадлежащих C, весь отрезок прямой между $\mathbf{u}$ и $\mathbf{v}$ принадлежит C. Иначе говоря, для любого $\alpha \in [0, 1]$ вектор $\alpha\mathbf{u} + (1 - \alpha)\mathbf{v} \in C$.

На рисунке ниже приведены примеры выпуклых и невыпуклых множеств в $\mathbb{R}^2$. Для невыпуклых множеств можно найти такие две точки, что соединяющий их отрезок не принадлежит целиком множеству.

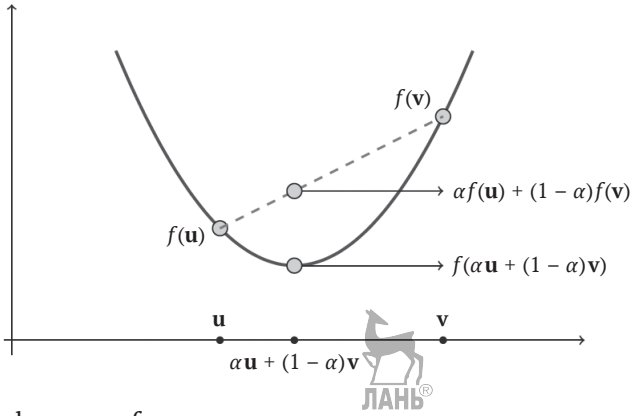


Если $\alpha \in [0, 1]$, то вектор $\alpha\mathbf{u} + (1 - \alpha)\mathbf{v}$ называется *выпуклой комбинацией* векторов $\mathbf{u}$ и $\mathbf{v}$.

Определение 12.2 (выпуклая функция). Пусть C – выпуклое множество. Функция $f : C \to \mathrm{R}$ называется выпуклой, если для любых $\mathbf{u}, \mathbf{v} \in C$ и $\alpha \in [0, 1]$,

$$f(\alpha\mathbf{u} + (1 - \alpha)\mathbf{v}) \le \alpha f(\mathbf{u}) + (1 - \alpha) f(\mathbf{v}).$$

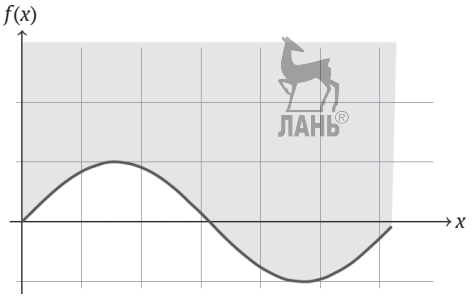
Иными словами, f выпукла, если для любых $\mathbf{u}$ и $\mathbf{v}$ часть графика f между $\mathbf{u}$ и $\mathbf{v}$ лежит выше отрезка, соединяющего $f(\mathbf{u})$ и $f(\mathbf{v})$. На рисунке ниже показан пример выпуклой функции $f : \mathbb{R} \to \mathbb{R}$:



Надграфиком функции f называется множество

$$\mathrm{epigraph(f)} = \{(\mathbf{x}, \beta) : f(\mathbf{x}) \le \beta\}. \tag{12.1}$$

Легко проверить, что функция f выпукла тогда и только тогда, когда ее надграфик – выпуклое множество. На рисунке ниже приведен пример невыпуклой функции $f : \mathbb{R} \to \mathbb{R}$ и ее надграфика.



У выпуклых функций есть важное свойство – всякий локальный минимум одновременно является глобальным. Докажем это. Обозначим $B(\mathbf{u}, r) = \{\mathbf{v} : \|\mathbf{v} - \mathbf{u}\| \le r\}$ шар радиуса r с центром в точке $\mathbf{u}$. Говорят, что $f(\mathbf{u})$ является локальным минимумом f в $\mathbf{u}$, если существует $r > 0$ такое, что для любой точки $\mathbf{v} \in B(\mathbf{u}, r)$ имеет место неравенство $f(\mathbf{v}) \ge f(\mathbf{u})$. Отсюда следует, что для любой $\mathbf{v}$ (не обязательно принадлежащей B) существует достаточно малое $\alpha > 0$ такое, что $\mathbf{u} + \alpha(\mathbf{v} - \mathbf{u}) \in B(\mathbf{u}, r)$ и, следовательно,

$$f(\mathbf{u}) \le f(\mathbf{u} + \alpha(\mathbf{v} - \mathbf{u})). \tag{12.2}$$

Если f выпукла, то имеет также неравенство

$$f(\mathbf{u} + \alpha(\mathbf{v} - \mathbf{u})) = f(\alpha\,\mathbf{v} + (1 - \alpha)\mathbf{u}) \le (1 - \alpha) f(\mathbf{u}) + \alpha f(\mathbf{v}). \tag{12.3}$$

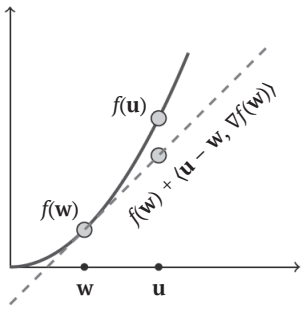
Объединяя обе формулы и перегруппировывая члены, заключаем, что $f(\mathbf{u}) \le f(\mathbf{v})$. Так как это верно для любой точки $\mathbf{v}$, то $f(\mathbf{u})$ является глобальным минимумом f.

Еще одно важное свойство выпуклых функций заключается в том, что касательная к f в любой точке $\mathbf{w}$ всюду расположена ниже f. Если f дифференцируема, то касательная – это линейная функция $l(\mathbf{u}) = f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{u} - \mathbf{w} \rangle$, где $\nabla f(\mathbf{w})$ – градиент f в точке $\mathbf{w}$, т. е. вектор частных производных f: $\nabla f(\mathbf{w}) = \left(\dfrac{\partial f(\mathbf{w})}{\partial w_1}, \ldots, \dfrac{\partial f(\mathbf{w})}{\partial w_d} \right)$.

Таким образом, для дифференцируемых функций

$$\forall \mathbf{u} f(\mathbf{u}) \ge f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{u} - \mathbf{w} \rangle. \tag{12.4}$$

В главе 14 мы обобщим это неравенство на недифференцируемые функции. На рисунке ниже приведена иллюстрация неравенства (12.4).



Для скалярных дифференцируемых функций существует простой критерий выпуклости.

Лемма 12.3. *Пусть f: $\mathbb{R} \to \mathbb{R}$ скалярная дважды дифференцируемая функция, и f', f'' – ее первая и вторая производные соответственно. Тогда следующие утверждения эквивалентны:*

1) *f выпукла;*
2) *f' монотонно не убывает;*
3) *f'' неотрицательна.*

Пример 12.1

- Скалярная функция $f(x) = x^2$ выпукла. Чтобы убедиться в этом, заметим, что $f'(x) = 2x$ и $f''(x) = 2 > 0$.
- Скалярная функция $f(x) = \log(1 + \exp(x))$ выпукла. Действительно, $f'(x) = \exp(x)/(1 + \exp(x)) = 1/(\exp(-x) + 1)$. Эта функция монотонно убывает, потому что экспонента – монотонно возрастающая функция.

Следующее утверждение показывает, что композиция выпуклой скалярной функции с линейной функцией дает выпуклую векторную функцию.

Утверждение 12.4. *Предположим, что f: $\mathbb{R}^d \to \mathbb{R}$ можно записать в виде $f(\mathbf{w}) = g(\langle \mathbf{w}, \mathbf{x} \rangle + y)$ для некоторых $\mathbf{x} \in \mathbb{R}^d$, $y \in \mathbb{R}$ и g: $\mathbb{R} \to \mathbb{R}$. Тогда из выпуклости g следует выпуклость f.*

Доказательство. Пусть $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^d$, и $\alpha \in [0, 1]$. Имеем

$$\begin{aligned} f(\alpha \mathbf{w}_1 + (1 - \alpha)\mathbf{w}_2) &= g(\langle \alpha \mathbf{w}_1 + (1 - \alpha)\mathbf{w}_2, \mathbf{x} \rangle + y) \\ &= g(\alpha \langle \mathbf{w}_1, \mathbf{x} \rangle + (1 - \alpha)\langle \mathbf{w}_2, \mathbf{x} \rangle + y) \\ &= g(\alpha(\langle \mathbf{w}_1, \mathbf{x} \rangle + y) + (1 - \alpha)(\langle \mathbf{w}_2, \mathbf{x} \rangle + y)) \\ &\leq \alpha g(\langle \mathbf{w}_1 \mathbf{x} \rangle + y) + (1 - \alpha) g(\langle \mathbf{w}_2, \mathbf{x} \rangle + y), \end{aligned}$$

где последнее неравенство вытекает из выпуклости g. □

Пример 12.2

- Пусть даны $\mathbf{x} \in \mathbb{R}^d$ и $y \in \mathbb{R}$, и пусть $f : \mathbb{R}^d \to \mathbb{R}$ определена следующим образом: $f(\mathbf{w}) = (\langle \mathbf{w}, \mathbf{x} \rangle - y)^2$. Тогда f является композицией функции $g(a) = a^2$ с линейной функцией, а значит, f выпукла.
- Пусть даны $\mathbf{x} \in \mathbb{R}^d$ и $y \in \{\pm 1\}$, и пусть $f : \mathbb{R}^d \to \mathbb{R}$ определена следующим образом: $f(\mathbf{w}) = \log(1 + \exp(-y\langle \mathbf{w}, \mathbf{x} \rangle))$. Тогда f является композицией функции $g(a) = \log(1 + \exp(a))$ с линейной функцией, а значит, f выпукла.

Наконец, следующее утверждение показывает, что максимум выпуклых функций – выпуклая функция и взвешенная сумма выпуклых функций с неотрицательными весами также выпукла.

Утверждение 12.5. *Для $i = 1, \ldots, r$ пусть $f_i : \mathbb{R}^d \to \mathbb{R}$ – выпуклая функция. Тогда следующие функции из $\mathbb{R}^d$ в $\mathbb{R}$ также выпуклы:*

- $g(x) = \max_{i \in [r]} f_i(x)$;
- $g(x) = \sum_{i=1}^{r} w_i f_i(x)$, *где для всех i $w_i \geq 0$.*

Доказательство. Первое утверждение вытекает из следующей цепочки

$$\begin{aligned} g(\alpha u + (1-\alpha)v) &= \max_i f_i(\alpha u + (1-\alpha)v) \\ &\le \max_i [\alpha f_i(u) + (1-\alpha) f_i(v)] \\ &\le \alpha \max_i f_i(u) + (1-\alpha) \max_i f_i(v) \\ &= \alpha g(u) + (1-\alpha) g(v). \end{aligned}$$

Что касается второго утверждения, то:

$$\begin{aligned} g(\alpha u + (1-\alpha)v) &= \sum_i w_i f_i(\alpha u + (1-\alpha)v) \\ &\le \sum_i w_i [\alpha f_i(u) + (1-\alpha) f_i(v)] \\ &\le \alpha \sum_i w_i f_i(u) + (1-\alpha) \sum_i w_i f_i(v) \\ &= \alpha g(u) + (1-\alpha) g(v). \end{aligned}$$

Пример 12.3. Функция $g(x) = |x|$ выпукла. В самом деле, заметим, что $g(x) = \max\{x, -x\}$ и обе функции $f_1(x) = x$ и $f_2(x) = -x$ выпуклы.

12.1.2. Липшицевость

Ниже приведено определение липшицевости относительно евклидовой нормы в $\mathbb{R}^d$. Но, в принципе, можно определить липшицевость относительно любой нормы.

Определение 12.6 (липшицевость). Пусть $C \subset \mathbb{R}^d$. Функция $f : \mathbb{R}^d \to \mathbb{R}^k$ называется ρ-липшицевой на C, если для любых $\mathbf{w}_1, \mathbf{w}_2 \in C$ имеет место неравенство $||f(\mathbf{w}_1) - f(\mathbf{w}_2)|| \le \rho ||\mathbf{w}_1 - \mathbf{w}_2||$.

Интуитивно понятно, что липшицева функция не может изменяться слишком быстро. Отметим, что если $f : \mathbb{R} \to \mathbb{R}$ дифференцируема, то по теореме о среднем значении имеем

$$f(w_1) - f(w_2) = f'(u)(w_1 - w_2),$$

где u – некоторая точка между w_1 и w_2. Отсюда следует, что если производная f всюду ограничена (по абсолютной величине) значением ρ, то f является ρ-липшицевой.

Пример 12.4

- Функция $f(x) = |x|$ является 1-липшицевой на $\mathbb{R}$. Это следует из неравенства треугольника: для любых x_1, x_2

$$|x_1| - |x_2| = |x_1 - x_2 + x_2| - |x_2| \le |x_1 - x_2| + |x_2| - |x_2| = |x_1 - x_2|.$$

  Так как это справедливо как для x_1, x_2, так и для x_2, x_1, то получаем, что $||x_1| - |x_2|| \le |x_1 - x_2|$.
- Функция $f(x) = \log(1 + \exp(x))$ является 1-липшицевой на $\mathbb{R}$. Чтобы доказать это, заметим, что

$$|f'(x)| = \left|\frac{\exp(x)}{1+\exp(x)}\right| = \left|\frac{1}{\exp(-x)+1}\right| \le 1.$$

- Функция $f(x) = x^2$ не является ρ-липшицевой на $\mathbb{R}$ ни при каком ρ. Действительно, возьмем $x_1 = 0$ и $x_2 = 1 + \rho$, тогда

$$f(x_2) - f(x_1) = (1 + \rho)^2 > \rho(1 + \rho) = \rho|x_2 - x_1|.$$

  Однако эта функция является ρ-липшицевой на множестве $C = \{x : |x| \le \rho/2\}$. В самом деле, для любых $x_1, x_2 \in C$ имеем

$$|x_1^2 - x_2^2| = |x_1 + x_2|\,|x_1 - x_2| \le 2(\rho/2)|x_1 - x_2| = \rho|x_1 - x_2|.$$

- Линейная функция $f : \mathbb{R}^d \to \mathbb{R}$ вида $f(\mathbf{w}) = \langle \mathbf{v}, \mathbf{w}\rangle + b$ для некоторого $\mathbf{v} \in \mathbb{R}^d$, является $||\mathbf{v}||$-липшицевой. Действительно, согласно неравенству Коши–Буняковского,

$$|f(\mathbf{w}_1) - f(\mathbf{w}_2)| = |\langle \mathbf{v}, \mathbf{w}_1 - \mathbf{w}_2\rangle| \le ||\mathbf{v}||\,||\mathbf{w}_1 - \mathbf{w}_2||.$$

Следующее утверждение показывает, что композиция липшицевых функций вновь является липшицевой.

Утверждение 12.7. *Пусть* $f(\mathbf{x}) = g_1(g_2(\mathbf{x}))$, *где* g_1 – ρ_1*-липшицева функция*, *а* g_2 – ρ_2*-липшицева функция. Тогда* f *является* $(\rho_1\rho_2)$*-липшицевой функцией. В частности, если* g_2 – *линейная функция*, $g_2(\mathbf{x}) = \langle \mathbf{v}, \mathbf{x}\rangle + b$, *для некоторых* $\mathbf{v} \in \mathbb{R}^d$, $b \in \mathbb{R}$, *то* f *является* $(\rho_1||\mathbf{v}||)$*-липшицевой.*

Доказательство.

$$\begin{aligned} |f(\mathbf{w}_1) - f(\mathbf{w}_2)| &= |g_1(g_2(\mathbf{w}_1)) - g_1(g_2(\mathbf{w}_2))| \\ &\le \rho_1||g_2(\mathbf{w}_1) - g_2(\mathbf{w}_2)|| \\ &\le \rho_1\rho_2||\mathbf{w}_1 - \mathbf{w}_2||. \end{aligned}$$

□

12.1.3. Гладкость

Определение гладкой функции опирается на понятие градиента. Напомним, что градиентом дифференцируемой функции f: $\mathbb{R}^d \to \mathbb{R}$ в точке $\mathbf{w}$, обозначаемым $\nabla f(\mathbf{w})$, называется вектор частных производных f, т. е. $\nabla f(\mathbf{w}) = \left(\frac{\partial f(\mathbf{w})}{\partial w_1}, \ldots, \frac{\partial f(\mathbf{w})}{\partial w_d}\right)$.

Определение 12.8 (гладкость). Дифференцируемая функция $f : \mathbb{R}^d \to \mathbb{R}^d$ называется β-гладкой, если ее градиент – β-липшицева функция, т. е. для любых $\mathbf{v}$, $\mathbf{w}$ имеем $|\nabla f(\mathbf{v}) - \nabla f(\mathbf{w})|| \le \beta||\mathbf{v} - \mathbf{w}||$.

Можно показать, что из гладкости следует, что для любых $\mathbf{v}$, $\mathbf{w}$

$$f(\mathbf{v}) \le f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{v} - \mathbf{w}\rangle + (\beta/2)||\mathbf{v} - \mathbf{w}||^2. \tag{12.5}$$

Напомним, что из выпуклости f следует, что $f(\mathbf{v}) \ge f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{v} - \mathbf{w}\rangle$. Поэтому, если функция одновременно выпуклая и гладкая, то разность между функцией и ее приближением первого порядка можно ограничить сверху и снизу.

Полагая $\mathbf{v} = \mathbf{w} - (1/\beta)\nabla f(\mathbf{w})$ в правой части неравенства (12.5), после перегруппировки членов получаем

$$(1/2\beta)||\nabla f(\mathbf{v} - \mathbf{w})||^2 \le f(\mathbf{w}) - \le f(\mathbf{v}).$$

Если к тому же предположить, что $f(\mathbf{v}) \ge 0$ для всех $\mathbf{v}$, то мы заключаем, что из гладкости вытекает следующее неравенство

$$||\nabla f(\mathbf{w})||^2 \le 2\beta f(\mathbf{w}). \tag{12.6}$$

Функция, обладающая таким свойством, называется *самоограниченной*.

Пример 12.5

- Функция $f(x) = x^2$ 2-гладкая. Это сразу следует из того факта, что $f'(x) = 2x$. Отметим, что для этой функции неравенства (12.5) и (12.6) обращаются в равенства.
- Функция $f(x) = \log(1 + \exp(x))$ (1/4)-гладкая. Действительно, поскольку $f'(x) = 1/(1 + \exp(-x))$, имеем

$$\left|f''(x)\right| = \frac{\exp(-x)}{(1+\exp(-x))^2} = \frac{1}{(1+\exp(-x))(1+\exp(x))} \le 1/4.$$

  Следовательно, f' является (1/4)-липшицевой. Поскольку эта функция неотрицательна, выполняется также неравенство (12.6).

Следующее утверждение показывает, что композиция гладкой скалярной функции и линейной функции также является гладкой.

Утверждение 12.9. Пусть $f(\mathbf{w}) = g(\langle \mathbf{w}, \mathbf{x}\rangle + b)$, где $g : \mathbb{R} \to \mathbb{R}$ – β-гладкая функция, $\mathbf{x} \in \mathbb{R}^d$ и $b \in \mathbb{R}$. Тогда f является $(\beta||\mathbf{x}||^2)$-гладкой.

Доказательство. По правилу дифференцирования сложной функции, $\nabla f(\mathbf{w}) = g'(\langle \mathbf{w}, \mathbf{x}\rangle + b)\mathbf{x}$, где g'– производная g. В силу гладкости g и неравенства Коши–Буняковского получаем

$$\begin{aligned}
f(v) &= g(\langle \mathbf{v}, \mathbf{w}\rangle + \beta) \\
&\le g(\langle \mathbf{w}, \mathbf{x}\rangle + \beta) + g'(\langle \mathbf{w}, \mathbf{x}\rangle + \beta)\langle \mathbf{v} - \mathbf{w}, \mathbf{x}\rangle + \frac{\beta}{2}(\langle \mathbf{v} - \mathbf{w}, \mathbf{x}\rangle)^2 \\
&\le g(\langle \mathbf{w}, \mathbf{x}\rangle + \beta) + g'(\langle \mathbf{w}, \mathbf{x}\rangle + \beta)\langle \mathbf{v} - \mathbf{w}, \mathbf{x}\rangle + \frac{\beta}{2}(\|\mathbf{v} - \mathbf{w}\|\|\mathbf{x}\|)^2 \\
&= f(\mathbf{w}) + \left\langle \nabla f(\mathbf{w}), \mathbf{v} - \mathbf{w}\right\rangle + \frac{\beta\|\mathbf{x}\|^2}{2}\|\mathbf{v} - \mathbf{w}\|^2 .
\end{aligned}$$

□

Пример 12.6

- Для любых $\mathbf{x} \in \mathbb{R}^d$ и $y \in \mathbb{R}$ положим $f(\mathbf{w}) = (\langle \mathbf{w}, \mathbf{x}\rangle - y)^2$. Тогда f является $(2||\mathbf{x}||^2)$-гладкой.
- Для любых $\mathbf{x} \in \mathbb{R}^d$ и $y \in \{\pm 1\}$ положим $f(\mathbf{w}) = \log(1 + \exp(-y\langle \mathbf{w}, \mathbf{x}\rangle))$. Тогда f является $(||\mathbf{x}||^2/4)$-гладкой.

12.2. Выпуклые проблемы обучения

Напомним, что в нашем общем определении обучения (определение 3.4) имеется класс гипотез $\mathcal{H}$, множество примеров Z и функция потерь $\ell: \mathcal{H} \times Z \to \mathbb{R}_+$. До сих пор мы обычно считали, что Z – декартово произведение области образцов и области меток, $Z = \mathcal{X} \times \mathcal{Y}$, а $\mathcal{H}$ – множество функций из $\mathcal{X}$ в $\mathcal{Y}$. Однако же $\mathcal{H}$ может быть произвольным множеством. И в этой главе мы будем рассматривать классы гипотез $\mathcal{H}$, являющиеся подмножествами евклидова пространства $\mathbb{R}^d$. То есть каждая гипотеза будет вещественным вектором. Поэтому будем обозначать принадлежащие $\mathcal{H}$ гипотезы буквой $\mathbf{w}$. Вот теперь мы наконец можем определить, что такое выпуклая проблема обучения.

Определение 12.10 (выпуклая проблема обучения). Проблема обучения $(\mathcal{H}, Z, \ell)$ называется выпуклой, если класс гипотез $\mathcal{H}$ – выпуклое множество и для всех $z \in Z$ функция потерь $\ell(\cdot, z)$ является выпуклой (где для любого $z\ell(\cdot, z)$ обозначает функцию $f: \mathcal{H} \to \mathbb{R}$, определенную следующим образом: $f(\mathbf{w}) = \ell(\mathbf{w}, z)$).

Пример 12.7 (линейная регрессия с квадратичной функцией потерь). Напомним, что линейная регрессия – это средство моделирования связи между некоторыми «объясняющими» переменными и вещественным результатом (см. главу 9). Область образцов $\mathcal{X}$ – подмножество $\mathbb{R}^d$ для некоторого d, а область меток $\mathcal{Y}$ – множество вещественных чисел. Мы хотим обучить такую линейную функцию $h : \mathbb{R}^d \to \mathbb{R}$, которая лучше всего аппроксимирует связь между переменными. В главе 9 мы определили класс гипотез как множество однородных линейных функций, $\mathcal{H} = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle : \mathbf{w} \in \mathbb{R}^d\}$ и использовали квадратичную функцию потерь, $\ell(h,(\mathbf{x}, y)) = (h(\mathbf{x}) - y)^2$. Однако существует эквивалентная модель, представляющая эту проблему как выпуклую проблему обучения. Каждая линейная функция параметризуется вектором $\mathbf{w} \in \mathbb{R}^d$. Поэтому мы можем определить $\mathcal{H}$ как множество всех таких параметров, т. е. $\mathcal{H} = \mathbb{R}^d$. Множество примеров $Z = \mathcal{X} \times \mathcal{Y} = \mathbb{R}^d \times \mathbb{R} = \mathbb{R}^{d+1}$, а функция потерь имеет вид $\ell(\mathbf{w}, (\mathbf{x}, y)) = (\langle \mathbf{w}, \mathbf{x} \rangle - y)^2$. Очевидно, что $\mathcal{H}$ – выпуклое множество. Функция потерь также выпукла относительно первого аргумента (см. пример 12.2).

Лемма 12.11. *Если ℓ – выпуклая функция потерь, а $\mathcal{H}$ – выпуклый класс, то* $\mathrm{ERM}_{\mathcal{H}}$*-проблема минимизации эмпирической потери на $\mathcal{H}$ является задачей выпуклой оптимизации (т. е. задачей минимизации выпуклой функции на выпуклом множестве).*

Доказательство. Напомним, что $\mathrm{ERM}_{\mathcal{H}}$-проблема определяется так:

$$\mathrm{ERM}_{\mathcal{H}}(S) = \operatorname*{argmin}_{\mathbf{w} \in \mathcal{H}} L_S(\mathbf{w}).$$

Поскольку для выборки $S = z_1, \ldots, z_m$ и для любого $\mathbf{w} L_S(\mathbf{w}) = (1/m)\sum_{i=1}^{m} \ell(\mathbf{w}, z_i)$, то из утверждения 12.5 следует, что $L_S(\mathbf{w})$ – выпуклая функция. Так как правило ERM – это задача минимизации выпуклой функции при заданном ограничении, то решение должно быть выпуклым множеством. □

При довольно мягких условиях такие задачи можно эффективно решить, применяя общие алгоритмы оптимизации. В частности, в главе 14 мы познакомимся с очень простым алгоритмом минимизации выпуклых функций.

12.2.1. Обучаемость выпуклых проблем обучения

Мы утверждали, что во многих случаях реализация правила ERM для выпуклых проблем обучения может быть произведена эффективно. Но является ли выпуклость достаточным условием обучаемости?

Поставим вопрос более конкретно: обсуждая теорию Вапника–Червоненкиса, мы видели, что полупространства в $\mathbb{R}^d$ обучаемы (быть может, неэффективно). В главе 9 мы также говорили, что с помощью дискретизации мы можем обучить проблему с d параметрами, и выборочная сложность будет функцией от d. Это значит, что при постоянном d проблема должна быть обучаема. Так, может быть, все выпуклые проблемы обучения в $\mathbb{R}^d$ обучаемы?

Пример 12.8 ниже показывает, что ответ отрицательный, даже при малых d. Не все выпуклые проблемы в $\mathbb{R}^d$ обучаемы. И здесь нет противоречия с теорией Вапника–Червоненкиса, поскольку эта теория относится только к бинарной классификации, а мы рассматриваем более широкий круг проблем. Нет противоречия и с методом дискретизации, поскольку там мы предполагали, что функция потерь ограничена, а в представлении каждого параметра участвует конечное число битов. Ниже мы покажем, что при некоторых дополнительных условиях, выполняющихся во многих практических ситуациях, выпуклые проблемы допускают обучение.

Пример 12.8 (необучаемость линейной регрессии даже при $d = 1$). Пусть $\mathcal{H} = \mathbb{R}$, а функция потерь квадратичная: $\ell(w, (x, y)) = (wx - y)^2$ (мы рассматриваем однородный случай). Пусть A – произвольный детерминированный алгоритм[1]. Предположим противное: что A является успешным PAC-обучаемым для этой проблемы. Это означает, что существует функция $m(\cdot, \cdot)$ такая, что для любого распределения $\mathcal{D}$ и для любых ϵ, δ справедливо следующее утверждение: если A получает обучающий набор размера $m \geq m(\epsilon, \delta)$, то с вероятностью не менее $1 - \delta$ он должен вывести гипотезу $\hat{w} = A(S)$ такую, что $L_{\mathcal{D}}(\hat{w}) - \min_w L_{\mathcal{D}}(w) \leq \epsilon$.

Выберем $\epsilon = 1/100$, $\delta = 1/2$, положим $m \geq m(\epsilon, \delta)$ и возьмем $\mu = \log(100/99)/(2m)$. Мы определим два распределения и покажем, что A с большой вероятностью потерпит неудачу по крайней мере на одном из них. Первое распределение $\mathcal{D}_1$ сосредоточено на двух примерах $z_1 = (1, 0)$ и $z_2 = (\mu, -1)$, так что масса вероятности первого примера равна μ, а масса вероятности второго – $1 - \mu$. Второе распределение $\mathcal{D}_2$ сосредоточено целиком на примере z_2.

Заметим, что для обоих распределений вероятность того, что все примеры из обучающего набора принадлежат второму типу, составляет не менее 99%. Это очевидно для $\mathcal{D}_2$, а для $\mathcal{D}_1$ вероятность этого события равна

$$(1 - \mu)^m \geq e^{-2\mu m} = 0{,}99.$$

Мы предположили, что A – детерминированный алгоритм, поэтому, получив обучающий набор из m примеров, равных $(\mu, -1)$, алгоритм выводит некоторую гипотезу $\hat{w}$. Если оказывается, что $\hat{w} < -1/(2\mu)$, то мы берем в качестве распределения $\mathcal{D}_1$. Тогда

[1] Точнее, при заданной выборке S выход A детерминирован. Это требование введено для простоты. Немного усложнив рассуждение, можно показать, что и недетерминированные алгоритмы не способны обучить эту проблему.

$$L_{\mathcal{D}_1}(\hat{w}) \geq \mu(\hat{w})^2 \geq 1/(4\mu).$$

Однако

$$\min_w L_{\mathcal{D}_1}(w) \leq L_{\mathcal{D}_1}(0) = (1 - \mu).$$

Отсюда следует, что

$$L_{\mathcal{D}_1}(\hat{w}) - \min_w L_{\mathcal{D}_1}(w) \geq 1/(4\mu) - (1 - \mu) > \epsilon.$$

Поэтому такой алгоритм терпит неудачу на $\mathcal{D}_1$. С другой стороны, если оказывается, что $\hat{w} \geq -1/(2\mu)$, то мы берем в качестве распределения $\mathcal{D}_2$. Тогда $L_{\mathcal{D}2}(\hat{w}) \geq 1/4$, тогда как $\min_w L_{\mathcal{D}2}(w) = 0$, поэтому A терпит неудачу на $\mathcal{D}_2$. В итоге мы показали, что для любого A существует распределение, на котором A терпит неудачу, а это значит, что проблема не допускает PAC-обучения.

Возможный выход – добавить ограничение на класс гипотез. Помимо выпуклости, мы потребуем, чтобы класс $\mathcal{H}$ был *ограничен*, т. е. должен существовать такой скаляр B, что для любой гипотезы $\mathbf{w} \in \mathcal{H} \|\mathbf{w}\| \leq B$.

Только лишь выпуклости и ограниченности еще недостаточно, для того чтобы проблема была обучаемой. Это показывает следующий пример.

Пример 12.9. Как и в примере 12.8, рассмотрим проблему регрессии с квадратичной функцией потерь. Но на этот раз пусть $\mathcal{H} = \{w : |w| \leq 1\} \subset \mathbb{R}$ будет ограниченным классом гипотез. Легко проверить, что $\mathcal{H}$ – выпуклое множество. Мы будем рассуждать так же, как в примере 12.8, только теперь оба распределения $\mathcal{D}_1$и $\mathcal{D}_2$ будут сосредоточены на $z_1 = (1/\mu, 0)$ и $z_2 = (1, -1)$. Если алгоритм A возвращает $\hat{w} < -1/2$, получив m примеров второго типа, то берем в качестве распределения $\mathcal{D}_1$ и имеем

$$L_{\mathcal{D}_1}(\hat{w}) - \min_w L_{\mathcal{D}_1}(w) \geq \mu(\hat{w}/\mu)^2 - L_{\mathcal{D}_1}(0) \geq 1/(4\mu) - (1 - \mu) > \epsilon.$$

Аналогично, если $\hat{w} \geq -1/2$, то берем в качестве распределения $\mathcal{D}_2$ и имеем

$$L_{\mathcal{D}_2}(\hat{w}) - \min_w L_{\mathcal{D}_2}(w) \geq (-1/2 + 1)^2 - 0 > \epsilon.$$

Этот пример показывает, что нужны дополнительные предположения о задаче обучения, и теперь мы потребуем, чтобы функция потерь была липшицевой или гладкой. Эти приводит нас к определению двух семейств проблем обучения: выпуклая–липшицева–ограниченная и выпуклая–гладкая–ограниченная.

12.2.2. Выпуклые-липшицевы/гладкие-ограниченные проблемы обучения

Определение 12.12 (выпуклая-липшицева-ограниченная проблема обучения). Проблема обучения $(\mathcal{H}, Z, \ell)$ называется выпуклой-липшицевой-ограниченной с параметрами ρ, B, если выполняются следующие условия:

- класс гипотез $\mathcal{H}$ – выпуклое множество и для любого $\mathbf{w} \in \mathcal{H}$ имеем $\|\mathbf{w}\| \leq B$;
- для всех $z \in Z$ функция потерь $\ell(\cdot, z)$ является выпуклой и ρ-липшицевой.

Пример 12.10. Пусть $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \leq \rho\}$ и $\mathcal{Y} = \mathbb{R}$. Положим $\mathcal{H} = \{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\| \leq B\}$ и возьмем такую функцию потерь $\ell(\mathbf{w}, (\mathbf{x}, y)) = |\langle \mathbf{w}, \mathbf{x} \rangle - y|$. Это соответствует проблеме регрессии с абсолютной величиной в качестве функции потерь, когда мы предполагаем, что образцы находятся в шаре радиуса ρ, а гипотезы могут быть только однородными линейными функциями, определяемыми вектором $\mathbf{w}$ с нормой не больше B. Тогда получающаяся проблема является выпуклой–липшицевой–ограниченной с параметрами ρ, B.

ЛАНЬ®

Определение 12.13 (выпуклая–гладкая–ограниченная проблема обучения). Проблема обучения $(\mathcal{H}, Z, \ell)$ называется выпуклой–гладкой–ограниченной с параметрами β, B, если выполняются следующие условия:

- класс гипотез $\mathcal{H}$ – выпуклое множество и для любого $\mathbf{w} \in \mathcal{H}$ имеем $\|\mathbf{w}\| \leq B$;
- для всех $z \in Z$ функция потерь $\ell(\cdot, z)$ является выпуклой, неотрицательной и β-гладкой.

Отметим, что мы потребовали также, чтобы функция потерь была неотрицательной. Это необходимо, чтобы функция потерь гарантированно была самоограниченной (см. предыдущий раздел).

Пример 12.11. Пусть $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \leq \beta/2\}$ и $\mathcal{Y} = \mathbb{R}$. Положим $\mathcal{H} = \{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\| \leq B\}$ и возьмем такую функцию потерь $\ell(\mathbf{w}, (\mathbf{x}, y)) = (\langle \mathbf{w}, \mathbf{x} \rangle - y)^2$. Это соответствует проблеме регрессии с квадратичной функцией потерь, когда мы предполагаем, что образцы находятся в шаре радиуса $\beta/2$, а гипотезы могут быть только однородными линейными функциями, определяемыми вектором $\mathbf{w}$ с нормой не больше B. Тогда получающаяся проблема является выпуклой–гладкой–ограниченной с параметрами β, B.

Мы утверждаем, что оба эти семейства проблем являются обучаемыми. То есть свойств выпуклости, ограниченности и липшицевости или гладкости функции потерь достаточно для обучаемости. Мы докажем это в следующих разделах, описав алгоритмы, которые успешно обучают такие проблемы.

ЛАНЬ®

12.3. Суррогатные функции потерь

Как уже было сказано и как мы увидим в следующих главах, выпуклые проблемы допускают эффективное обучение. Однако во многих случаях естественная функция потерь не является выпуклой и реализовать правило ERM трудно.

В качестве примера рассмотрим проблему обучения класса полупространств относительно бинарной функции потерь вида

$$\ell^{0-1}(\mathbf{w}, (\mathbf{x}, y)) = \mathbb{1}_{[y \neq \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)]} = \mathbb{1}_{[y \langle \mathbf{w}, \mathbf{x} \rangle \leq 0]}.$$

Эта функция потерь не выпукла относительно $\mathbf{w}$ и действительно при попытке минимизировать относительно нее эмпирический риск мы можем столкнуться с локальными минимумами (см. упражнение 12.1). Кроме того, в главе 8 отмечалось, что проблема ERM относительно бинарной функции потерь в нереализуемом случае – NP-трудная задача.

Чтобы обойти этот результат о трудности, часто применяют такой прием: ограничивают сверху невыпуклую функцию потерь выпуклой суррогатной функ-

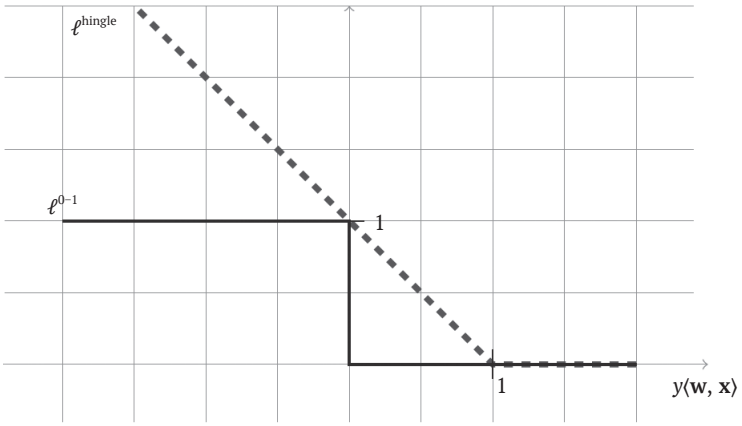
цией потерь. Как следует из названия, к суррогатной функции потерь предъявляются следующие требования:

1) она должна быть выпуклой;
2) она должна быть верхней границей для исходной функции потерь.

Например, в контексте обучения полупространств можно определить следующую кусочно-линейную функцию потерь в качестве выпуклого суррогата бинарной потери:

$$\ell^{\text{hinge}}(\mathbf{w}, (\mathbf{x}, y)) \stackrel{\text{def}}{=} \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}.$$

Очевидно, что для любых $\mathbf{w}$ и $(\mathbf{x}, y)$ $\ell^{0-1}(\mathbf{w}, (\mathbf{x}, y)) \leq \ell^{\text{hinge}}(\mathbf{w}, (\mathbf{x}, y))$. Выпуклость кусочно-линейной потери следует непосредственно из утверждения 12.5. Поэтому кусочно-линейная потеря удовлетворяет требованиям, предъявляемым к суррогатной функции потерь для бинарной потери. Ниже приведены графики функций ℓ^{0-1} и ℓ^{hinge}.



Определив суррогатную выпуклую функцию потерь, мы можем обучить проблему относительно нее. Общее требование для обучаемого с кусочно-линейной потерей будет иметь вид

$$L_{\mathcal{D}}^{\text{hinge}}(A(S)) \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{\text{hinge}}(\mathbf{w}) + \epsilon,$$

где $L_{\mathcal{D}}^{\text{hinge}}(\mathbf{w}) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\ell^{\text{hinge}}(\mathbf{w}, (\mathbf{x}, y))]$. В силу свойства суррогатности мы можем ограничить снизу левую часть величиной $L_{\mathcal{D}}^{0-1}(A(S))$, что дает

$$L_{\mathcal{D}}^{0-1}(A(S)) \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{\text{hinge}}(\mathbf{w}) + \epsilon.$$

Далее можно переписать верхнюю границу в виде:

$$L_{\mathcal{D}}^{0-1}(A(S)) \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{0-1}(\mathbf{w}) + \left(\left(\min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{\text{hingle}}(\mathbf{w}) - \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{0-1}(\mathbf{w})\right) + \epsilon.\right.$$

Таким образом, бинарная ошибка обученного предиктора ограничена сверху тремя членами:

- *ошибка аппроксимации* – это член $\min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{0-1}(\mathbf{w})$, который измеряет, насколько хорошо эта гипотеза соответствует распределению. Мы уже подробно говорили по поводу этого члена ошибки в главе 5;
- *ошибка оценивания.* – это ошибка, проистекающая из того факта, что мы получаем только обучающий набор, а само распределение $\mathcal{D}$ не наблюдаем. Об этом члене мы тоже много говорили в главе 5;
- *ошибка оптимизации* – это член $\left(\min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{\text{hinge}}(\mathbf{w}) - \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}^{0-1}(\mathbf{w})\right)$, который измеряет разность между ошибкой аппроксимации относительно суррогатной потери и ошибкой аппроксимации относительно исходной потери. Ошибка оптимизации – результат нашей неспособности минимизировать потерю обучения относительно исходной потери. Величина этой ошибки зависит от конкретного распределения данных и конкретной суррогатной функции потерь.

12.4. Резюме

Мы ввели два семейства проблем обучения: выпуклые–липшицевы–ограниченные и выпуклые–гладкие–ограниченные. В следующих двух главах мы опишем два общих алгоритма обучения для этих семейств. Мы также ввели понятие выпуклой суррогатной функции потерь, что позволяет использовать аппарат выпуклой оптимизации для обучения невыпуклых проблем.

12.5. Библиографические сведения

Есть несколько отличных книг по выпуклому анализу и оптимизации (Boyd & Vandenberghe, 2004; Borwein & Lewis, 2006; Bertsekas, 1999; Hiriart-Urruty & Lemaréchal, 1993). Что касается проблем обучения, семейство выпуклых–липшицевых–ограниченных функций впервые было изучено в работе Zinkevich (2003) в контексте онлайнового обучения и в работе Shalev-Shwartz, Shamir, Sridharan и Srebro (2009) в контексте PAC-обучения.

12.6. Упражнения

12.1. Приведите пример, показывающий, что обучение с бинарной функцией потерь может иметь локальные минимумы. Точнее, постройте обучающую выборку $S \in (\mathcal{X} \times \{\pm 1\})^m$ (скажем, для $\mathcal{X} = \mathbb{R}^2$), для которой существует вектор $\mathbf{w}$ и $\epsilon > 0$ такие, что

1) для любого $\mathbf{w}'$ такого, что $\|\mathbf{w} - \mathbf{w}'\| \le \epsilon$, имеет место $L_S(\mathbf{w}) \le LS(\mathbf{w}')$ (при этом используется бинарная функция потерь). Это означает, что $\mathbf{w}$ – локальный минимум L_S;
2) существует такой $\mathbf{w}^*$, что $L_S(\mathbf{w}^*) < L_S(\mathbf{w})$. Это означает, что $\mathbf{w}$ не является глобальным минимумом L_S.

12.2. Рассмотрим проблему обучения логистической регрессии: пусть $\mathcal{H} = \mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \le B\}$ при некотором $B > 0$, $\mathcal{Y} = \{\pm 1\}$, и функция потерь ℓ определена

так: $\ell(\mathbf{w}, (\mathbf{x}, y)) = \log(1 + \exp(-y\langle \mathbf{w}, \mathbf{x}\rangle))$. Покажите, что такая проблема обучения является одновременно выпуклой–липшицевой–ограниченной и выпуклой–гладкой–ограниченной. Определите параметры липшицевости и гладкости.

12.3. Рассмотрим проблему обучения полупространств с кусочно-линейной функцией потерь. Ограничим область определения евклидовым шаром радиуса R, т. е. положим $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\|^2 \leq R\}$. В качестве множества меток возьмем $\mathcal{Y} = \{\pm 1\}$, а функцию потерь определим так: $\ell(\mathbf{w}, (\mathbf{x}, y)) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}$. Мы уже знаем, что эта функция потерь выпукла. Покажите, что она также R-липшицева.

12.4. (*) Выпуклости–липшицевости–ограниченности недостаточно для вычислительной эффективности. В следующей главе мы покажем, что со статистической точки зрения все выпуклые–липшицевы–ограниченные проблемы обучаемы (в агностической модели PAC). Однако наш основной интерес к таким проблемам связан с вычислительной перспективой – выпуклая оптимизация часто допускает эффективную реализацию. Тем не менее, в этом упражнении мы покажем, что одной выпуклости еще недостаточно для эффективности, а именно будет показано, что в случае $d = 1$ существует выпуклая–липшицева–ограниченная проблема, для которой эффективный алгоритм обучения отсутствует.

Возьмем в качестве класса гипотез $\mathcal{H} = [0, 1]$, а в качестве множества примеров Z множество всех машин Тьюринга. Определим функцию потерь следующим образом. Для любой машины Тьюринга $T \in Z$ положим $\ell(0, T) = 1$, если T останавливается на входе 0 и $\ell(0, T) = 0$, если T не останавливается на входе 0. Аналогично $\ell(1, T) = 0$, если T останавливается на входе 0 и $\ell(1, T) = 1$, если T не останавливается на входе 0. Наконец, для $h \in (0, 1)$ положим $\ell(h, T) = h\ell(0, T) + (1 - h)\ell(1, T)$.

1. Покажите, что эта проблема обучения выпуклая–липшицева–ограниченная.
2. Покажите, что для нее не существует вычислимого алгоритма обучения.

ЛАНЬ®

Глава 13

РЕГУЛЯРИЗАЦИЯ И УСТОЙЧИВОСТЬ

В предыдущей главе мы ввели семейства выпуклых–липшицевых–ограниченных и выпуклых–гладких–ограниченных проблем обучения. В этой главе мы покажем, что все проблемы из обоих семейств обучаемы. Для некоторых проблем обучения такого типа возможно доказать равномерную сходимость, поэтому они допускают обучение по правилу ERM. Однако это верно не для всех таких проблем обучения. Тем не менее мы введем еще одно правило обучения и покажем, что с его помощью можно обучить любую выпуклую–липшицеву–ограниченную или выпуклую–гладкую–ограниченную проблему.

Новая парадигма обучения называется *минимизацией регуляризированной потери* (Regularized Loss Minimization – RLM). В этом случае мы минимизируем сумму эмпирического риска и некоей функции регуляризации. На интуитивном уровне функция регуляризации измеряет сложность гипотез. В действительности одна из интерпретаций функции регуляризации в парадигме структурной минимизации риска обсуждалась в главе 7. Возможен и другой взгляд на регуляризацию – как на *стабилизатор* алгоритма обучения. Алгоритм называется устойчивым, если небольшое изменение входа приводит к небольшому изменению выхода. Мы формально определим понятие устойчивости (что понимается под «небольшим изменением входа» и что значит «приводит к небольшому изменению выхода») и докажем существование тесной связи этого понятия с обучаемостью. Наконец, мы покажем, что использование квадратичной нормы ℓ_2 в качестве функции регуляризации стабилизирует все выпуклые–липшицевы и выпуклые–гладкие проблем обучения. Поэтому RLM можно использовать в качестве общего правила обучения для этих семейств.

13.1. Минимизация регуляризированной потери

Минимизация регуляризированной потери (RLM) – это правило обучения, при котором производится совместная минимизация эмпирического риска и функции регуляризации. Формально функция регуляризации определяется как отображение $R : \mathbb{R}^d \to \mathbb{R}$, а правило минимизации регуляризированной потери ищет гипотезу в множестве

$$\operatorname{argmin}_{\mathbf{w}} (L_S(\mathbf{w}) + R(\mathbf{w})). \tag{13.1}$$

У минимизации регуляризированной потери есть общие черты с алгоритмами минимальной длины описания и структурной минимизации риска (см. главу 7). Интуитивно «сложность» гипотез измеряется значением функции регуляризации, а алгоритм ищет баланс между низким эмпирическим риском и «более простой», или «менее сложной», гипотезой.

Существует много функций регуляризации, отражающих некоторое априорное знание о проблеме (по аналогии с языком описания в парадигме минимальной длины описания). В этом разделе мы сосредоточимся на одной из самых простых функций регуляризации: $R(\mathbf{w}) = \lambda||\mathbf{w}||^2$, где $\lambda > 0$ – скаляр, а в качестве нормы выступает норма ℓ_2, $||\mathbf{w}|| = \sqrt{\sum_{i=1}^{d} w_i^2}$. В результате получаем такое правило обучения:

$$A(S) = \operatorname{argmin}_{\mathbf{w}} (L_S(\mathbf{w}) + \lambda||\mathbf{w}||^2). \tag{13.2}$$

Функцию регуляризация такого типа часто называют регуляризацией Тихонова. Как уже отмечалось, одна из интерпретаций формулы (13.2) – структурная минимизация риска, когда норма $\mathbf{w}$ измеряет «сложность». Напомним, что в предыдущей главе мы ввели понятие ограниченного класса гипотез. Поэтому можно определить последовательность классов гипотез $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \mathcal{H}_3 \ldots$, где $\mathcal{H}_i = \{\mathbf{w} : ||\mathbf{w}||^2 \leq i\}$. Если выборочная сложность каждого $\mathcal{H}_i$ зависит от i, то правило RLM аналогично правилу SRM для этой последовательности вложенных классов.

Регуляризацию можно также интерпретировать как стабилизатор. В следующем разделе мы определим понятие устойчивости и докажем, что устойчивое правило обучения не подвержено переобучению. Но сначала продемонстрируем правило RLM для линейной регрессии с квадратичной потерей.

13.1.1. Гребневая регрессия

Применив правило RLM с регуляризацией Тихонова к линейной регрессии с квадратичной потерей, мы получим такое правило обучения:

$$\operatorname*{arg\,min}_{\mathbf{w}\in\mathbb{R}^d} \left(\lambda \|\mathbf{w}\|_2^2 + \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 \right). \tag{13.3}$$

Линейная регрессия, описываемая выражением (13.3), называется *гребневой регрессией*.

Для решения задачи (13.3) мы приравниваем градиент целевой функции к нулю и получаем систему линейных уравнений:

$$(2\lambda m I + A)\mathbf{w} = \mathbf{b},$$

где I – единичная матрица, а A, $\mathbf{b}$ определены формулами (9.6), а именно:

$$A = \left(\sum_{i=1}^{m} \mathbf{x}_i \mathbf{x}_i^\top \right) \quad \text{и} \quad \mathbf{b} = \sum_{i=1}^{m} y_i \mathbf{x}_i \,. \tag{13.4}$$

Поскольку A – положительно полуопределенная матрица, все собственные значения матрицы $2\lambda mI + A$ ограничены снизу величиной $2\lambda m$. Следовательно, эта матрица обратима и решение уравнения гребневой регрессии принимает вид

$$\mathbf{w} = (2\lambda mI + A)^{-1}\mathbf{b}. \tag{13.5}$$

В следующем разделе мы формально покажем, как регуляризация стабилизирует алгоритм и предотвращает переобучение. В частности, из этого анализа (см. следствие 13.11) будет вытекать такая теорема.

Теорема 13.1. *Пусть $\mathcal{D}$ – распределение на $\mathcal{X} \times [-1, 1]$, где $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \leq 1\}$. Пусть $\mathcal{H} = \{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\| \leq B\}$. Для любого $\epsilon \in (0, 1)$ возьмем $m \geq 150\ B^2/\epsilon^2$. Тогда результат применения алгоритма гребневой регрессии с параметром $\lambda = \epsilon/(3B^2)$ удовлетворяет неравенству*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S))] \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \epsilon.$$

Замечание 13.1. Эта теорема говорит, сколько примеров необходимо, чтобы гарантировать, что *математическое ожидание* риска обученного предиктора ограничено ошибкой аппроксимации класса плюс ϵ. В обычном определении агностического PAC-обучения мы требуем, чтобы риск обученного предиктора был ограничен с вероятностью не менее $1 - \delta$. В упражнении 13.1 мы покажем, как можно использовать алгоритм с ограниченным ожидаемым риском для построения агностического PAC-обучаемого.

13.2. Устойчивые правила не подвержены переобучению

Интуитивно понятно, что алгоритм обучения устойчив, если небольшое изменение входа не приводит к большому изменению выхода. Конечно, есть много способов определить, что такое «небольшое изменение входа» и что понимается под словами «не приводит к большому изменению выхода». В этом разделе мы определим одно конкретное понятие устойчивости и докажем, что при таком определении устойчивые правила не подвержены переобучению.

Пусть A – алгоритм обучения, $S = (z_1, \ldots, z_m)$ – обучающий набор m примеров, а $A(S)$ – выход A. Алгоритм A переобучен, если разность между истинным риском его выхода, $L_{\mathcal{D}}(A(S))$, и эмпирическим риском, $L_S(A(S))$, велика. Как отмечалось в замечании 13.1, в этой главе нас в первую очередь интересует математическое ожидание этой величины (относительно выбора S), т. е. $\mathbb{E}_S[L_{\mathcal{D}}(A(S)) - L_S(A(S))]$.

Далее определим понятие устойчивости. Пусть дан обучающий набор S и дополнительный пример z'. Обозначим $S^{(i)}$ обучающий набор, полученный из S заменой i-ого примера на z', т. е. $S^{(i)} = (z_1, \ldots, z_{i-1}, z', z_{i+1}, \ldots, z_m)$. В нашем определении устойчивости «небольшое изменение входа» означает, что A подается набор $S^{(i)}$ вместо S, т. е. мы заменяем только один обучающий пример. Чтобы измерить влияние этого изменения на выход A, мы сравниваем потерю гипотезы $A(S)$ на z_i с потерей гипотезы $A(S^{(i)})$ на z_i. Интуитивно понятно, что для хорошего алго-

ритма $\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i) \geq 0$, поскольку в уменьшаемом алгоритм обучения вообще не наблюдает z_i, а в вычитаемом z_i наблюдается. Если эта разность очень велика, то алгоритм можно заподозрить в переобученности, поскольку он кардинально меняет предсказание на z_i, если наблюдает его в обучающем наборе. Это формализовано в следующей теореме.

Теорема 13.2. *Пусть $\mathcal{D}$ – некоторое распределение, $S = (z_1, \ldots, z_m)$ – последовательность независимых и одинаково распределенных примеров, а z' – еще один независимый пример, выбранный из того же распределения. Обозначим $U(m)$ равномерное распределение на множестве $[m]$. Тогда для любого алгоритма обучения*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S)) - L_S(A(S))] = \mathop{\mathbb{E}}_{(S,z')\sim\mathcal{D}^{m+1},\, i\sim U(m)}[\ell(A(S^{(i)}, z_i)) - \ell(A(S), z_i)]. \tag{13.6}$$

Доказательство. Так как S и z' независимо выбираются из распределения $\mathcal{D}$, то для любого i имеем

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(A(S))] = \mathop{\mathbb{E}}_{S,z'}[\ell(A(S, z'))] - \mathop{\mathbb{E}}_{S,z'}[\ell(A(S^{(i)})), z_i)].$$

С другой стороны, можно написать

$$\mathop{\mathbb{E}}_{S}[L_S(A(S))] = \mathop{\mathbb{E}}_{S,i}[\ell(A(S, z_i)].$$

Объединение обеих формул завершает доказательство. □

Когда правая часть равенства (13.6) мала, говорят, что алгоритм A *устойчивый* – изменение одного примера в обучающем наборе не приводит к существенным изменениям. Дадим формальное определение.

Определение 13.3 (устойчивость в среднем относительно одиночной замены). Пусть $\epsilon : \mathbb{N} \to \mathbb{R}$ – монотонно убывающая функция. Говорят, что алгоритм обучения A является устойчивым в среднем относительно одиночной замены (on-average-replace-one-stable) со скоростью $\epsilon(m)$, если для любого распределения $\mathcal{D}$

$$\mathop{\mathbb{E}}_{(S,z')\sim\mathcal{D}^{m+1},\, i\sim U(m)}[\ell(A(S^{(i)}, z_i)) - \ell(A(S), z_i)] \leq \epsilon_m.$$

Теорема 13.2 говорит, то алгоритм не подвержен переобучению тогда и только тогда, когда он устойчив в среднем относительно одиночной замены. Конечно, алгоритм, не подверженный переобучению, не обязательно хорош – взять, к примеру, алгоритм A, который всегда выводит одну и ту же гипотезу. Полезный алгоритм должен отыскивать гипотезу, которая, с одной стороны, аппроксимирует обучающий набор (т. е. имеет низкий эмпирический риск), а с другой стороны – не склонна к переобучению. Или, в свете теоремы 13.2, алгоритм должен аппроксимировать обучающий набор и в то же время быть устойчивым. Как мы увидим, параметр λ правила RLM управляет балансом между хорошей аппроксимацией и устойчивостью.

13.3. Регуляризация Тихонова как стабилизатор

В предыдущем разделе мы видели, что устойчивые правила не подвержены переобучению. А сейчас мы покажем, что применение правила RLM с регуля-

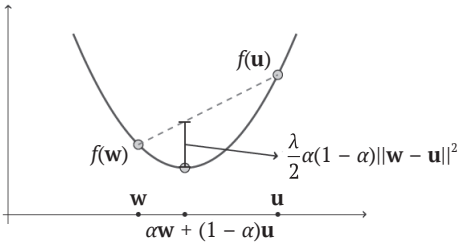
ризацией Тихонова $\lambda||\mathbf{w}||^2$ дает устойчивый алгоритм. Будем предполагать, что функция потерь выпуклая и либо липшицева, либо гладкая.

Основное нужное нам свойство регуляризации Тихонова состоит в том, что оно делает целевую функцию RLM *строго выпуклой*.

Определение 13.4 (строго выпуклая функция). Функция f называется λ-строго выпуклой, если для всех $\mathbf{w}$, $\mathbf{u}$ и $\alpha \in (0, 1)$ имеем

$$f(\alpha\mathbf{w} + (1 - \alpha)\mathbf{u}) \leq \alpha f(\mathbf{w}) + (1 - \alpha)f(\mathbf{u}) - \frac{\lambda}{2}\alpha(1 - \alpha)||\mathbf{w} - \mathbf{u}||^2.$$

Очевидно, что всякая выпуклая функция является 0-строго выпуклой. Рисунок ниже иллюстрирует строгую выпуклость.



Из следующей леммы следует, что целевая функция RLM (2λ)-строго выпукла. Кроме того, она подчеркивает важное свойство строгой выпуклости.

Лемма 13.5

1. *Функция* $f(\mathbf{w}) = \lambda||\mathbf{w}||^2$ *является* 2λ*-строго выпуклой.*
2. *Если* f *является* λ*-строго выпуклой, а* g *– выпуклая функция, то* $f + g$ *будет* λ*-строго выпуклой.*
3. *Если* f *является* λ*-строго выпуклой, а* $\mathbf{u}$ *доставляет минимум* f*, то для любого* $\mathbf{w}$ *имеет место неравенство*

$$f(\mathbf{w}) - f(\mathbf{u}) \geq \frac{\lambda}{2}||\mathbf{w} - \mathbf{u}||^2.$$

Доказательство. Первые два утверждения вытекают непосредственно из определения. Чтобы доказать третье, разделим определение строгой выпуклости на α и перегруппируем члены

$$\frac{f(\mathbf{u} + \alpha(\mathbf{w} - \mathbf{u})) - f(\mathbf{u})}{\alpha} \leq f(\mathbf{w}) - f(\mathbf{u}) - \frac{\lambda}{2}(1 - \alpha)\|\mathbf{w} - \mathbf{u}\|^2.$$

Взяв предел при $\alpha \to 0$, получим, что правая часть стремится к $f(\mathbf{w}) - f(\mathbf{u}) - (\lambda/2)||\mathbf{w} - \mathbf{u}||^2$. С другой стороны, левая часть стремится к производной функции $g(\alpha) = f(\mathbf{u} + \alpha(\mathbf{w} - \mathbf{u}))$ в точке $\alpha = 0$. Из того, что $\mathbf{u}$ доставляет минимум f, следует, что $\alpha = 0$ доставляет минимум g, и, следовательно, левая часть неравенства стремится к нулю при $\alpha \to 0$, что и завершает доказательство. □

Теперь обратимся к доказательству устойчивости RLM. Пусть $S = (z_1, \ldots, z_m)$ – обучающий набор, z' – дополнительный пример и $S^{(i)} = (z_1, \ldots, z_{i-1}, z', z_{i+1}, \ldots, z_m)$. Пусть A – правило RLM, т. е.

$$A(S) = \operatorname*{argmin}_{\mathbf{w}} \left(L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2\right).$$

Обозначим $f_S(\mathbf{w}) = L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2$, тогда из леммы 13.5 мы знаем, что f_S является (2λ)-строго выпуклой. Из части 3 леммы следует, что для любого $\mathbf{v}$

$$f_S(\mathbf{v}) - f_S(A(S)) \geq \lambda\|\mathbf{v} - A(S)\|^2. \tag{13.7}$$

С другой стороны, для любых $\mathbf{v}$ и $\mathbf{u}$ и для любого i имеем

$$\begin{aligned} f_S(\mathbf{v}) - f_S(\mathbf{u}) &= L_S(\mathbf{v}) + \lambda\|\mathbf{v}\|^2 - (L_S(\mathbf{u}) + \lambda\|\mathbf{u}\|^2) \\ &= L_{S^{(i)}}(\mathbf{v}) + \lambda\|\mathbf{v}\|^2 - (L_{S^{(i)}}(\mathbf{u}) + \lambda\|\mathbf{u}\|^2) \\ &\quad + \frac{\ell(\mathbf{v}, z_i) - \ell(\mathbf{u}, z_i)}{m} + \frac{\ell(\mathbf{u}, z') - \ell(\mathbf{v}, z')}{m}. \end{aligned} \tag{13.8}$$

В частности, взяв $\mathbf{v} = A(S^{(i)})$, $\mathbf{u} = A(S)$ и воспользовавшись тем фактом, что $\mathbf{v}$ доставляет минимум $L_{S^{(i)}}(\mathbf{w}) + \lambda\|\mathbf{w}\|^2$, получаем

$$f_S(A(S^{(i)})) - f_S(A(S)) \leq \frac{\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i)}{m} + \frac{\ell(A(S), z') - \ell(A(S^{(i)}), z')}{m}. \tag{13.9}$$

В сочетании с (3.7) это дает

$$\lambda\|A(S^{(i)}) - A(S)\|^2 \leq \frac{\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i)}{m} + \frac{\ell(A(S), z') - \ell(A(S^{(i)}), z')}{m}. \tag{13.10}$$

В двух следующих подразделах мы продолжим анализ устойчивости липшицевых или гладких функций потерь. Для обоих семейств функций мы покажем, что правило RLM устойчиво и, значит, не подвержено переобучению.

13.3.1. Липшицева потеря

Если функция потерь $\ell(\cdot, z_i)$ является ρ-липшицевой, то по определению липшицевости

$$\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i) \leq \|A(S^{(i)}) - A(S)\|. \tag{13.11}$$

Аналогично,

$$\ell(A(S), z') - \ell(A(S^{(i)}), z') \leq \|A(S^{(i)}) - A(S)\|.$$

Подставляя оба эти неравенства в (13.10), получаем

$$\lambda\|A(S^{(i)}) - A(S)\|^2 \leq \frac{2\rho\|A(S^{(i)}) - A(S)\|}{m},$$

откуда

$$\|A(S^{(i)}) - A(S)\| \leq \frac{2\rho}{\lambda m}.$$

Подставляя это назад в (13.11), приходим к выводу, что

$$\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i) \leq \frac{2\rho^2}{\lambda m}.$$

Так как это справедливо для любых S, z', i, мы немедленно получаем

Следствие 13.6. *Предположим, что функция потерь выпуклая и ρ-липшицева. Тогда правило RLM с регуляризатором $\lambda\|\mathbf{w}\|^2$ является устойчивым в среднем относительно одиночной замены со скоростью $(2\rho^2)/(\lambda m)$. Отсюда следует (в силу теоремы 13.2), что*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S)) - L_S(A(S))] \leq \frac{2\rho^2}{\lambda m}.$$

13.3.2. Гладкая неотрицательная потеря

Если функция потерь является β-гладкой и неотрицательной, то она также самоограничена (см. раздел 12.1):

$$\|\nabla f(\mathbf{w})\|^2 \leq 2\beta f(\mathbf{w}). \tag{13.12}$$

Предположим еще, что $\lambda \geq 2\beta/m$ или, иными словами, что $\beta \leq \lambda m/2$. В силу предположения о гладкости имеем

$$\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i) \leq \langle \nabla \ell(A(S), z_i), A(S^{(i)}) - A(S)\rangle + (\beta/2)\|A(S^{(i)}) - A(S)\|. \tag{13.13}$$

Из неравенства Коши–Буняковского и формулы (12.6) получаем, что

$$\begin{aligned}
&\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i) \\
&\qquad \leq \|\nabla \ell(A(S), z_i)\| \, \|A(S^{(i)}) - A(S)\| + (\beta/2)\|A(S^{(i)}) - A(S)\|. \\
&\qquad \leq \sqrt{2\beta \ell(A(S), z_i)} \, \|A(S^{(i)}) - A(S)\| + (\beta/2)\|A(S^{(i)}) - A(S)\|.
\end{aligned} \tag{13.14}$$

Симметричное рассуждение показывает, что

$$\begin{aligned}
&\ell(A(S^{(i)}), z') - \ell(A(S), z') \\
&\qquad \leq \sqrt{2\beta \ell(A(S), z')} \, \|A(S^{(i)}) - A(S)\| + (\beta/2)\|A(S^{(i)}) - A(S)\|.
\end{aligned}$$

Подставляя эти неравенства в (13.10), после перегруппировки членов получаем

$$\|A(S^{(i)}) - A(S)\| \leq \frac{\sqrt{2\beta}}{(\lambda m - \beta)}\left(\sqrt{\ell(A(S), z_i)} + \sqrt{\ell(A(S^{(i)}), z')}\right).$$

Объединение этого неравенства с предположением $\beta \leq \lambda m/2$ дает

$$\|A(S^{(i)}) - A(S)\| \leq \frac{\sqrt{8\beta}}{\lambda m}\left(\sqrt{\ell(A(S), z_i)} + \sqrt{\ell(A(S^{(i)}), z')}\right).$$

Объединяя с (13.14) и вновь используя предположение $\beta \le \lambda m/2$, получаем

$$\begin{aligned}
&\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i) \\
&\quad \le \sqrt{2\beta\ell(A(S), z_i)}\, \|A(S^{(i)}) - A(S)\| + \frac{\beta}{2}\|A(S^{(i)}) - A(S)\|^2 \\
&\quad \le \left(\frac{4\beta}{\lambda m} + \frac{8\beta^2}{(\lambda m)^2}\right)\left(\sqrt{\ell(A(S), z_i)} + \sqrt{\ell(A(S^{(i)}), z')}\right)^2 \\
&\quad \le \frac{8\beta}{\lambda m}\left(\sqrt{\ell(A(S), z_i)} + \sqrt{\ell(A(S^{(i)}), z')}\right)^2 \\
&\quad \le \frac{24\beta}{\lambda m}\left(\ell(A(S), z_i) + \ell(A(S^{(i)}), z')\right),
\end{aligned}$$

где на последнем шаге мы воспользовались неравенством $(a + b)^2 \le 3(a^2 + b^2)$. Беря математическое ожидание по S, z' и i и замечая, что $\mathbb{E}[\ell(A(S), z_i)] = \mathbb{E}[\ell(A(S^{(i)}), z')] = \mathbb{E}[L_S(A(S))]$, приходим к такому выводу:

Следствие 13.7. *Предположим, что функция потерь является β-гладкой и неотрицательной. Тогда правило RLM с регуляризатором $\lambda\|\mathbf{w}\|^2$, где $\lambda \ge 2\beta/m$, удовлетворяет условию*

$$\mathbb{E}[\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i)] \le \frac{48\beta}{\lambda m}\mathbb{E}[L_S(A(S))].$$

Заметим, что если для всех z имеет место неравенство $\ell(\mathbf{0}, z) \le C$ для некоторого скаляра $C > 0$, то для любого S

$$L_S(A(S)) \le L_S(A(S)) + \lambda\|A(S)\|^2 \le L_S(\mathbf{0}) + \lambda\|\mathbf{0}\|^2 = L_S(\mathbf{0}) \le C.$$

Поэтому из следствия 13.7 также вытекает, что

$$\mathbb{E}[\ell(A(S^{(i)}), z_i) - \ell(A(S), z_i)] \le \frac{48\beta C}{\lambda m}.$$

13.4. Управление компромиссом между аппроксимацией и устойчивостью

Мы можем переписать выражение для ожидаемого риска алгоритма обучения в виде

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(A(S))] = \mathop{\mathbb{E}}_{S}[L_S(A(S))] + \mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(A(S)) - L_S(A(S))]. \tag{13.15}$$

Первый член показывает, насколько хорошо $A(S)$ аппроксимирует обучающий набор, а второй член отражает разность между истинным и эмпирическим риском $A(S)$. В теореме 13.2 мы доказали, что второй член эквивалентен устойчивости A. Поскольку наша цель – минимизировать риск алгоритма, то нужно, чтобы сумма обоих членов была мала.

В предыдущем разделе мы ограничили член, отвечающий за устойчивость. Мы показали, что он уменьшается вместе с ростом параметра регуляризации λ.

С другой стороны, эмпирический риск увеличивается с ростом λ. Таким образом, приходится выбирать между аппроксимацией и переобучением. Ситуация очень похожа на компромисс между смещением и сложностью, с которым мы встречались в этой книге ранее.

Теперь ограничим член, отвечающий за эмпирический риск, в правиле RLM. Напомним, что правило RLM определено как $A(S) = \operatorname{argmin}_{\mathbf{w}}(L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2)$. Зафиксируем произвольный вектор $\mathbf{w}^*$. Имеем

$$L_S(A(S)) \leq L_S(A(S)) + \lambda\|A(S)\|^2 \leq L_S(\mathbf{w}^*) + \lambda\|\mathbf{w}^*\|^2.$$

Взяв математическое ожидание обеих частей по S и заметив, что $\mathbb{E}_S[L_S(\mathbf{w}^*)] = L_{\mathcal{D}}(\mathbf{w}^*)$, получаем

$$\mathbb{E}_S[L_S(A(S))] \leq L_{\mathcal{D}}(\mathbf{w}^*) + \lambda\|\mathbf{w}^*\|^2. \tag{13.16}$$

Подставляя это в (13.15), получаем

$$\mathbb{E}_S[L_{\mathcal{D}}(A(S))] \leq L_{\mathcal{D}}(\mathbf{w}^*) + \lambda\|\mathbf{w}^*\|^2 + \mathbb{E}_S[L_{\mathcal{D}}(A(S)) - L_S(A(S))].$$

Объединяя со следствием 13.6, мы приходим к такому выводу.

Следствие 13.8. *Предположим, что функция потерь выпуклая и ρ-липшицева. Тогда правило RLM с функцией регуляризации $\lambda\|\mathbf{w}\|^2$ удовлетворяет условию*

$$\forall \mathbf{w}^*,\ \mathbb{E}_S[L_{\mathcal{D}}(A(S))] \leq L_{\mathcal{D}}(\mathbf{w}^*) + \lambda\|\mathbf{w}^*\|^2 + \frac{2\rho^2}{\lambda m}.$$

Эту границу часто называют *неравенством оракула* – если мы рассматриваем $\mathbf{w}^*$ как гипотезу с низким риском, то эта граница говорит, сколько примеров было бы необходимо, чтобы $A(S)$ была почти так же хороша, как $\mathbf{w}^*$, если бы мы знали норму $\mathbf{w}^*$. На практике, однако, норма $\mathbf{w}^*$ обычно неизвестна. Поэтому мы настраиваем λ, ориентируясь на контрольный набор, как описано в главе 11.

Из следствия 13.8 легко также вывести PAC-подобную гарантию[1] для выпуклых–липшицевых–ограниченных проблем обучения.

Следствие 13.9. *Пусть $(\mathcal{H}, Z, \ell)$ – выпуклая–липшицева–ограниченная проблема обучения с параметрами ρ, B. Для любого размера обучающего набора m положим $\lambda = \sqrt{\frac{2\rho^2}{B^2 m}}$. Тогда правило RLM с функцией регуляризации $\lambda\|\mathbf{w}\|^2$ удовлетворяет условию*

$$\mathbb{E}_S[L_{\mathcal{D}}(A(S))] \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \rho B\sqrt{\frac{8}{m}}.$$

В частности, для любого $\epsilon > 0$, если $m \geq (8\rho^2 B^2)/\epsilon^2$, то для любого распределения $\mathcal{D}$ имеем $\mathbb{E}_S[L_{\mathcal{D}}(A(S))] \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \epsilon$.

[1] И снова приведенная ниже граница относится к ожидаемому риску, но с помощью упражнения 13.1 можно вывести гарантию типа агностического PAC-обучения.

Это следствие справедливо для липшицевых функций потерь. Если же функция потерь гладкая и неотрицательная, то, объединив (13.16) со следствием 13.7, получим

Следствие 13.10. *Предположим, что функция потерь выпуклая, β-гладкая и неотрицательная. Тогда правило RLM с функцией регуляризации $\lambda\|\mathbf{w}\|^2$ для любого $\lambda \geq 2\beta/m$ удовлетворяет следующему условия для всех $\mathbf{w}^*$:*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(A(S))] \leq \left(1+\frac{48\beta}{\lambda m}\right)\mathop{\mathbb{E}}_{S}[L_S(A(S))] \leq \left(1+\frac{48\beta}{\lambda m}\right)(L_{\mathcal{D}}(\mathbf{w}^*)+\lambda\|\mathbf{w}^*\|^2).$$

Например, если взять $\lambda = 48\beta/m$, то получим, что ожидаемый истинный риск $A(S)$ не более чем в два раза превосходит ожидаемый эмпирический риск $A(S)$. Для того же значения λ ожидаемый эмпирический риск $A(S)$ не превосходит $L_{\mathcal{D}}(\mathbf{w}^*) + (48\beta/m)\|\mathbf{w}^*\|^2$.

Из следствия 13.10 можно также вывести гарантию обучаемости для выпуклых–гладких–ограниченных проблем.

Следствие 13.11. *Пусть $(\mathcal{H}, Z, \ell)$ – выпуклая–гладкая–ограниченная проблема обучения с параметрами β, B. Предположим дополнительно, что $\ell(\mathbf{0}, z) \leq 1$ для всех $z \in Z$. Для любого $\epsilon \in (0, 1)$ выберем $m \geq 150\beta B^2/\epsilon^2$ и положим $\lambda = \epsilon/(3B^2)$. Тогда для любого распределения $\mathcal{D}$*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(A(S))] \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \epsilon.$$

13.5. Резюме

Мы ввели понятие устойчивости и показали, что устойчивые алгоритмы не подвержены переобучению. Мы также показали, что для выпуклых–липшицевых–ограниченных или выпуклых–гладких–ограниченных проблем правило RLM с регуляризацией Тихонова приводит к устойчивому алгоритму обучения. Мы обсудили, как параметр регуляризации λ управляет компромиссом между аппроксимацией и переобучением. Наконец, мы показали, что все проблемы обучения, принадлежащие семействам выпуклых–липшицевых–ограниченных и выпуклых–гладких–ограниченных проблем, обучаемы по правилу RLM. Парадигма RLM лежит в основе многих популярных алгоритмов обучения, в т. ч. гребневой регрессии (см. предыдущую главу) и метода опорных векторов (см. главу 15).

В следующей главе мы представим метод стохастического градиентного спуска, который дает практически удобную альтернативу обучению выпуклых–липшицевых–ограниченных и выпуклых–гладких–ограниченных проблем и может быть также применен для эффективной реализации правила RLM.

13.6. Библиографические сведения

Устойчивость встречается в самых разных математических контекстах. Например, устойчивость как необходимое условие для корректной постановки так на-

зываемых обратных задач впервые была осознана Адамаром в работе Hadamard (1902). Идея регуляризации и ее связь с устойчивостью получили широкую известность благодаря работам Tikhonov (1943) и Phillips (1962). Применение устойчивости в контексте современной теории обучения восходит к работе Rogers and Wager (1978), которые отметили, что от чувствительности алгоритма обучения к малым изменениям выборки зависит дисперсия оценки с исключением по одному. Авторы воспользовались этим наблюдением, чтобы обобщить границы для алгоритма k ближайших соседей (см. главу 19). Впоследствии эти результаты были обобщены на другие «локальные» алгоритмы обучения (см. Devroye, Györfi & Lugosi, 1996) и приведенные там ссылки). Были также разработаны практические методы для обеспечения устойчивости алгоритмов обучения, например, метод баггинга (Breiman, 1996).

В течение последних десяти лет устойчивость изучалась как общее условие обучаемости. См. Kearns & Ron, 1999; Bousquet & Elisseeff, 2002; Kutin & Niyogi, 2002; Rakhlin, Mukherjee & Poggio, 2005; Mukherjee, Niyogi, Poggio & Rifkin, 2006. В своем изложении мы следовали работе Shalev-Shwartz, Shamir, Srebo, and Sridharan (2010), где показано, что устойчивость является необходимым и достаточным условием обучаемости. Там же показано, что выпуклые–липшицевы–ограниченные проблемы допускают обучение по правилу RLM, несмотря даже на то, что для некоторых проблем такого вида равномерная сходимость в строгом смысле отсутствует.

13.7. Упражнения

13.1. От ограниченного ожидаемого риска к агностическому PAC-обучению. Пусть A – алгоритм, гарантирующий следующее: если $m \geq m_{\mathcal{H}}(\epsilon)$, то для любого распределения $\mathcal{D}$ справедливо неравенство

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S))] = \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(h) + \epsilon.$$

- Покажите, что для любого $\delta \in (0, 1)$, если $m \geq m_{\mathcal{H}}(\epsilon\delta)$, то с вероятностью не менее $1 - \delta$ имеет место оценка $L_{\mathcal{D}}(A(S)) \leq \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$.
  Указание. Заметьте, что случайная величина $L_{\mathcal{D}}(A(S)) \leq \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h)$ неотрицательна и воспользуйтесь неравенством Маркова.
- Для любого $\delta \in (0, 1)$ положим

$$m_{\mathcal{H}}(\epsilon,\delta) = m_{\mathcal{H}}(\epsilon/2)\lceil\log_2(1/\delta)\rceil + \left\lceil \frac{\log(4/\delta) + \log\left(\lceil\log_2(1/\delta)\rceil\right)}{\epsilon^2} \right\rceil.$$

  Предложите процедуру агностического PAC-обучения проблемы с выборочной сложностью $m_{\mathcal{H}}(\epsilon, \delta)$ в предположении, что функция потерь ограничена сверху 1.
  Указание. Пусть $k = \lceil\log_2(1/\delta)\rceil$. Разделите данные на $k + 1$ порций, так чтобы в каждой из первых k порций было по $m_{\mathcal{H}}(\epsilon/2)$ примеров. Обучите первые k порций с помощью алгоритма A. Опираясь на предыдущую часть упражнения, докажите, что вероятность того, что для всех этих порций $L_{\mathcal{D}}(A(S)) \leq$

$\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$ не превосходит $2^{-k} \leq \delta/2$. Наконец, используйте последнюю порцию в качестве контрольного набора.

13.2. Обучаемость без равномерной сходимости. Пусть B – единичный шар в $\mathbb{R}^d$, $\mathcal{H} = B$, $Z = B \times \{0, 1\}^d$ и ℓ: $Z \times \mathcal{H} \to \mathbb{R}$ определена следующим образом:

$$\ell(\mathbf{w}, (\mathbf{x}, \boldsymbol{\alpha})) = \sum_{i=1}^{d} \alpha_i (x_i - w_i)^2.$$

Эта проблема соответствует задаче обучения *без учителя*, в которой мы не пытаемся предсказать метку **x**. Вместо этого мы хотим найти «центр тяжести» распределения на B. Однако есть дополнительный нюанс, моделируемый вектором $\boldsymbol{\alpha}$. Каждый пример представляет собой пару ($\mathbf{x}$, $\boldsymbol{\alpha}$), где $\mathbf{x}$ – образец, а $\boldsymbol{\alpha}$ показывает, какие признаки $\mathbf{x}$ «активны», а какие «выключены». Гипотеза – это вектор $\mathbf{w}$, представляющий центр тяжести распределения, а в роли функции потерь выступает квадрат евклидова расстояния между $\mathbf{x}$ и $\mathbf{w}$, вычисленный только по «активным» элементам $\mathbf{x}$.

- Покажите, что эта проблема допускает обучение по правилу RLM с выборочной сложностью, не зависящей от d.
- Рассмотрим распределение $\mathcal{D}$ на Z, определенное следующим образом: $\mathbf{x}$ принимает некоторое фиксированное значение $\mathbf{x}_0$, и каждому элементу α случайным образом присваивается значение 0 или 1, выбираемое с равной вероятностью. Покажите, что скорость равномерной сходимости этой проблемы возрастает вместе с d.
  Указание. Пусть m – размер обучающего набора. Покажите, что если $d \gg 2^m$, то высока вероятность выбрать такой набор примеров, что существует $j \in [d]$, для которого $\alpha_j = 1$ для всех примеров из этого набора. Покажите, что такая выборка не может быть ϵ-репрезентативной. Отсюда сделайте вывод, что выборочная сложность равномерной сходимости должна расти как $\log(d)$.
- Покажите, что если устремить d к бесконечности, то мы получим проблему, допускающую обучение, для которой свойство равномерной сходимости не выполняется. Сравните с фундаментальной теоремой статистического обучения.

13.3. Устойчивость и асимптотическая ERM – достаточные условия обучаемости. Говорят, что правило обучения A является AERM (Asymptotic Empirical Risk Minimizer – асимптотическая точка минимума эмпирического риска) со скоростью $\epsilon(m)$, если для любого распределения $\mathcal{D}$ имеет место неравенство:

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} \left[L_S(A(S)) - \min_{h \in \mathcal{H}} L_S(h) \right] \leq \epsilon(m).$$

Говорят, что правило обучения A обучает класс $\mathcal{H}$ со скоростью $\epsilon(m)$, если для любого распределения $\mathcal{D}$ имеет место неравенство:

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} \left[L_{\mathcal{D}}(A(S)) - \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \right] \leq \epsilon(m).$$

Докажите следующую теорему.

Теорема 13.12. *Если алгоритм обучения A является устойчивым в среднем относительно одиночной замены со скоростью $\epsilon_1(m)$ и одновременно AERM со скоростью $\epsilon_2(m)$, то он обучает $\mathcal{H}$ со скоростью $\epsilon_1(m) + \epsilon_2(m)$.*

ЛАНЬ®

13.4. Сильная выпуклость относительно норм общего вида.

Всюду в этом разделе мы использовали норму ℓ_2. В этом упражнении мы обобщим некоторые результаты на произвольные нормы. Пусть $\|\cdot\|$ – произвольная норма, и f – строго выпуклая функция относительно этой нормы (см. определение 13.4).

1. Докажите, что утверждения 2–3 леммы 13.5 остаются справедливы для любой нормы.
2. (*) Приведите пример нормы, для которой утверждение 1 леммы 13.5 не выполняется.
3. Пусть $R(\mathbf{w})$ – функция, (2λ)-строго выпуклая относительно некоторой нормы $\|\cdot\|$. Пусть A – правило RLM относительно R, т. е.

$$A(S) = \operatorname*{argmin}_{\mathbf{w}} (L_S(\mathbf{w}) + R(\mathbf{w})).$$

Предположим, что для любого z функция потерь $\ell(\cdot, z)$ является ρ-липшицевой относительно той же нормы, т. е.

$$\forall z, \forall \mathbf{w}, \mathbf{v}, \quad \ell(\mathbf{w}, z) - \ell(\mathbf{v}, z) \leq \rho\|\mathbf{w} - \mathbf{v}\|.$$

Докажите, что A в среднем устойчиво относительно одиночной замены со скоростью $2\rho^2/(\lambda m)$.

4. (*) Пусть $q \in (1, 2)$, рассмотрим норму ℓ_q

$$\|\mathbf{w}\|_q = \left(\sum_{i=1}^{d} |w_i|^q\right)^{1/q}.$$

Можно показать (см., например, Shalev-Shwartz (2007)), что функция

$$R(\mathbf{w}) = \frac{1}{2(q-1)}\|\mathbf{w}\|_q^2$$

является 1-строго выпуклой относительно $\|\mathbf{w}\|_q$. Покажите, что если $q = \log(d)/(\log(d) - 1)$, то $R(\mathbf{w})$ является $(1/3\log(d))$-строго выпуклой относительно нормы ℓ_1 на $\mathbb{R}^d$.

ЛАНЬ®

Глава 14

СТОХАСТИЧЕСКИЙ ГРАДИЕНТНЫЙ СПУСК

Напомним, что цель обучения – минимизировать функцию риска $L_{\mathcal{D}}(h) = \mathbb{E}_{z\sim\mathcal{D}}[\ell(h, z)]$. Мы не можем минимизировать функцию риска напрямую, потому что она зависит от неизвестного распределения $\mathcal{D}$. До сих пор мы обсуждали методы обучения, опирающиеся на эмпирический риск. То есть сначала мы выбираем обучающий набор S и определяем функцию эмпирического риска $L_S(h)$. Затем обучаемый выбирает гипотезу, основываясь на значении $L_S(h)$. Например, правило ERM требует выбирать гипотезу, которая минимизирует $L_S(h)$ на классе гипотез $\mathcal{H}$. А в предыдущей главе мы обсуждали минимизацию регуляризированного риска, когда выбирается гипотеза, которая совместно минимизирует $L_S(h)$ и функцию регуляризации на h.

В этой главе мы опишем и проанализируем совсем другой подход к обучению, который называется стохастическим градиентным спуском (СГС). Как и в главе 12, нас будет интересовать важное семейство выпуклых проблем обучения, и, следуя введенной там нотации, мы будем считать гипотезами векторы $\mathbf{w}$, которые берутся из выпуклого класса гипотез. В методе СГС мы пытаемся минимизировать функцию риска $L_{\mathcal{D}}(\mathbf{w})$, непосредственно применяя процедуру градиентного спуска. Градиентный спуск – это процедура итеративной оптимизации, когда на каждой итерации мы улучшаем решение, делая шаг в направлении, противоположном градиенту минимизируемой функции в текущей точке. В нашем случае минимизируется функция риска, а поскольку мы не знаем $\mathcal{D}$, то не знаем и градиента $L_{\mathcal{D}}(\mathbf{w})$. СГС обходит эту трудность, разрешая процедуре оптимизации делать шаг в случайном направлении при условии, что математическое ожидание направления противоположно градиенту. И, как мы увидим, найти случайное направление, ожидаемое значение которого соответствует градиенту, довольно просто, даже если мы не знаем истинного распределения $\mathcal{D}$.

В контексте выпуклых проблем обучения преимущество СГС над минимизацией регуляризированного риска состоит в том, что СГС – эффективный алгоритм, который можно реализовать в нескольких строчках кода, и при этом его выборочная сложность такая же, как у правила минимизации регуляризированного риска.

Мы начнем эту главу с базового алгоритма градиентного спуска и проанализируем скорость его сходимости для выпуклых липшицевых функций. Затем мы

введем понятие субградиента и покажем, что метод градиентного спуска применим и к недифференцируемым функциям. В главном разделе 14.3 мы опишем алгоритм стохастического градиентного спуска и несколько его полезных вариантов. Мы покажем, что ожидаемая скорость сходимости СГС близка к скорости сходимости градиентного спуска. Наконец, мы поговорим о применимости СГС к проблемам обучения.

ЛАНЬ®

14.1. Градиентный спуск

Прежде чем описывать метод стохастического градиентного спуска, мы расскажем о стандартном подходе к минимизации дифференцируемой выпуклой функции $f(\mathbf{w})$ путем градиентного спуска.

Градиентом дифференцируемой функции $f : \mathbb{R}^d \to \mathbb{R}$ в точке $\mathbf{w}$, обозначаемым $\nabla f(\mathbf{w})$, называется вектор частных производных f, т. е. $\nabla f(\mathbf{w}) = \left(\frac{\partial f(\mathbf{w})}{\partial w[1]}, \ldots, \frac{\partial f(\mathbf{w})}{\partial w[d]} \right)$.

Метод градиентного спуска – это итеративный алгоритм. Мы отправляемся от некоторого начального значения $\mathbf{w}$ (скажем, $\mathbf{w}^{(1)} = \mathbf{0}$). Затем на каждой итерации мы делаем шаг в направлении, противоположном градиенту в текущей точке, т. е. один шаг обновления имеет вид:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla f(\mathbf{w}^{(t)}), \tag{14.1}$$

где $\eta > 0$ – параметр, который мы обсудим ниже. Интуитивно понятно, что поскольку градиент указывает в направлении наискорейшего возрастания f в окрестности точки $\mathbf{w}^{(t)}$, то алгоритм, который делает небольшой шаг в противоположном направлении, уменьшает значение функции. В конечном итоге, после T итераций алгоритм выдает усредненный вектор $\bar{\mathbf{w}} = (1/T)\sum_{t=1}^{T} \mathbf{w}^{(t)}$. Можно было бы также вернуть последний вектор $\mathbf{w}^{(T)}$ или лучший вектор, $\operatorname{argmin}_{t \in [T]} f(\mathbf{w}^{(t)})$, но усреднение оказывается довольно полезным, особенно при обобщении градиентного спуска на недифференцируемые функции и на стохастический случай.

Еще одно обоснование градиентного спуска – аппроксимация рядом Тейлора. Градиент f в точке $\mathbf{w}$ дает аппроксимацию f первого порядка в окрестности $\mathbf{w}$: $f(\mathbf{u}) \approx f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla f(\mathbf{w}) \rangle$. Если f выпукла, то эта аппроксимация дает нижнюю границу, т. е.

$$f(\mathbf{u}) \geq f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla f(\mathbf{w}) \rangle.$$

Поэтому для $\mathbf{w}$, близких к $\mathbf{w}(t)$, имеем $f(\mathbf{w}) \approx f(\mathbf{w}^{(t)}) + \langle \mathbf{w} - \mathbf{w}^{(t)}, \nabla f(\mathbf{w}^{(t)}) \rangle$, так что мы можем минимизировать не саму $f(\mathbf{w})$, а ее аппроксимацию. Однако аппроксимация может разойтись с $\mathbf{w}$, отдалившись тем самым от $\mathbf{w}^{(t)}$. Поэтому мы хотели бы совместно минимизировать расстояние между $\mathbf{w}$ и $\mathbf{w}^{(t)}$ и аппроксимацию f в окрестности $\mathbf{w}^{(t)}$. Если параметр η управляет компромиссом между этими двумя членами, то мы получаем правило обновления:

$$\mathbf{w}^{(t+1)} = \operatorname*{argmin}_{\mathbf{w}} \tfrac{1}{2}||\mathbf{w} - \mathbf{w}^{(t)}||^2 + \eta(f(\mathbf{w}^{(t)}) + \langle \mathbf{w} - \mathbf{w}^{(t)}, \nabla f(\mathbf{w}^{(t)}) \rangle).$$

Приравнивая нулю производную по $\mathbf{w}$, мы получаем то же правило, что (14.1).

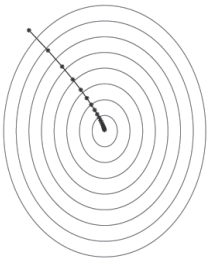


Рис. 14.1. Иллюстрация алгоритма f градиентного спуска. Минимизируется функция $1.25(x_1 + 6)^2 + (x_2 - 8)^2$

14.1.1. Анализ метода ГС для выпуклых липшицевых функций

Чтобы проанализировать скорость сходимости алгоритма ГС, ограничимся случаем выпуклых липшицевых функций (как мы видели, многие проблемы легко сводятся к этому случаю). Пусть $\mathbf{w}^*$ – произвольный вектор, и B – верхняя граница $\|\mathbf{w}^*\|$. Удобно считать, что $\mathbf{w}^*$ – точка минимума $f(\mathbf{w})$, но последующее рассуждение проходит для любого $\mathbf{w}^*$.

Нам хотелось бы оценить сверху субоптимальность нашего решения относительно $\mathbf{w}^*$, т. е. $f(\bar{\mathbf{w}}) - f(\mathbf{w}^*)$, где $\bar{\mathbf{w}} = (1/T)\sum_{t=1}^{T}\mathbf{w}^{(t)}$. Пользуясь определением $\bar{\mathbf{w}}$ и неравенством Йенсена, получаем:

$$\begin{aligned} f(\bar{\mathbf{w}}) - f(\mathbf{w}^*) &= f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}\right) - f(\mathbf{w}^*) \\ &\le \frac{1}{T}\sum_{t=1}^{T}\left(f(\mathbf{w}^{(t)})\right) - f(\mathbf{w}^*) \\ &\le \frac{1}{T}\sum_{t=1}^{T}\left(f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*)\right). \end{aligned} \tag{14.2}$$

Для каждого t в силу выпуклости f имеем

$$f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*) \le \langle \mathbf{w}^{(t)} - \mathbf{w}^*, \nabla f(\mathbf{w}^{(t)})\rangle. \tag{14.3}$$

Объединяя с (14.2), получаем

$$f(\bar{\mathbf{w}}) - f(\mathbf{w}^*) \le \frac{1}{T}\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \nabla f(\mathbf{w}^{(t)})\rangle.$$

Чтобы ограничить сверху правую часть, воспользуемся следующей леммой.

Лемма 14.1. *Пусть* $\mathbf{v}_1, \ldots, \mathbf{v}_T$ *– произвольная последовательность векторов. Любой алгоритм с начальным значением* $\mathbf{w}^{(1)} = 0$ *и правилом обновления вида*

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta\mathbf{v}_t \tag{14.4}$$

удовлетворяет условию

$$\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle \le \frac{\|\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2. \tag{14.5}$$

В частности, для любых B, $\rho > 0$, если для всех t имеет место $||\mathbf{v}_t|| \le \rho$, то, положив $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$, мы для всех $\mathbf{w}^*$ с $||\mathbf{w}^*|| \le B$ будем иметь

$$\frac{1}{T}\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \rangle \le \frac{B\rho}{\sqrt{T}}.$$

Доказательство. Выполняя алгебраические преобразования (дополнение до полного квадрата), получаем

$$\begin{aligned}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \rangle &= \frac{1}{\eta}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \eta\mathbf{v}_t \rangle \\ &= \frac{1}{2\eta}(-\|\mathbf{w}^{(t)} - \mathbf{w}^* - \eta\mathbf{v}_t\|^2 + \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 + \eta^2\|\mathbf{v}_t\|^2) \\ &= \frac{1}{2\eta}(-\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 + \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2) + \frac{\eta}{2}\|\mathbf{v}_t\|^2,\end{aligned}$$

где последнее равенство следует из определения правила обновления. Суммирование по t дает

$$\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \rangle = \frac{1}{2\eta}\sum_{t=1}^{T}(-\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 + \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2) + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2. \quad (14.6)$$

В первой сумме сокращаются все члены, кроме первого и последнего, так что остается

$$||\mathbf{w}^{(1)} - \mathbf{w}^*||^2 - ||\mathbf{w}^{(T+1)} - \mathbf{w}^*||^2.$$

Подставляя в (14.6), получаем

$$\begin{aligned}\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \rangle &= \frac{1}{2\eta}(\|\mathbf{w}^{(1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(T+1)} - \mathbf{w}^*\|^2) + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2 \\ &\le \frac{1}{2\eta}(\|\mathbf{w}^{(1)} - \mathbf{w}^*\|^2) + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2 \\ &= \frac{1}{2\eta}\|\mathbf{w}^*\|^2 + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2,\end{aligned}$$

где последнее равенство вытекает из того, что по определению $\mathbf{w}^{(1)} = 0$. Тем самым доказана первая часть леммы (неравенство (14.5)). Чтобы доказать вторую часть, ограничим $||\mathbf{w}^*||$ сверху величиной B, $||\mathbf{v}_t||$ – величиной ρ, разделим на T и подставим значение η. □

Лемма 14.1 применима к алгоритму ГС, если положить $\mathbf{v}_t = \nabla f(\mathbf{w}^{(t)})$. Ниже, в лемме 14.7, мы покажем, что если f ρ-липшицева, то $||\nabla f(\mathbf{w}^{(t)})|| \le \rho$. Таким образом, условия леммы удовлетворены, и мы получаем такое следствие.

Следствие 14.2. *Пусть f – выпуклая, ρ-липшицева функция и* $\mathbf{w}^* \in \operatorname{argmin}_{\{\mathbf{w}:\|\mathbf{w}\|\le B\}} f(\mathbf{w})$. *Если выполнить T шагов алгоритма ГС для f с параметром* $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$, *то получающийся на выходе вектор удовлетворяет неравенству*

$$f(\bar{\mathbf{w}}) - f(\mathbf{w}^*) \le \frac{B\rho}{\sqrt{T}}.$$

Кроме того, при любом $\epsilon > 0$ для достижения условия $f(\bar{\mathbf{w}}) - f(\mathbf{w}^*) \le \epsilon$ достаточно выполнить T итераций алгоритма ГС, где

$$T \ge \frac{B^2\rho^2}{\epsilon^2}.$$

14.2. Субградиенты

В алгоритме ГС функция f должна быть дифференцируемой. Теперь мы обобщим его на недифференцируемые функции. Мы покажем, что ГС можно применять и к недифференцируемым функциям, если воспользоваться не градиентом, а так называемым субградиентом $f(\mathbf{w})$ в точке $\mathbf{w}^{(t)}$.

Чтобы обосновать определение субградиента, вспомним, что для выпуклой функции f градиент в точке $\mathbf{w}$ определяет наклон касательной, расположенной целиком ниже f, т. е.

$$\forall \mathbf{u}, \quad f(\mathbf{u}) \ge f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla f(\mathbf{w}) \rangle. \tag{14.7}$$

Это показано в левой части рис. 14.2.

Существование касательной, расположенной ниже f, – важное свойство выпуклых функций, оно даже является альтернативной характеристикой выпуклости.

Лемма 14.3. *Пусть S – выпуклое множество. Функция $f : S \to \mathbb{R}$ является выпуклой тогда и только тогда, когда для любого $\mathbf{w} \in S$ существует $\mathbf{v}$ такой, что*

$$\forall \mathbf{u} \in S, \quad f(\mathbf{u}) \ge f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \mathbf{v} \rangle. \tag{14.8}$$

Доказательство этой леммы приведено во многих учебниках по выпуклому анализу (в т. ч., Borwein & Lewis, 2006). Это неравенство подводит нас к определению субградиентов.

Определение 14.4 (субградиенты). Вектор $\mathbf{v}$, удовлетворяющий условию (14.8), называется *субградиентом f* в точке $\mathbf{w}$. Множество субградиентов f в точке $\mathbf{w}$ *субдифференциалом* и обозначается $\partial f(\mathbf{w})$.

Наглядная иллюстрация субградиентов приведена в правой части рис. 14.2. В случае скалярных функций субградиент выпуклой функции f в точке w равен тангенсу угла наклона прямой, которая касается f в w и нигде не проходит выше f.

14.2.1. Вычисление субградиентов

Как построить субградиенты заданной выпуклой функции? Если функция дифференцируема в точке **w**, то построение субдифференциала – тривиальная задача, как показывает следующее утверждение.

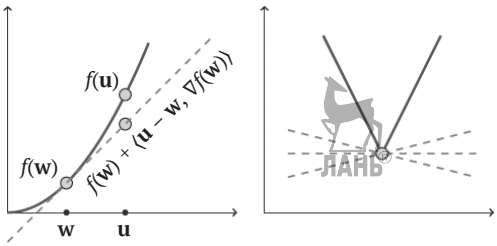


Рис. 14.2. Слева: правая часть неравенства (14.7) – это уравнение касательной к f в точке **w**. Для выпуклой функции касательная расположена ниже f. Справа: несколько субградиентов недифференцируемой выпуклой функции

Утверждение 14.5. *Если f дифференцируема в точке* **w**, *то* $\partial f(\mathbf{w})$ *состоит из единственного элемента – градиента f в* **w**, $\nabla f(\mathbf{w})$.

Пример 14.1 (субдифференциал абсолютной величины). Рассмотрим функцию абсолютной величины $f(x) = |x|$. Применяя утверждение (14.5), мы легко можем построить субдифференциал для любой дифференцируемой части f, а особого внимания требует только точка $x_0 = 0$. Легко проверить, что в ней субдифференциал состоит из множества всех чисел от −1 до 1. Следовательно:

$$\partial f(x) = \begin{cases} \{1\}, & \text{если } x > 0 \\ \{-1\}, & \text{если } x < 0 \\ [-1, 1], & \text{если } x = 0 \end{cases}.$$

Для многих практических надобностей нам не нужно вычислять все множество субградиентов в данной точке, хватает и одного его члена. Следующее утверждение показывает, как построить субградиент максимума нескольких функций.

Утверждение 14.6. *Пусть* $g(\mathbf{w}) = \max_{i \in [r]} g_i(\mathbf{w})$ *для r выпуклых дифференцируемых функций* $g_1, \ldots, g_r$. *Для заданной точки* **w** *положим* $j \in \operatorname{argmax}_i g_i(\mathbf{w})$. *Тогда* $\nabla g_j(\mathbf{w}) \in \partial g(\mathbf{w})$.

Доказательство. Поскольку g_j выпукла, для любого **u** имеем

$$g_j(\mathbf{u}) \geq g_j(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla g_j(\mathbf{w}) \rangle.$$

Так как $g(\mathbf{w}) = g_j(\mathbf{w})$ и $g(\mathbf{u}) \geq g_j(\mathbf{u})$, получаем, что

$$g(\mathbf{u}) \geq g(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla g_j(\mathbf{w}) \rangle,$$

что и завершает доказательство. □

Пример 14.2 (субградиент кусочно-линейной функции потерь). Напомним, что кусочно-линейная функции потерь определена в разделе 12.3 формулой $f(\mathbf{w}) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x} \rangle\}$ для некоторого вектора **x** и скаляра y. Чтобы вычислить субгра-

диент этой функции в некоторой точке $\mathbf{w}$, мы воспользуемся только что доказанным утверждением, согласно которому вектор $\mathbf{v}$, определенный, как показано ниже, является субградиентом кусочно-линейной функции потерь в точке $\mathbf{w}$:

$$\mathbf{v} = \begin{cases} \mathbf{0}, & \text{если } 1 - y\langle \mathbf{w}, \mathbf{x} \rangle \leq 0 \\ -y\mathbf{x}, & \text{если } 1 - y\langle \mathbf{w}, \mathbf{x} \rangle > 0 \end{cases}.$$

14.2.2. Субградиенты липшицевых функций

Напомним, что функция $f : A \to \mathbb{R}$ называется ρ-липшицевой, если для всех $\mathbf{u}, \mathbf{v} \in A$

$$|f(\mathbf{u}) - f(\mathbf{v})| \leq \rho ||\mathbf{u} - \mathbf{v}||.$$

Следующая лемма дает эквивалентное определение с помощью норм субградиентов.

Лемма 14.7. *Пусть A – открытое выпуклое множество и $f : A \to \mathbb{R}$ – выпуклая функция. Тогда f является ρ-липшицевой на A тогда и только тогда, когда для любого $\mathbf{w} \in A$ и $\mathbf{v} \in \partial f(\mathbf{w})$ имеет место неравенство $||\mathbf{v}|| \leq \rho$.*

Доказательство. Предположим, что для любого $\mathbf{v} \in \partial f(\mathbf{w})$ имеем $||\mathbf{v}|| \leq \rho$. Поскольку $\mathbf{v} \in \partial f(\mathbf{w})$, то

$$f(\mathbf{w}) - f(\mathbf{u}) \leq \langle \mathbf{v}, \mathbf{w} - \mathbf{u} \rangle.$$

Применив к правой части неравенство Коши–Буняковского, получаем

$$f(\mathbf{w}) - f(\mathbf{u}) \leq \langle \mathbf{v}, \mathbf{w} - \mathbf{u} \rangle \leq ||\mathbf{v}|| ||\mathbf{w} - \mathbf{u}|| \leq \rho ||\mathbf{w} - \mathbf{u}||.$$

С помощью аналогичного рассуждения можно показать, что $f(\mathbf{u}) - f(\mathbf{w}) \leq \rho ||\mathbf{w} - \mathbf{u}||$. Следовательно, f является ρ-липшицевой.

Теперь предположим, что f – ρ-липшицева функция. Выберем некоторые $\mathbf{w} \in A$, $\mathbf{v} \in \partial f(\mathbf{w})$. Так как A открыто, то существует $\epsilon > 0$ такое, что $\mathbf{u} = \mathbf{w} + \epsilon \mathbf{v}/||\mathbf{v}||$ принадлежит A. Следовательно, $\langle \mathbf{u} - \mathbf{w}, \mathbf{v} \rangle = \epsilon ||\mathbf{v}||$ и $||\mathbf{u} - \mathbf{w}|| = \epsilon$. Из определения субградиента следует, что

$$f(\mathbf{u}) - f(\mathbf{w}) \geq \langle \mathbf{v}, \mathbf{u} - \mathbf{w} \rangle = ||\mathbf{v}||.$$

С другой стороны, в силу липшицевости f имеем

$$\rho\epsilon = \rho ||\mathbf{u} - \mathbf{w}|| \geq f(\mathbf{u}) - f(\mathbf{w}).$$

Объединяя оба неравенства, приходим к выводу, что $||\mathbf{v}|| \leq \rho$. □

14.2.3. Субградиентный спуск

Алгоритм градиентного спуска можно обобщить на недифференцируемые функции, если использовать субградиент $f(\mathbf{w})$ в точке $\mathbf{w}^{(t)}$ вместо градиента. Анализ скорости сходимости остается неизменным, нужно просто заметить, что неравенство (14.3) справедливо и для субградиентов.

14.3. Стохастический градиентный спуск (СГС)

В алгоритме стохастического градиентного спуска мы не требуем, чтобы направление обновления в точности определялось градиентом. Вектор направления может быть случайным при условии, что его математическое ожидание на каждой итерации совпадает с направлением градиента. Или, более общо, мы требуем, чтобы математическое ожидание случайного вектора было субградиентом функции в текущей точке.

ЛАНЬ®

Метод стохастического градиентного спуска (СГС) для минимизации $f(\mathbf{w})$

параметры: скаляр $\eta > 0$, целое $T > 0$
инициализация: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \ldots, T$
  случайным образом выбрать из некоторого распределения $\mathbf{v}_t$, так чтобы $\mathbb{E}[\mathbf{v}_t | \mathbf{w}^{(t)}] \in \partial f(\mathbf{w}^{(t)})$
  произвести обновление $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
выход: $\bar{\mathbf{w}} = (1/T) \sum_{t=1}^{T} \mathbf{w}^{(t)}$

На рис. 14.3 приведено сравнение стохастического градиентного спуска с обычным. В разделе 14.5 мы увидим, что в контексте проблем обучения нетрудно найти случайный вектор, математическое ожидание которого является субградиентом функции риска.

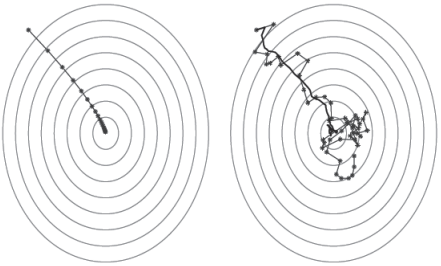


Рис. 14.3. Иллюстрация алгоритма градиентного спуска (слева) и стохастического градиентного спуска (справа). Минимизируется функция $1{,}25(x+6)^2 + (y-8)^2$. В стохастическом случае сплошная линия изображает усредненное значение $\mathbf{w}$

14.3.1. Анализ СГС для выпуклых–липшицевых–ограниченных функций

ЛАНЬ®

Вспомните границу, полученную для алгоритма ГС в следствии 14.2. Для стохастического случая, в котором субдифференциалу $\partial f(\mathbf{w}^{(t)})$ принадлежит только математическое ожидание $\mathbf{v}_t$, не получится напрямую применить неравенство (14.3). Однако, поскольку ожидаемое значение $\mathbf{v}_t$ является субградиентом f в точке $\mathbf{w}^{(t)}$, мы все-таки можем вывести похожую границу *ожидаемого* выхода алгоритма СГС. Это формализовано в следующей теореме.

Теорема 14.8. *Пусть $B, \rho > 0$. Пусть f – выпуклая функция, и $\mathbf{w}^* \in \operatorname{argmin}_{\mathbf{w}:\|\mathbf{w}\|\le B} f(\mathbf{w})$. Предположим, что выполнено T итераций алгоритма СГС с параметром $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$. Предположим также, что для всех $t \|\mathbf{v}_t\| \le \rho$ с вероятностью 1. Тогда*

$$\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{w}^*) \le \frac{B\rho}{\sqrt{T}}.$$

Поэтому для любого $\epsilon > 0$ для достижения условия $\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{w}^*) \le \epsilon$ достаточно выполнить T итераций алгоритма СГС, где

$$T \ge \frac{B^2\rho^2}{\epsilon^2}.$$

Доказательство. Будем обозначать $\mathbf{v}_{1:t}$ последовательность $\mathbf{v}_1, \ldots, \mathbf{v}_t$. Взяв математическое ожидание от (14.2), получим

$$\mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}[f(\bar{\mathbf{w}}) - f(\mathbf{w}^*)] \le \mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}\left[\frac{1}{T}\sum_{t=1}^{T}(f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*))\right].$$

Поскольку лемма 14.1 справедлива для любой последовательности $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_T$, она применима и к СГС. Взяв математическое ожидание от границы в лемме, получим

$$\mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}\left[\frac{1}{T}\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle\right] \le \frac{B\rho}{\sqrt{T}}. \tag{14.9}$$

Осталось показать, что

$$\mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}\left[\frac{1}{T}\sum_{t=1}^{T}(f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*))\right] \le \mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}\left[\frac{1}{T}\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle\right], \tag{14.10}$$

что мы сейчас и сделаем.

В силу линейности математического ожидания имеем

$$\mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}\left[\frac{1}{T}\sum_{t=1}^{T}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle\right] = \frac{1}{T}\sum_{t=1}^{T}\mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}[\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle].$$

Теперь напомним *формулу полного математического ожидания*: для любых двух случайных величин α, β и функции g, $\mathbb{E}_\alpha[g(\alpha)] = \mathbb{E}_\beta \mathbb{E}_\alpha[g(\alpha)|\beta]$. Полагая $\alpha = \mathbf{v}_{1:t}$ и $\beta = \mathbf{v}_{1:t-1}$, получаем, что

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}[\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle] &= \mathop{\mathbb{E}}_{\mathbf{v}_{1:t}}[\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle] \\ &= \mathop{\mathbb{E}}_{\mathbf{v}_{1:t-1}}\mathop{\mathbb{E}}_{\mathbf{v}_{1:t}}[\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle | \mathbf{v}_{1:t-1}].\end{aligned}$$

Раз известно $\mathbf{v}_{1:t-1}$, то значение $\mathbf{w}^{(t)}$ больше не является случайным, и потому

$$\mathop{\mathbb{E}}_{\mathbf{v}_{1:t-1}}\mathop{\mathbb{E}}_{\mathbf{v}_{1:t}}[\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle | \mathbf{v}_{1:t-1}] = \mathop{\mathbb{E}}_{\mathbf{v}_{1:t-1}}\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathop{\mathbb{E}}_{\mathbf{v}_t}[\mathbf{v}_t | \mathbf{v}_{1:t-1}]\rangle.$$

Поскольку $\mathbf{w}^{(t)}$ зависит только от $\mathbf{v}_{1:t-1}$ и СГС требует, чтобы $\mathbb{E}_{\mathbf{v}_t}[\mathbf{v}_t | \mathbf{w}^{(t)}] \in \partial f(\mathbf{w}^{(t)})$, то $\mathbb{E}_{\mathbf{v}_t}[\mathbf{v}_t | \mathbf{v}_{1:t-1}] \in \partial f(\mathbf{w}^{(t)})$. Таким образом,

$$\mathop{\mathbb{E}}_{\mathbf{v}_{1:t-1}} \langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathop{\mathbb{E}}_{\mathbf{v}_t}[\mathbf{v}_t | \mathbf{v}_{1:t-1}] \rangle \geq \mathop{\mathbb{E}}_{\mathbf{v}_{1:t-1}} [f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*)].$$

Итак, мы показали, что

$$\begin{aligned} \mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}[\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \rangle] &\geq \mathop{\mathbb{E}}_{\mathbf{v}_{1:t-1}} [f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*)] \\ &= \mathop{\mathbb{E}}_{\mathbf{v}_{1:T}}[f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*)]. \end{aligned}$$

Просуммировав по t, разделив на T и воспользовавшись линейностью математического ожидания, получаем, что имеет место (14.10), что и требовалось доказать. □

14.4. Варианты

В этом разделе мы опишем несколько вариантов стохастического градиентного спуска.

14.4.1. Добавление шага проецирования

В проведенном выше анализе алгоритмов ГС и СГС мы требовали, чтобы норма $\mathbf{w}^*$ не превосходила B, т. е. чтобы $\mathbf{w}^*$ принадлежал множеству $\mathcal{H} = \{\mathbf{w}: ||\mathbf{w}|| \leq B\}$. В терминах обучения это означает, что мы довольствуемся B-ограниченным классом гипотез. И тем не менее любой шаг в направлении антиградиента (или в случайном направлении, в среднем совпадающим с направлением антиградиента) может вывести за пределы этого множества, и даже нет гарантии, что $\bar{\mathbf{w}}$ удовлетворяет этому условию. Ниже мы покажем, как преодолеть эту проблему, не уменьшая скорость сходимости.

Основная идея заключается в том, чтобы добавить шаг проецирования, т. е. теперь правило обновления состоит из двух шагов: сначала мы вычитаем субградиент из текущего значения $\mathbf{w}$, а затем проецируем результирующий вектор на $\mathcal{H}$. Формально:

1) $\mathbf{w}^{(t+1/2)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$;

2) $\mathbf{w}^{(t+1)} = \operatorname{argmin}_{\mathbf{w} \in \mathcal{H}} ||\mathbf{w} - \mathbf{w}^{(t+1/2)}||$.

На шаге проецирования текущее значение $\mathbf{w}$ заменяется ближайшим к нему вектором из $\mathcal{H}$. Очевидно, чтопроецирование гарантирует, что $\mathbf{w}^{(t)} \in \mathcal{H}$ для всех t. Поскольку $\mathcal{H}$ выпукло, то отсюда также следует, что $\bar{\mathbf{w}} \in \mathcal{H}$, как и должно быть. Далее мы покажем, что анализ алгоритма СГС с проецированием остается неизменным. Для этого нам понадобится следующая лемма.

Лемма 14.9 (лемма о проецировании). Пусть $\mathcal{H}$ – *замкнутое выпуклое множество, и* $\mathbf{v}$ – *проекция* $\mathbf{w}$ *на* $\mathcal{H}$, *т. е.*

$$\mathbf{v} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{H}} ||\mathbf{x} - \mathbf{w}||^2.$$

Тогда для любого $\mathbf{u} \in \mathcal{H}$,

$$||\mathbf{w} - \mathbf{u}||^2 - ||\mathbf{v} - \mathbf{u}||^2 \geq 0.$$

Доказательство. В силу выпуклости $\mathcal{H}$ для любого $\alpha \in (0, 1)$ имеем $\mathbf{v} + \alpha(\mathbf{u} - \mathbf{v}) \in \mathcal{H}$. Поэтому из оптимальности $\mathbf{v}$ получаем

$$\begin{aligned} ||\mathbf{v} - \mathbf{w}||^2 &\leq ||\mathbf{v} + \alpha(\mathbf{u} - \mathbf{v}) - \mathbf{w}||^2 \\ &= ||\mathbf{v} - \mathbf{w}||^2 + 2\alpha\langle \mathbf{v} - \mathbf{w}, \mathbf{u} - \mathbf{v}\rangle + \alpha^2||\mathbf{u} - \mathbf{v}||^2. \end{aligned}$$

После перегруппировки членов это неравенство принимает вид

$$2\langle \mathbf{v} - \mathbf{w}, \mathbf{u} - \mathbf{v}\rangle \geq -\alpha||\mathbf{u} - \mathbf{v}||^2.$$

Устремив α к 0, получаем, что

$$\langle \mathbf{v} - \mathbf{w}, \mathbf{u} - \mathbf{v}\rangle \geq 0.$$

Отсюда

$$\begin{aligned} ||\mathbf{w} - \mathbf{u}||^2 &= ||\mathbf{w} - \mathbf{v} + \mathbf{v} - \mathbf{u}||^2 \\ &= ||\mathbf{w} - \mathbf{v}||^2 + ||\mathbf{v} - \mathbf{u}||^2 + 2\langle \mathbf{v} - \mathbf{w}, \mathbf{u} - \mathbf{v}\rangle \\ &\geq ||\mathbf{v} - \mathbf{u}||^2. \end{aligned}$$

□

Вооружившись этой леммой, мы легко можем адаптировать анализ СГС к случаю, когда добавлено проецирование на замкнутое выпуклое множество. Просто отметим, что для любого t

$$\begin{aligned} &||\mathbf{w}^{(t+1)} - \mathbf{w}^*||^2 - ||\mathbf{w}^{(t)} - \mathbf{w}^*||^2 \\ &\quad = ||\mathbf{w}^{(t+1)} - \mathbf{w}^*||^2 - ||\mathbf{w}^{(t+1/2)} - \mathbf{w}^*||^2 + ||\mathbf{w}^{(t+1/2)} - \mathbf{w}^*||^2 - ||\mathbf{w}^{(t)} - \mathbf{w}^*||^2 \\ &\quad \leq ||\mathbf{w}^{(t+1/2)} - \mathbf{w}^*||^2 - ||\mathbf{w}^{(t)} - \mathbf{w}^*||^2. \end{aligned}$$

Поэтому лемма 14.1 справедлива и при добавлении шагов проецирования, а значит, и весь остальной анализ сохраняет силу.

14.4.2. Переменный размер шага

Еще в одном варианте СГС размер шага является убывающей функцией от t. То есть параметр η уже не константа, а переменная величина η_t. Например, можно положить $\eta_t = B/(\rho\sqrt{t})$ и получить границу, похожую на ту, что приведена в теореме 14.8. Идея в том, чтобы по мере приближения к минимуму функции выбирать размер шага осторожнее, стараясь не «проскочить» минимум.

14.4.3. Другие способы усреднения

Мы взяли выходной вектор $\bar{\mathbf{w}} = (1/T)\sum_{t=1}^{T}\mathbf{w}^{(t)}$. Существуют альтернативные подходы, например, вывод $\mathbf{w}^{(t)}$ для некоторого случайного $t \in [t]$ или вывод результата усреднения $\mathbf{w}^{(t)}$ по последним αT итерациямдля некоторого $\alpha \in (0, 1)$. Можно также брать взвешенное среднее по нескольким последним итерациям. Такие хитроумные схемы усреднения могут улучшить скорость сходимости в некоторых ситуациях, например, в случае строго выпуклых функций, рассматриваемых ниже.

*14.4.4. Строго выпуклые функции**

В этом разделе мы приведем вариант СГС, обладающий более высокой скоростью сходимости в проблемах со строго выпуклой целевой функцией (см. определение 13.4 строгой выпуклости в предыдущей главе). Мы опираемся на следующее утверждение, обобщающее лемму 13.5.

Утверждение 14.10. *Если f – λ-строго выпуклая функция, то для любых* $\mathbf{w}$, $\mathbf{u}$ *и* $\mathbf{v} \in \partial f(\mathbf{w})$ *имеет место неравенство*

$$\langle \mathbf{w} - \mathbf{u}, \mathbf{v} \rangle \geq f(\mathbf{w}) - f(\mathbf{u}) + (\lambda/2)\|\mathbf{w} - \mathbf{u}\|^2.$$

Доказательство похоже на доказательство леммы 13.5. Мы оставляем его читателю в качестве упражнения.

СГС для минимизации λ-строго выпуклой функции

цель: найти $\min_{\mathbf{w} \in \mathcal{H}} f(\mathbf{w})$
параметр: T
инициализация: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, \ldots, T$
  выбрать случайный вектор $\mathbf{v}_t$ такой, что $\mathbb{E}[\mathbf{v}_t | \mathbf{w}^{(t)}] \in \partial f(\mathbf{w}^{(t)})$
  положить $\eta_t = 1/(\lambda t)$
  положить $\mathbf{w}^{(t+1/2)} = \mathbf{w}^{(t)} - \eta_t \mathbf{v}_t$
  положить $\mathbf{w}^{(t+1)} = \operatorname{argmin}_{\mathbf{w} \in \mathcal{H}} \|\mathbf{w} - \mathbf{w}^{(t+1/2)}\|^2$
выход: $\bar{\mathbf{w}} = (1/T)\sum_{t=1}^{T} \mathbf{w}^{(t)}$

Теорема 14.11. *Предположим, что f λ-строго выпукла и* $\mathbb{E}[\|\mathbf{v}_t\|^2] \leq \rho^2$. *Пусть* $\mathbf{w}^* \in \operatorname{argmin}_{\mathbf{w} \in \mathcal{H}} f(\mathbf{w})$ *– оптимальное решение. Тогда*

$$\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{w}^*) \leq \frac{\rho^2}{2\lambda T}(1 + \log(T)).$$

Доказательство. Обозначим $\nabla^{(t)} = \mathbb{E}[\mathbf{v}_t | \mathbf{w}^{(t)}]$. Поскольку f строго выпукла и $\nabla^{(t)}$ принадлежит множеству субградиентов f в точке $\mathbf{w}^{(t)}$, то

$$\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \nabla^{(t)} \rangle \geq f(\mathbf{w}^{(t)}) - f(\mathbf{w}^*) + (\lambda/2)\|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2. \tag{14.11}$$

Далее мы покажем, что

$$\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \nabla^{(t)} \rangle \leq \frac{\mathbb{E}[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2]}{2\eta_t} + \frac{\eta_t}{2}\rho^2. \tag{14.12}$$

Поскольку $\mathbf{w}^{(t+1)}$ – проекция $\mathbf{w}^{(t+1/2)}$ на $\mathcal{H}$ и $\mathbf{w}^* \in \mathcal{H}$, то $\|\mathbf{w}^{(t+1/2)} - \mathbf{w}^*\|^2 \geq \|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2$. Поэтому

$$\begin{aligned}\|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 &\geq \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t+1/2)} - \mathbf{w}^*\|^2 \\ &= 2\eta_t \langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \rangle - \eta_t^2 \|\mathbf{v}_t\|^2.\end{aligned}$$

Если взять математическое ожидание от обеих частей, перегруппировать члены и воспользоваться предположением $\mathbb{E}[\|\mathbf{v}_t\|^2] \leq \rho^2$, то получим неравенство (14.12). Сопоставляя (14.11) и (14.12) и производя суммирование по t, получаем

$$\sum_{t=1}^{T}(\mathbb{E}[f(\mathbf{w}^{(t)})]-f(\mathbf{w}^{*})) \leq \mathbb{E}\left[\sum_{t=1}^{T}\left(\frac{\|\mathbf{w}^{(t)}-\mathbf{w}^{*}\|^2-\|\mathbf{w}^{(t+1)}-\mathbf{w}^{*}\|^2}{2\eta_t}+\frac{\lambda}{2}\|\mathbf{w}^{(t)}-\mathbf{w}^{*}\|^2\right)\right]+\frac{\rho^2}{2}\sum_{t=1}^{T}\eta_t.$$

Далее воспользуемся определением $\eta_t = 1/(\lambda t)$ и заметим, что первая сумма в правой части этого неравенства упрощается до $-\lambda T\|\mathbf{w}^{(T+1)} - \mathbf{w}^{*}\|^2 \leq 0$. Таким образом

$$\sum_{t=1}^{T}(\mathbb{E}[f(\mathbf{w}^{(t)})]-f(\mathbf{w}^{*})) \leq \frac{\rho^2}{2\lambda}\sum_{t=1}^{T}\frac{1}{t} \leq \frac{\rho^2}{2\lambda}(1+\log(T)).$$

Теорема следует из предыдущей, если разделить на T и воспользоваться неравенством Йенсена. □

Замечание 14.3. В работе Rakhlin, Shamir and Sridharan (2012) доказана скорость сходимости без члена $\log(T)$ для варианта алгоритма, в котором выходом является среднее по последним $T/2$ итерациям, $\bar{\mathbf{w}} = (2/T)\sum_{t=T/2+1}^{T}\mathbf{w}^{(t)}$. В работе Shamir and Zhang (2013) показано, что теорема 14.11 остается справедливой, даже если $\bar{\mathbf{w}} = \mathbf{w}^{(T)}$.

14.5. Обучение с помощью СГС

До сих пор мы описали и проанализировали алгоритм СГС для выпуклых функций общего вида. Теперь поговорим о его применимости к задачам обучения.

14.5.1. Применение СГС для минимизации риска

Напомним, что при обучении мы сталкиваемся с проблемой минимизации функции риска

$$L_{\mathcal{D}}(\mathbf{w}) = \mathop{\mathbb{E}}_{z\sim\mathcal{D}}[\ell(\mathbf{w}, z)].$$

Мы видели метод минимизации эмпирического риска, когда минимум эмпирического риска $L_S(\mathbf{w})$ считается оценкой минимума $L_{\mathcal{D}}(\mathbf{w})$. СГС открывает возможность другого подхода, когда $L_{\mathcal{D}}(\mathbf{w})$ минимизируется непосредственно. Поскольку $\mathcal{D}$ неизвестно, мы не можем просто вычислить градиент $\nabla L_{\mathcal{D}}(\mathbf{w}^{(t)})$ и минимизировать его методом ГС. Но в случае СГС нам только и нужно найти несмещенную оценку градиента $L_{\mathcal{D}}(\mathbf{w})$, т. е. случайный вектор с условным математическим ожиданием $\nabla L_{\mathcal{D}}(\mathbf{w}^{(t)})$. Сейчас мы покажем, как легко можно построить такую оценку.

Для простоты сначала рассмотрим случай дифференцируемой функции потери. Это значит, что и функция риска $L_{\mathcal{D}}$ тоже дифференцируема. Случайный вектор $\mathbf{v}_t$ будем строить следующим образом. Сначала выберем $z \sim \mathcal{D}$. Затем определим $\mathbf{v}_t$ как градиент функции $\ell(\mathbf{w}, z)$ по $\mathbf{w}$ в точке $\mathbf{w}^{(t)}$. Тогда в силу линейности градиента имеем

$$\mathbb{E}[\mathbf{v}_t \mid \mathbf{w}^{(t)}] = \mathop{\mathbb{E}}_{z\sim\mathcal{D}}[\nabla \ell(\mathbf{w}^{(t)}, z)] = \nabla \mathop{\mathbb{E}}_{z\sim\mathcal{D}}[\ell(\mathbf{w}^{(t)}, z)] = \nabla L_{\mathcal{D}}(\mathbf{w}^{(t)}). \tag{14.13}$$

Следовательно, градиент функции потери $\ell(\mathbf{w}, z)$ в точке $\mathbf{w}^{(t)}$ является несмещенной оценкой градиента функции риска $L_{\mathcal{D}}(\mathbf{w}^{(t)})$ и легко строится путем выборки одного нового примера $z \sim \mathcal{D}$ на каждой итерации t.

То же рассуждение проходит и для недифференцируемой функции потери. Просто мы полагаем $\mathbf{v}_t$ равным субградиенту $\ell(\mathbf{w}, z)$ в точке $\mathbf{w}^{(t)}$. Тогда для каждого $\mathbf{u}$ имеем

$$\ell(\mathbf{u}, z) - \ell(\mathbf{w}^{(t)}, z) \geq \langle \mathbf{u} - \mathbf{w}^{(t)}, \mathbf{v}_t \rangle.$$

Взяв математическое ожидание от обеих частей относительно $z \sim D$ и обусловив его значением $\mathbf{w}^{(t)}$, получаем

$$\begin{aligned} L_{\mathcal{D}}(\mathbf{u}) - L_{\mathcal{D}}(\mathbf{w}^{(t)}) &= \mathbb{E}[\ell(\mathbf{u}, z) - \ell(\mathbf{w}^{(t)}, z) \mid \mathbf{w}^{(t)}] \\ &\geq \mathbb{E}[\langle \mathbf{u} - \mathbf{w}^{(t)}, \mathbf{v}_t \rangle \mid \mathbf{w}^{(t)}] \\ &= \langle \mathbf{u} - \mathbf{w}^{(t)}, \mathbb{E}[\mathbf{v}_t \mid \mathbf{w}^{(t)}] \rangle. \end{aligned}$$

Отсюда следует, что $\mathbb{E}[\mathbf{v}_t \mid \mathbf{w}^{(t)}]$ является субградиентом $L_{\mathcal{D}}(\mathbf{w})$ в точке $\mathbf{w}^{(t)}$.

Итак, применение алгоритма стохастического градиентного спуска для минимизации риска выглядит следующим образом.

Применение СГС для минимизации $L_{\mathcal{D}}(\mathbf{w})$

параметры: скаляр $\eta > 0$, целое $T > 0$
инициализация: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \ldots, T$
  произвести выборку $z \sim \mathcal{D}$
  выбрать $\mathbf{v}_t \in \partial \ell(\mathbf{w}^{(t)}, z)$
  обновить $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
выход: $\bar{\mathbf{w}} = (1/T) \sum_{t=1}^{T} \mathbf{w}^{(t)}$

Теперь мы применим наш анализ СГС к анализу выборочной сложности обучения выпуклых-липшицевых-ограниченных проблем. Из теоремы 14.8 вытекает

Следствие 14.12. *Рассмотрим выпуклую–липшицеву–ограниченную проблему обучения с параметрами ρ, B. Тогда для любого $\epsilon > 0$, если выполнить T итераций (т. е. количество примеров) алгоритма СГС для минимизации $L_{\mathcal{D}}(\mathbf{w})$, где*

$$T \geq \frac{B^2 \rho^2}{\epsilon^2}$$

и $\eta = \sqrt{\dfrac{B^2}{\rho^2 T}}$, то результат его работы удовлетворяет условию

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})] \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \epsilon.$$

Интересно отметить, что требуемая выборочная сложность по порядку величины совпадает с гарантией выборочной сложности, которую мы вывели для минимизации регуляризированной потери. Выборочная сложность СГС даже лучше в 8 раз.

14.5.2. Анализ СГС для выпуклых–гладких проблем обучения

В предыдущей главе мы видели, что правило минимизации регуляризированной потери обучает также класс выпуклых–гладких–ограниченных проблем обучения. Теперь мы покажем, что и алгоритм СГС применим к таким проблемам.

Теорема 14.13. *Предположим, что для всех z функция потерь $\ell(\cdot, z)$ выпуклая, β-гладкая и неотрицательная. Тогда, если выполнить алгоритм СГС для минимизации $L_{\mathcal{D}}(\mathbf{w})$, то для любого $\mathbf{w}^*$ будем иметь*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})] \leq \frac{1}{1-\eta\beta}\left(L_{\mathcal{D}}(\mathbf{w}^*) + \frac{\|\mathbf{w}^*\|^2}{2\eta T}\right).$$

Доказательство. Напомним, что если функция β-гладкая и неотрицательная, то она самоограничена:

$$\|\nabla f(\mathbf{w})\|^2 \leq 2\beta f(\mathbf{w}).$$

Чтобы проанализировать СГС для выпуклых–гладких проблем, определим последовательность $z_1, \ldots, z_T$ случайно выбранных примеров в алгоритме СГС. Положим $f_t(\cdot) = \ell(\cdot, z_t)$ и заметим, что $\mathbf{v}_t = \nabla f_t(\mathbf{w}^{(t)})$. Для любого t функция f_t выпукла, и потому $f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*) \leq \langle \mathbf{v}_t, \mathbf{w}^{(t)} - \mathbf{w}^* \rangle$. Суммируя по t и применяя лемму 14.1, получаем

$$\sum_{t=1}^{T}(f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*)) \leq \sum_{t=1}^{T}\langle \mathbf{v}_t, \mathbf{w}^{(t)} - \mathbf{w}^* \rangle \leq \frac{\|\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2.$$

В сочетании с самоограниченностью f_t это дает

$$\sum_{t=1}^{T}(f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*)) \leq \frac{\|\mathbf{w}^*\|^2}{2\eta} + \eta\beta\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}).$$

Поделив на T и перегруппировав члены, получим

$$\frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) \leq \frac{1}{1-\eta\beta}\left(\frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{w}^*) + \frac{\|\mathbf{w}^*\|^2}{2\eta T}\right).$$

Далее возьмем от обеих частей последнего неравенства математическое ожидание по $z_1, \ldots, z_T$. Очевидно, что $\mathbb{E}[f_t(\mathbf{w}^*)] = L_{\mathcal{D}}(\mathbf{w}^*)$. А применяя то же рассуждение, что при доказательстве теоремы 14.8, мы получаем, что

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)})\right] = \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} L_{\mathcal{D}}(\mathbf{w}^{(t)})\right] \geq \mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})].$$

Объединив все вместе, мы завершим доказательство. □

Отсюда непосредственно вытекает

Следствие 14.14. *Рассмотрим выпуклую–гладкую–ограниченную проблему обучения с параметрами β, B. Дополнительно предположим, что $\ell(\mathbf{0}, z) \le 1$ для всех $z \in Z$. Для любого $\epsilon > 0$ положим $\eta = 1/(\beta(1+3/\epsilon))$. Тогда выполнение СГС с $T \ge 12B^2\beta/\epsilon^2$ дает*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})] \le \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \epsilon.$$

14.5.3. Применение СГС для минимизации регуляризированной потери

Мы показали, что выборочная сложность СГС в худшем случае такая же, как при минимизации регуляризированной потери. Но на некоторых распределениях минимизация регуляризированной потери может дать лучшее решение. Поэтому в некоторых случаях возникает желание решить задачу оптимизации, возникающую в связи с минимизацией регуляризированной потери, а именно[1]

$$\min_{\mathbf{w}} \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + L_S(\mathbf{w}) \right). \tag{14.14}$$

Поскольку мы имеем дело с выпуклыми проблемами обучения, в которых функция потерь выпукла, эта проблема также является задачей выпуклой оптимизации, которую можно решить применением СГС, как мы увидим в этом разделе.

Определим $f(\mathbf{w}) = (\lambda/2)\|\mathbf{w}\|^2 + L_S(\mathbf{w})$. Заметим, что f – λ-сильно выпуклая функция, поэтому можно применить вариант СГС, описанный в разделе 14.4.4 (с $\mathcal{H} = \mathbb{R}^d$). Для этого нужно только каким-то образом построить несмещенную оценку субградиента f в точке $\mathbf{w}^{(t)}$. Это легко сделать, заметив, что если выбрать z из S случайным образом с равномерным распределением и взять $\mathbf{v}_t \in \partial\ell(\mathbf{w}^{(t)}, z)$, то математическое ожидание $\lambda\mathbf{w}^{(t)} + \mathbf{v}_t$ является субградиентом f в $\mathbf{w}^{(t)}$.

Чтобы проанализировать получившийся алгоритм, мы сначала перепишем правило обновления (в предположении, что $\mathcal{H} = \mathbb{R}^d$, и потому шаг проецирования не играет роли) следующим образом:

$$\begin{aligned}
\mathbf{w}^{(t+1)} &= \mathbf{w}^{(t)} - \frac{1}{\lambda t}(\lambda\mathbf{w}^{(t)} + \mathbf{v}_t) \\
&= \left(1 - \frac{1}{t}\right)\mathbf{w}^{(t)} - \frac{1}{\lambda t}\mathbf{v}_t \\
&= \frac{t-1}{t}\mathbf{w}^{(t)} - \frac{1}{\lambda t}\mathbf{v}_t \\
&= \frac{t-1}{t}\left(\frac{t-2}{t-1}\mathbf{w}^{(t-1)} - \frac{1}{\lambda(t-1)}\mathbf{v}_{t-1}\right) - \frac{1}{\lambda t}\mathbf{v}_t \\
&= -\frac{1}{\lambda t}\sum_{i=1}^{t}\mathbf{v}_i.
\end{aligned} \tag{14.15}$$

[1] Для удобства мы поделили λ на 2.

Если предположить, что функция потерь ρ-липшицева, то для всех t мы будем иметь $||\mathbf{v}_t|| \leq \rho$ и, значит, $||\lambda \mathbf{w}^{(t)}|| \leq \rho$, откуда следует, что

$$||\lambda \mathbf{w}^{(t)} + \mathbf{v}_t|| \leq 2\rho.$$

Поэтому в силу теоремы 14.11 после выполнения T итераций будет справедливо неравенство:

$$\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{w}^*) \leq \frac{4\rho^2}{\lambda T}(1 + \log(T)).$$

ЛАНЬ®

14.6. Резюме

Мы познакомились с алгоритмами градиентного спуска и стохастического градиентного спуска, а также с несколькими их вариантами. Мы проанализировали их скорость сходимости и вычислили, сколько итераций гарантирует, что математическое ожидание целевой функции будет не больше оптимальной целевой функции плюс ϵ. Но самое важное – мы показали, что СГС позволяет минимизировать функцию риска непосредственно. Для этого мы выбираем независимые и одинаково распределенные точки с распределением $\mathcal{D}$ и используем субградиент потери текущей гипотезы $\mathbf{w}^{(t)}$ в каждой такой точке как несмещенную оценку градиента (или субградиента) функции риска. Отсюда следует, что верхняя граница числа итераций является также верхней границей выборочной сложности. Наконец, мы показали, как применять метод СГС к проблеме минимизации регуляризированного риска. В последующих главах мы покажем, что это дает на удивление простые решатели некоторых задач оптимизации, связанных с минимизацией регуляризированного риска.

14.7. Библиографические сведения

СГС восходит к работе Robbins and Monro (1951). Этот метод особенно эффективен для решения крупномасштабных проблем машинного обучения. См. например, Murata, 1998; Le Cun, 2004; Zhang, 2004; Bottou & Bousquet, 2008; Shalev-Shwartz, Singer & Srebro, 2007; Shalev-Shwartz & Srebro, 2008. Он изучался, в частности, в контексте стохастической оптимизации. См. например, Nemirovski & Yudin, 1978; Nesterov & Nesterov, 2004; Nesterov, 2005; Nemirovski, Juditsky, Lan & Shapiro, 2009; Shapiro, Dentcheva & Ruszczyński, 2009.

Выведенная нами верхняя граница для строго выпуклых функций заимствована из работы Hazan, Agarwal, and Kale (2007). Как отмечалось выше, в работе Rakhlin, Shamir & Sridharan (2012) были получены улучшенные оценки.

14.8. Упражнения

14.1. Докажите утверждение 14.10.

Указание. Обобщите доказательство леммы 13.5.

14.2. Докажите следствие 14.14.

14.3. Перцептрон как алгоритм субградиентного спуска. Пусть $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)) \in (\mathbb{R}^d \times \{\pm 1\})^m$. Предположим, что существует вектор $\mathbf{w} \in \mathbb{R}^d$ такой, что для любого $i \in [m]$ имеет место $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1$, и пусть $\mathbf{w}^*$ – вектор с минимальной нормой среди всех таких векторов. Обозначим $R = \max_i \|\mathbf{x}_i\|$. Определим функцию

$$f(\mathbf{w}) = \max_{i \in [m]} (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle).$$

- Покажите, что $\min_{\mathbf{w}: \|\mathbf{w}\| \leq \|\mathbf{w}^*\|} f(\mathbf{w}) = 0$ и что любой вектор $\mathbf{w}$, для которого $f(\mathbf{w}) < 1$, разделяет примеры, принадлежащие S.
- Покажите, как вычислить субградиент f.
- Опишите и проанализируйте алгоритм субградиентного спуска для этого случая. Сравните сам алгоритм и его анализ с алгоритмом пакетного перцептрона, который был описан в разделе 9.1.2.

14.4. Переменный размер шага (*). Докажите аналог теоремы 14.8 для варианта СГС с переменным размером шага $\eta_t = B/(\rho\sqrt{t})$.

Глава 15

МЕТОД ОПОРНЫХ ВЕКТОРОВ

ЛАНЬ®

В этой и следующей главах мы обсудим весьма полезный инструмент машинного обучения: парадигму опорных векторов (support vector machine – SVM) для обучения линейных предикторов в многомерных пространствах признаков. Высокая размерность этого пространства влечет за собой проблемы с точки зрения как выборочной, так и вычислительной сложности.

В алгоритме SVM проблему выборочной сложности решают, пытаясь найти разделители с «большим зазором». Грубо говоря, полупространство разбивает обучающий набор с большим зазором, если все примеры не только находятся по правильную сторону от разделяющей гиперплоскости, но и достаточно далеко от нее. Требуя, чтобы алгоритм находил только разделитель с большим зазором, мы можем сильно уменьшить вычислительную сложность, даже если размерность пространства признаков высока (а то и бесконечна). Мы введем концепцию зазора и сопоставим ее с парадигмой минимизации регуляризированной потери, а также со скоростью сходимости алгоритма перцептрона.

В следующей главе мы подойдем к проблеме вычислительной сложности, воспользовавшись идеей ядра.

15.1. Зазор и SVM с жестким зазором

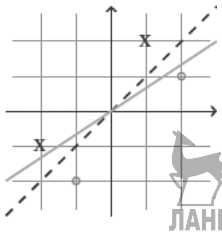
Пусть $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ – обучающий набор, в котором каждый пример $\mathbf{x}_i \in \mathbb{R}^d$ и $y_i \in \{\pm 1\}$. Говорят, что этот набор линейно разделим, если существует полупространство $(\mathbf{w}, b)$ такое, что $y_i = \text{sign}(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$ для всех i. По-другому это условие можно переписать в виде

$$\forall i \in [m], \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) > 0.$$

Все полупространства $(\mathbf{w}, b)$, удовлетворяющие этому условию, являются ERM-гипотезами (их бинарная ошибка равна нулю, т. е. является минимально возможной). Для любого разделимого обучающего набора существует много ERM-полупространств. Какое из них должен выбрать обучаемый?

Рассмотрим, к примеру, обучающий набор, представленный на рисунке ниже.

ЛАНЬ®



Четыре примера разделяются и сплошной, и штриховой гиперплоскостью, но интуитивно мы предпочли бы штриховую. Формализовать это предпочтение можно, введя понятие *зазора*.

Зазором гиперплоскости относительно обучающего набора называется минимальное расстояние от гиперплоскости до точки, принадлежащей набору. Если у гиперплоскости большой зазор, то она будет разделять обучающий набор, даже при малом шевелении каждого образца.

Ниже мы увидим, что истинную ошибку полупространства можно ограничить сверху в терминах ее зазора относительно обучающей выборки (чем больше зазор, тем меньше ошибка) вне зависимости от размерности объемлющего евклидова пространства.

SVM с жестким зазором (*Hard-SVM*) – это правило обучения, при котором мы возвращаем ERM-гиперплоскость, которая разделяет обучающий набор и при этом имеет максимально возможный зазор. Чтобы формально определить правило Hard-SVM, мы сначала выразим расстояние от точки $\mathbf{x}$ до гиперплоскости через параметры, определяющие полупространство.

Утверждение 15.1. *Расстояние между точкой* $\mathbf{x}$ *и гиперплоскостью, определенной параметрами* $(\mathbf{w}, b)$, *где* $\|\mathbf{w}\| = 1$, *равно* $|\langle \mathbf{w}, \mathbf{x} \rangle + b|$.

Доказательство. Расстояние между точкой $\mathbf{x}$ и гиперплоскостью, по определению, равно

$$\min\{\|\mathbf{x} - \mathbf{v}\| : \langle \mathbf{w}, \mathbf{v} \rangle + b = 0\}.$$

Если взять $\mathbf{v} = \mathbf{x} - (\langle \mathbf{w}, \mathbf{x} \rangle + b)\mathbf{w}$, то будем иметь

$$\langle \mathbf{w}, \mathbf{v} \rangle + b = \langle \mathbf{w}, \mathbf{x} \rangle - (\langle \mathbf{w}, \mathbf{x} \rangle + b)\|\mathbf{w}\|^2 + b = 0$$

и

$$\|\mathbf{x} - \mathbf{v}\| = |\langle \mathbf{w}, \mathbf{x} \rangle + b| \, \|\mathbf{w}\| = |\langle \mathbf{w}, \mathbf{x} \rangle + b|.$$

Следовательно, расстояние не превосходит $|\langle \mathbf{w}, \mathbf{x} \rangle + b|$. Далее, возьмем любую точку $\mathbf{u}$ на гиперплоскости, т. е. такую, что $\langle \mathbf{w}, \mathbf{u} \rangle + b = 0$. Имеем

$$\begin{aligned}
\|\mathbf{x} - \mathbf{u}\|^2 &= \|\mathbf{x} - \mathbf{v} + \mathbf{v} - \mathbf{u}\|^2 \\
&= \|\mathbf{x} - \mathbf{v}\|^2 + \|\mathbf{v} - \mathbf{u}\|^2 + 2\langle \mathbf{x} - \mathbf{v}, \mathbf{v} - \mathbf{u} \rangle \\
&\geq \|\mathbf{x} - \mathbf{v}\|^2 + 2\langle \mathbf{x} - \mathbf{v}, \mathbf{v} - \mathbf{u} \rangle \\
&= \|\mathbf{x} - \mathbf{v}\|^2 + 2(\langle \mathbf{w}, \mathbf{x} \rangle + b)\langle \mathbf{w}, \mathbf{v} - \mathbf{u} \rangle. \\
&= \|\mathbf{x} - \mathbf{v}\|^2,
\end{aligned}$$

где последнее неравенство справедливо, потому что $\langle \mathbf{w}, \mathbf{v} \rangle = \langle \mathbf{w}, \mathbf{u} \rangle = -b$. Таким образом, расстояние между $\mathbf{x}$ и $\mathbf{u}$ не меньше расстояния между $\mathbf{x}$ и $\mathbf{v}$, что и завершает доказательство. □

В силу этого утверждения, ближайшей к разделительной гиперплоскости точкой обучающего набора будет та, для которой достигается минимум $\min_{i \in [m]} |\langle \mathbf{w}, \mathbf{x}_i \rangle + b|$. Поэтому правило Hard-SVM записывается в виде

$$\arg\max_{(\mathbf{w}, b): \|\mathbf{w}\|=1} \min_{i \in [m]} |\langle \mathbf{w}, \mathbf{x}_i \rangle + b| \quad \text{s.t.} \quad \forall i, \; y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) > 0.$$

Если эта задача имеет решение (т. е. имеет место разделимая ситуация), то можно переписать ее в эквивалентном виде (см. упражнение 15.1):

$$\arg\max_{(\mathbf{w}, b): \|\mathbf{w}\|=1} \min_{i \in [m]} y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b). \tag{15.1}$$

Далее мы дадим еще одну эквивалентную формулировку правила Hard-SVM в виде задачи квадратичной оптимизации[1].

Hard-SVM

вход: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$

решить:

$$(\mathbf{w}_0, b_0) = \operatorname*{argmin}_{(\mathbf{w}, b)} \|\mathbf{w}\|^2 \text{ s.t. } \forall i, \; y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1. \tag{15.2}$$

выход: $\hat{\mathbf{w}} = \dfrac{\mathbf{w}_0}{\|\mathbf{w}_0\|}, \quad \hat{b} = \dfrac{b_0}{\|\mathbf{w}_0\|}$

Следующая лемма показывает, что результат работы SVM с жестким зазором действительно является разделительной гиперплоскостью с наибольшим зазором. Интуитивно понятно, что правило Hard-SVM ищет вектор $\mathbf{w}$ с минимальной нормой среди всех векторов, разделяющих данные, и таких, что $|\langle \mathbf{w}, \mathbf{x}_i \rangle + b| \geq 1$ для всех i. Иными словами, мы требуем, чтобы зазор был равен 1, только единицей измерения зазора является норма $\mathbf{w}$. Следовательно, нахождение полупространства с минимальным зазором сводится к нахождению $\mathbf{w}$ с минимальной нормой. Формально это можно выразить так:

Лемма 15.2. *Выход алгоритма Hard-SVM является решением задач (15.1).*

Доказательство. Пусть $(\mathbf{w}^*, b^*)$ – решение задачи (15.1). Обозначим зазор, достигаемый для полупространства $(\mathbf{w}^*, b^*)$, $\gamma^* = \min_{i \in [m]} y_i(\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*)$. Таким образом, для всех i имеем

$$y_i(\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) \geq \gamma^*$$

или, эквивалентно,

[1] Задачей квадратичной оптимизации называется задача оптимизации, в которой целевой функцией является выпуклая квадратичная функция, а ограничениями — линейные неравенства.

$$y_i\left(\left\langle \frac{\mathbf{w}^*}{\gamma^*}, \mathbf{x}_i \right\rangle + \frac{b^*}{\gamma^*}\right) \geq 1.$$

Следовательно, пара $\left(\frac{\mathbf{w}^*}{\gamma^*}, \frac{b^*}{\gamma^*}\right)$ удовлетворяет условиям задачи квадратичной оптимизации (15.2). Поэтому $\|\mathbf{w}_0\| \leq \left\|\frac{\mathbf{w}^*}{\gamma^*}\right\| = \frac{1}{\gamma^*}$. Отсюда следует, что для всех i

$$y_i(\langle \hat{\mathbf{w}}, \mathbf{x}_i \rangle + \hat{b}) = \frac{1}{\|\mathbf{w}_0\|} y_i(\langle \mathbf{w}_0, \mathbf{x}_i \rangle + b_0) \geq \frac{1}{\|\mathbf{w}_0\|} \geq \gamma^*.$$

Поскольку $\|\hat{\mathbf{w}}\| = 1$, получаем, что $(\hat{\mathbf{w}}, \hat{b})$ – оптимальное решение задачи (15.1). □

15.1.1. Однородный случай

Часто удобнее рассматривать однородные полупространства, которые проходят через начало координат и потому определены просто функцией $\text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$ без члена смещения b. Правило Hard-SVM для однородных полупространств сводится к решению задачи

$$\min_{\mathbf{w}} \|\mathbf{w}\|^2 \quad \text{s.t.} \quad \forall i,\ y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1. \tag{15.3}$$

Как отмечалось в главе 9, проблему обучения неоднородных полупространств можно свести к проблеме обучения однородных полупространств, добавив к каждому образцу $\mathbf{x}_i$ еще один признак и тем самым увеличив размерность до $d + 1$.

Заметим, однако, что задача оптимизации (15.2) не регуляризирует смещение b, а если мы обучаем однородное полупространство в $\mathbb{R}^{d+1}$ по правилу (15.3), то регуляризируем и член смещения (т. е. все $d + 1$ компонентов вектора весов). Впрочем, регуляризация b обычно не приводит к существенному увеличению выборочной сложности.

15.1.2. Выборочная сложность правила Hard-SVM

Напомним, что VC-размерность класса полупространств в $\mathbb{R}^d$ равна $d + 1$. Это значит, что выборочная сложность обучения полупространств растет вместе с размерностью задачи. Кроме того, согласно фундаментальной теореме обучения, если количество примеров значительно меньше d/ϵ, то никакой алгоритм не сможет обучить полупространство с верностью ϵ.

Чтобы справиться с этой проблемой, мы сделаем дополнительное предположение об истинном распределении данных. Точнее, введем предположение о «разделимости с зазором γ» и покажем, что если данные разделимы с зазором γ, то выборочная сложность ограничена сверху функцией от $1/\gamma^2$. Отсюда следует, что даже если размерность очень велика (или бесконечна), то при условии, что данные удовлетворяют предположению о разделимости с зазором, мы все

же сможем обойтись небольшой выборочной сложностью. Это не противоречит нижней границе в фундаментальной теореме обучения, поскольку мы сделали дополнительное предположение об истинном распределении данных.

Прежде чем формально определять предположение о разделимости с зазором, нам нужно разрешить вопрос с масштабом. Предположим, что обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ разделим с зазором γ, т. е. максимальное целевое значение (15.1) не меньше γ. Тогда для любого положительного скаляра $\alpha > 0$ обучающий набор $S' = (\alpha \mathbf{x}_1, y_1), \ldots, (\alpha \mathbf{x}_m, y_m)$ разделим с зазором $\alpha\gamma$. Следовательно, путем простого масштабирования данных мы можем сделать зазор сколь угодно большим. Поэтому, чтобы получить осмысленное определение зазора, нужно принять во внимание также масштаб примеров. В следующем определении эта идея выражена формально.

Определение 15.3. Пусть $\mathcal{D}$ – распределение на $\mathbb{R}^d \times \{\pm 1\}$. Говорят, что $\mathcal{D}$ разделимо с (γ, ρ)-зазором, если существует полупространство $(\mathbf{w}^*, b^*)$ такое, что $\|\mathbf{w}^*\| = 1$ и с вероятностью 1 для случайной выборки $(\mathbf{x}, y) \sim \mathcal{D}$ имеют место неравенства $y(\langle \mathbf{w}^*, \mathbf{x}^* \rangle + b^*) \geq \gamma$ и $\|\mathbf{x}\| \leq \rho$. Аналогично, говорят, что $\mathcal{D}$ разделимо однородным полупространством с (γ, ρ)-зазором, если все сказанное выше справедливо для полупространства вида $(\mathbf{w}^*, 0)$.

В главе 26 мы докажем, что выборочная сложность правила Hard-SVM зависит от $(\rho/\gamma)^2$ и не зависит от размерности d. В частности теорема 26.13 из раздела 26.3 утверждает следующее.

Теорема 15.4. *Пусть $\mathcal{D}$ – распределение на $\mathbb{R}^d \times \{\pm 1\}$, удовлетворяющее предположению о разделимости однородным полупространством с (γ, ρ)-зазором. Тогда с вероятностью не менее $1 - \delta$ для случайного обучающего набора размера m бинарная ошибка правила Hard-SVM не превосходит*

$$\sqrt{\frac{4(\rho/\gamma)^2}{m}} + \sqrt{\frac{2\log(2/\delta)}{m}}.$$

Замечание 15.1 (зазор и перцептрон). В разделе 9.1.2 мы описали и проанализировали алгоритм перцептрона для поиска ERM-гипотезы относительно класса полупространств. В частности, в теореме 9.1 мы ограничили сверху количество обновлений, которые перцептрон может произвести на данном обучающем наборе. Можно показать (см. упражнение 15.2), что точная верхняя граница равна $(\rho/\gamma)^2$, где ρ – радиус примеров, а γ – зазор.

15.2. SVM с мягким зазором и регуляризация по норме

Формулировка правила Hard-SVM предполагает, что обучающий набор линейно разделим, а это довольно сильное предположение. Правило SVM с мягким зазором, Soft-SVM, можно рассматривать как ослабление Hard-SVM, применимое даже к линейно неразделимым наборам. Задача оптимизации (15.2) налагает жесткие ограничения $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$ для всех i. Естественное ослабление – допустить нарушение этого ограничения для некоторых обучающих примеров. Это

можно смоделировать, введя неотрицательные переменные невязки $\xi_1, \ldots, \xi_m$ и заменив каждое ограничение $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$ ограничением $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$. То есть ξ_i измеряет, насколько сильно нарушается ограничение $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$. Правило Soft-SVM совместно минимизирует норму $\mathbf{w}$ (соответствует зазору) и среднюю величину ξ_i (соответствуют нарушениям ограничений). Компромисс между этими двумя членами управляется параметром λ. Таким образом, мы приходим к следующей задаче оптимизации Soft-SVM:

Soft-SVM

вход: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
решить:

$$\min_{\mathbf{w}, b, \xi} \left(\lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \xi_i \right) \tag{15.4}$$

при ограничениях $\forall i,\ y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$ и $\xi_i \geq 0$

выход: $\mathbf{w}$, b

Мы можем переписать (15.4) в виде задачи минимизации регуляризированной потери. Напомним, определение кусочно-линейной функции потерь:

$$\ell^{\text{hinge}}((\mathbf{w}, b), (\mathbf{x}, y)) = \max\{0, 1 - y(\langle \mathbf{w}, \mathbf{x} \rangle + b)\}.$$

Если даны полупространство $(\mathbf{w}, b)$ и обучающий набор S, то кусочно-линейная потеря, усредненная по S, обозначается $L_S^{\text{hinge}}((\mathbf{w}, b))$. Теперь рассмотрим проблему минимизации регуляризированной потери:

$$\min_{\mathbf{w}, b} \left(\lambda ||\mathbf{w}||^2 + L_S^{\text{hinge}}((\mathbf{w}, b)) \right). \tag{15.5}$$

Утверждение 15.5. *Задачи оптимизации (15.4) и (15.5) эквивалентны.*

Доказательство. Зафиксируем некоторые $\mathbf{w}$, b и рассмотрим минимизацию по ξ в задаче (15.4). Зафиксируем i. Поскольку ξ_i должно быть неотрицательно, то лучшим значением ξ_i будет 0, если $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$, и $1 - y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$ в противном случае Иными словами, $\xi_i = \ell^{\text{hinge}}((\mathbf{w}, b),(\mathbf{x}_i, y_i))$ для всех i, откуда следует доказываемое утверждение. □

Таким образом, мы видим, что правило Soft-SVM относится к парадигме минимизации регуляризированной потери, которую мы изучали в предыдущей главе. Алгоритм Soft-SVM, т. е. решение задачи (15.5), смещен в сторону разделителей с малой нормой. Целевая функция, которую мы стремимся минимизировать в задаче (15.5), штрафует не только за ошибки обучения, но и за большую норму.

Часто удобнее рассматривать правило Soft-SVM для обучения однородного полупространства, когда член смещения b равен нулю. Это дает нам следующую задачу оптимизации:

$$\min_{\mathbf{w}} \left(\lambda ||\mathbf{w}||^2 + L_S^{\text{hinge}}(\mathbf{w}) \right), \tag{15.6}$$

где

$$L_S^{\text{hinge}}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}\max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}_i\rangle\}.$$

15.2.1. Выборочная сложность Soft-SVM

Проанализируем теперь выборочную сложность правила Soft-SVM в случае однородных полупространств (т. е. решения задачи (15.6)). В следствии 13.8 мы вывели границу обобщаемости для проблем минимизации регуляризированной потери в предположении, что функция потерь выпуклая и липшицева. Мы уже показали, что кусочно-линейная потеря выпукла, так что осталось только установить ее липшицевость.

Утверждение 15.6. *Пусть* $f(\mathbf{w}) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}$. *Тогда* f *является* $\|\mathbf{x}\|$*-липшицевой.*

Доказательство. Легко проверить, что любой субградиент f в точке $\mathbf{w}$ имеет вид $\alpha\mathbf{x}$, где $|\alpha| \le 1$. Теперь утверждение следует из леммы 14.7. □

Таким образом, из следствия 13.8 вытекает

Следствие 15.7. *Пусть* $\mathcal{D}$ *– распределение на* $\mathcal{X} \times \{0, 1\}$, *где* $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\| \le \rho\}$. *Рассмотрим выполнение алгоритма Soft-SVM (задача (15.6)) на обучающем наборе* $S \sim \mathcal{D}^m$, *и пусть* $A(S)$ *– решение Soft-SVM. Тогда для любого* $\mathbf{u}$

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_S^{\text{hinge}}(A(S))] \le L_{\mathcal{D}}^{\text{hinge}}(\mathbf{u}) + \lambda\|\mathbf{u}\|^2 + \frac{2\rho^2}{\lambda m}.$$

Кроме того, поскольку кусочно-линейная потеря является верхней границей для бинарной потери, имеем также

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_S^{0-1}(A(S))] \le L_{\mathcal{D}}^{\text{hinge}}(\mathbf{u}) + \lambda\|\mathbf{u}\|^2 + \frac{2\rho^2}{\lambda m}.$$

Наконец, для любого $B > 0$, *если положить* $\lambda = \sqrt{\frac{2\rho^2}{B^2 m}}$, *то*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{0-1}(A(S))] \le \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{\text{hinge}}(A(S))] \le \min_{\mathbf{w}:\|\mathbf{w}\|\le B} L_{\mathcal{D}}^{\text{hinge}}(\mathbf{w}) + \sqrt{\frac{8\rho^2 B^2}{m}}.$$

Таким образом, мы видим, что можем контролировать выборочную сложность обучения полупространства как функцию нормы этого полупространства независимо от размерности объемлющего его евклидова пространства. Это особенно важно, когда обучение производится путем погружения в многомерное пространство признаков, как будет описано в следующей главе.

Замечание 15.2. Условие о том, что $\mathcal{X}$ состоит из векторов с ограниченной нормой, выполняется в силу требования о липшицевости функции потерь. Это не просто техническая деталь. Как было сказано выше, разделение с большим зазором бессмысленно, если не наложить ограничение на масштаб образцов. Дей-

ствительно, без такого ограничения мы всегда можем увеличить зазор, умножив все образцы на большой скаляр.

15.2.2. Сравнение границ, основанных на зазоре и норме, с размерностью

Выведенные нами границы для правил Hard-SVM и Soft-SVM не зависят от размерности пространства образцов. А зависят они от нормы примеров ρ, нормы полупространства B (или, эквивалентно, параметра зазора γ), а в неразделимом случае также от минимума кусочно-линейной потери на всех полупространствах с нормой $\leq B$. С другой стороны, VC-размерность класса однородных полупространств равна d, откуда следует, что ошибка ERM-гипотезы убывает как $\sqrt{d/m}$. Ниже мы приведем пример, в котором $\rho^2 B^2 \ll d$, а это значит, что граница из следствия 15.7 гораздо лучше VC-границы.

Рассмотрим проблему обучения, состоящую в классификации коротких текстовых документов по теме, например: относится документ к спорту или нет. Сначала нужно представить документы в виде векторов. Простой, но эффективный способ дает представление на основе *мешка слов*. Это значит, что мы определяем словарь и в качестве размерности d берем количество слов в нем. Документ представляется в виде вектора $\mathbf{x} \in \{0, 1\}^d$, где $x_i = 1$, если i-е слово словаря присутствует в документе, и $x_i = 0$ в противном случае. Следовательно, для этой проблемы ρ^2 равно максимальному количеству различных слов в документе.

Полупространство для этой проблемы назначает веса словам. Естественно предположить, что, назначив положительные и отрицательные веса нескольким десяткам слов, мы уже сможем с достаточной верностью определить, идет ли в документе речь о спорте или нет. Следовательно, в этой проблеме B^2 можно взять меньше 100. В итоге разумно считать, что значение $B^2\rho^2$ меньше 10 000.

С другой стороны, размер типичного словаря гораздо больше 10 000. Например, в английском языке более 100 000 слов. Это иллюстрирует общую проблему: обучение правила SVM и обучение полупространства с помощью обычного правила ERM могут различаться на порядок.

Разумеется, можно построить проблему, для которой граница SVM будет хуже границы VC. Использование SVM на самом деле вводит еще одну форму индуктивного смещения – мы отдаем предпочтение полупространствам с большим зазором. И, хотя это смещение может существенно уменьшить ошибку оценивания, оно способно также увеличить ошибку аппроксимации.

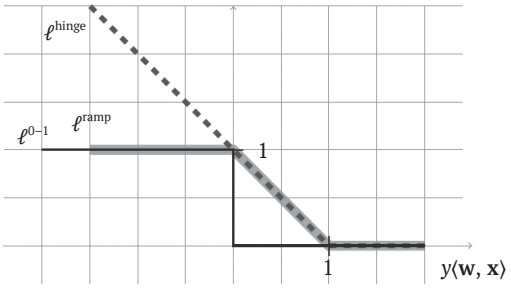
15.2.3. Рамповая функция потерь*

Выведенные в следствии 15.7 границы, основанные на зазоре, опираются на тот факт, что мы минимизируем кусочно-линейную потерю. В предыдущем разделе мы показали, что член $\sqrt{\rho^2 B^2/m}$ может быть намного меньше соответствующего члена в VC-границе – $\sqrt{d/m}$. Однако ошибка аппроксимации в следствии 15.7 измеряется относительно кусочно-линейной потери, а в VC-границах – относительно бинарной потери. Поскольку кусочно-линейная потеря является верхней границей для бинарной, ошибка аппроксимации относительно бинарной потери никогда не станет больше ошибки относительно кусочно-линейной потери.

Невозможно вывести верхнюю границу, в которую входит член ошибки оценивания $\sqrt{\rho^2 B^2/m}$ для бинарной потери. Это следует из того факта, что бинарная потеря нечувствительна к масштабу, а потому при измерении ошибки относительно бинарной потери норма $\mathbf{w}$ или ее зазор не имеют никакого смысла. Однако можно определить функцию потерь, которая, с одной стороны, чувствительна к масштабу и, стало быть, обладает ошибкой аппроксимации $\sqrt{\rho^2 B^2/m}$, а с другой стороны – больше походит на бинарную потерю. Одна из таких функций называется *рамповой потерей* (ramp loss) и определяется следующим образом:

$$\ell^{\text{ramp}}(\mathbf{w}, (\mathbf{x}, y)) = \min\{1, \ell^{\text{hinge}}(\mathbf{w}, (\mathbf{x}, y))\} = \min\{1, \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}\}.$$

Рамповая потеря штрафует за ошибки так же, как бинарная, и не штрафует примеры, разделенные с зазором. Разница между рамповой и бинарной потерей проявляется только на примерах, которые классифицированы правильно, но с недостаточным зазором. Обобщенные границы для рамповой функции потерь приведены в третьей части книги.



В SVM используется кусочно-линейная, а не рамповая функция потерь, потому что она выпукла и потому с *вычислительной* точки зрения ее можно минимизировать эффективно. Напротив, задача минимизации рамповой потери вычислительно неразрешима.

15.3. Условия оптимальности и «опорные векторы»*

Название «метод опорных векторов» проистекает из того факта, что решение SVM с жестким зазором, $\mathbf{w}_0$, опирается (т. е. является линейной оболочкой) на примеры, которые находятся точно на расстоянии $1/\|\mathbf{w}_0\|$ от разделительной гиперплоскости. Поэтому такие векторы называются *опорными*. Чтобы убедиться в этом, мы воспользуемся **условиями оптимальности Фрица Джона**.

Теорема 15.8. *Пусть $\mathbf{w}_0$ определено, как в (15.3), и обозначим $I = \{i : |\langle \mathbf{w}_0, \mathbf{x}_i\rangle| = 1\}$. Тогда существуют коэффициенты $\alpha_1, \ldots, \alpha_m$ такие, что*

$$\mathbf{w}_0 = \sum_{i \in I} \alpha_i \mathbf{x}_i .$$

Примеры $\{\mathbf{x}_i: i \in I\}$ называются *опорными векторами*.

Эта теорема доказывается применением следующей леммы к задаче (15.3).

Лемма 15.9 (Фрица Джона). *Предположим, что*

$$\mathbf{w}^* \in \operatorname{argmin}_{\mathbf{w}} f(\mathbf{w}) \quad \text{s.t.} \quad \forall i \in [m], g_i(\mathbf{w}) \le 0,$$

где $f, g_1, \ldots, g_m$ – дифференцируемые функции. Тогда существует вектор $\alpha \in \mathbb{R}^m$ такой, что $\nabla f(\mathbf{w}^*) + \sum_{i \in I} \alpha_i \nabla g_i(\mathbf{w}^*) = \mathbf{0}$, где $I = \{i : g_i(\mathbf{w}^*) = 0\}$.

15.4. Двойственность*

Исторически многие свойства SVM были получены путем рассмотрения задачи, *двойственной* к (15.3). В нашем изложении SVM двойственность не использовалась. Но для полноты картины мы покажем, как вывести уравнение, двойственное к (15.3).

Для начала перепишем задачу в следующей эквивалентной форме. Рассмотрим функцию

$$g(\mathbf{w}) = \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge 0} \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) = \begin{cases} 0, & \text{если } \forall i, y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \ge 1 \\ \infty, & \text{в противном случае} \end{cases}.$$

Тогда (15.3) можно переписать в виде

$$\min_{\mathbf{w}} (\|\mathbf{w}\|^2 + g(\mathbf{w})). \tag{15.7}$$

Перегруппируя члены, мы получаем, что задачу (15.3) можно переписать в виде

$$\min_{\mathbf{w}} \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge 0} \left(\frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \right). \tag{15.8}$$

Допустим теперь, что мы поменяли местами min и max в этом выражении. При этом целевая функция может только уменьшиться (см. упражнение 15.4), и мы получаем

$$\min_{\mathbf{w}} \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge 0} \left(\frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \right)$$
$$\ge \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge 0} \min_{\mathbf{w}} \left(\frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \right).$$

Это неравенство называется *слабой двойственностью*. Оказывается, что в нашем случае имеет место также *сильная двойственность*, т. е. неравенство обращается в равенство. Таким образом, *двойственная* задача имеет вид

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge 0} \min_{\mathbf{w}} \left(\frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \right). \tag{15.9}$$

Мы можем упростить двойственную задачу, заметив, что если α фиксирован, то мы имеем задачу оптимизации относительно $\mathbf{w}$ без ограничений, и целевая функция дифференцируема, поэтому в точке оптимума градиент обращается в нуль:

$$\mathbf{w} - \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i .$$

Это показывает, что решение должно принадлежать линейной оболочке примеров, – мы еще воспользуемся этим фактом при выводе SVM с ядрами. Подставляя это равенство в (15.9), получаем, что двойственную задачу можно переписать в виде

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \geq 0} \left(\frac{1}{2} \left\| \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \right\|^2 + \sum_{i=1}^{m} \alpha_i \left(1 - y_i \left\langle \sum_j \alpha_j y_j \mathbf{x}_j, \mathbf{x}_i \right\rangle \right) \right). \tag{15.10}$$

Перегруппировка членов дает такое представление двойственной задачи

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \geq 0} \left(\sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_j, \mathbf{x}_i \rangle \right). \tag{15.11}$$

Заметим, что в двойственной задаче встречаются только скалярные произведения образцов и не требуется прямой доступ к конкретным элементам образца. Это свойство важно при реализации SVM с ядрами (см. следующую главу).

15.5. Реализация Soft-SVM с помощью СГС

В этом разделе мы опишем очень простой алгоритм решения задачи оптимизации Soft-SVM, а именно

$$\min_{\mathbf{w}} \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \max\{0, 1 - y \langle \mathbf{w}, \mathbf{x}_i \rangle\} \right). \tag{15.12}$$

Мы опираемся на инфраструктуру СГС для решения задач минимизации регуляризированной потери, как описано в разделе 14.5.3.

Напомним, что на основе (14.15) мы можем переписать правило обновления СГС в виде

$$\mathbf{w}^{(t+1)} = -\frac{1}{\lambda t} \sum_{j=1}^{t} \mathbf{v}_j ,$$

где $\mathbf{v}_j$ – субградиент функции потерь в точке $\mathbf{w}^{(j)}$ на случайно выбранном примере на j-й итерации. В случае кусочно-линейной потери мы можем для данного примера $(\mathbf{x}, y)$ положить $\mathbf{v}_j = \mathbf{0}$, если $y\langle \mathbf{w}^{(j)}, \mathbf{x} \rangle \geq 1$, и $\mathbf{v}_j = -y\mathbf{x}$ в противном случае (см. пример 14.2). Обозначив $\theta^{(t)} = -\sum_{j<t} \mathbf{v}_j$, получаем следующую процедуру

Применение СГС для решения Soft-SVM

цель: решить задачу (15.12)
параметр: T
инициализация: $\theta^{(1)} = \mathbf{0}$
for $t = 1, \ldots, T$
  положить $\mathbf{w}^{(t)} = (1/\lambda t)\theta^{(t)}$
  случайно с равномерным распределением выбрать i из $[m]$
  ксли $(y_i\langle \mathbf{w}^{(t)}, \mathbf{x}_i\rangle < 1)$
    положить $\theta^{(t+1)} = \theta^{(t)} + y_i\mathbf{x}_i$
  иначе
    положить $\theta^{(t+1)} = \theta^{(t)}$
выход: $\bar{\mathbf{w}} = (1/T)\sum_{t=1}^{T}\mathbf{w}^{(t)}$

15.6. Резюме

SVM – алгоритм обучения полупространств с определенным типом априорного знания, а именно предпочтение отдается большому зазору. Алгоритм Hard-SVM ищет полупространство, которое идеально разделяет данные и имеет максимальный зазор, тогда как Soft-SVM не предполагает разделимости данных и позволяет до некоторой степени нарушать ограничения. Выборочная сложность обоих типов SVM отличается от выборочной сложности прямого обучения полупространств, поскольку зависит не от размерности области определения, а от таких параметров, как максимальные нормы $\mathbf{x}$ и $\mathbf{w}$.

Важность того факта, что выборочная сложность не зависит от размерности, будет в полной мере осознана в следующей главе, где мы обсудим погружение заданной области определения в некоторое многомерное пространство признаков как средство обогатить класс гипотез. Эта процедура поднимает проблемы, связанные с вычислительной и выборочной сложностью. Вторая решается применением SVM, а первая – применением SVM с ядрами: как именно, будет показано в следующей главе.

15.7. Библиографические сведения

Варианты SVM были впервые описаны в работах Cortes and Vapnik, 1992; Boser, Guyor and Vapnik, 1992. Есть немало хороших книг по теоретическим и практическим аспектам SVM, например, Vapnik, 1995; Cristianini & Shawe-Taylor, 2000; Schölkopf & Smola, 2002; Hsu et al., 2003; Steinwart and Christmann, 2008. Применение СГС для решения задачи SVM с мягким зазором предложено в работе Shalev-Shwartz et al. (2007).

15.8. Упражнения

15.1. Покажите, что правило Hard-SVM

$$\arg\max_{(\mathbf{w},b):\|\mathbf{w}\|=1} \min_{i\in[m]} |\langle \mathbf{w}, \mathbf{x}_i\rangle + b| \quad \text{s.t.} \quad \forall i,\ y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) > 0$$

можно переформулировать следующим образом:

$$\arg\max_{(\mathbf{w},b):\|\mathbf{w}\|=1} \min_{i\in[m]} y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b). \tag{15.13}$$

Указание. Определим $\mathcal{G} = \{(\mathbf{w}, b) : \forall i, y_i(\langle \mathbf{w}, x_i\rangle + b) > 0\}$.

1. Покажите, что

$$\arg\max_{(\mathbf{w},b):\|\mathbf{w}\|=1} \min_{i\in[m]} y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \in \mathcal{G}.$$

2. Покажите, что $\forall (\mathbf{w},b) \in \mathcal{G}$

$$\min_{i\in[m]} y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) = \min_{i\in[m]} |\langle \mathbf{w}, \mathbf{x}_i\rangle + b|.$$

15.2. Зазор и перцептрон. Рассмотрим обучающий набор, линейно разделимый с зазором γ и такой, что все образцы находятся в шаре радиуса ρ. Докажите, что при работе на таком обучающем наборе максимальное число обновлений в алгоритме пакетного перцептрона, приведенном в разделе 9.1.2, равно $(\rho/\gamma)^2$.

15.3. SVM с жестким и мягким зазором. Докажите или опровергните следующее утверждение:

существует такое $\lambda > 0$, *что для любой выборки* S, *содержащей* $m > 1$ *примеров, разделимой классом однородных полупространств, правила обучения Hard-SVM и Soft-SVM (с параметром* λ*) возвращают в точности одинаковые векторы весов.*

15.4. Слабая двойственность. Докажите, что для любой функции f от двух векторных переменных $\mathbf{x} \in \mathcal{X}$, $\mathbf{y} \in \mathcal{Y}$ справедливо неравенство

$$\min_{\mathbf{x}\in\mathcal{X}} \max_{\mathbf{y}\in\mathcal{Y}} f(\mathbf{x},\mathbf{y}) \geq \max_{\mathbf{y}\in\mathcal{Y}} \min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x},\mathbf{y}).$$

Глава 16

ЯДЕРНЫЕ МЕТОДЫ

В предыдущей главе мы описали парадигму SVM для обучения полупространств в многомерном пространстве признаков. Это позволяет повысить выразительную силу полупространств, сначала отобразив данные в многомерное пространство признаков, а затем обучив линейный предиктор в этом пространстве. Это напоминает алгоритм AdaBoost, который обучает композицию полупространства с базовыми гипотезами. Хотя этот подход значительно расширяет выразительность полупространств в качестве предикторов, он несет с собой проблемы в плане выборочной и вычислительной сложности. В предыдущей главе мы решили проблему выборочной сложности, воспользовавшись понятием зазора. В этой главе мы постараемся решить проблему вычислительной сложности с помощью *ядер*.

Мы начнем главу с описания идеи погружения данных в многомерное пространство признаков. Затем мы введем понятие ядра. Ядро – это вид меры сходства образцов. У ядерного сходства есть одно специальное свойство – его можно рассматривать как скалярное произведение в некотором гильбертовом пространстве (или евклидовом пространстве высокой размерности). Мы опишем «ядерный трюк», который позволяет реализовать обучение вычислительно эффективно, не прибегая к явной работе с многомерными представлениями образцов. Алгоритмы, основанные на ядрах, и, в частности, kernel-SVM, – это очень полезные и популярные средства машинного обучения. Своим успехом они обязаны как гибкости в части адаптации к конкретным априорным знаниям о предметной области, так и наличию тщательно разработанного набора эффективных алгоритмов реализации.

ЛАНЬ®

16.1. Погружение в пространство признаков

Выразительная способность полупространств во многом ограничена; например, следующий обучающий набор невозможно разделить полупространством.

В качестве области определения возьмем вещественную прямую. Рассмотрим образцы $\{-10, -9, -8, ..., 0, 1, ..., 9, 10\}$ и назначим метку $+1$ тем x, для которых $|x| > 2$, и -1 всем остальным.

Чтобы сделать класс полупространств более выразительным, мы можем сначала отобразить пространство образцов в другое пространство (возможно, более

высокой размерности), а затем обучить полупространство в этом пространстве. Например, возьмем приведенный выше пример. Вместо того чтобы обучать полупространство в исходном представлении, сначала определим отображение $\psi : \mathbb{R} \to \mathbb{R}^2$ следующим образом:

$$\psi(x) = (x, x^2).$$

Область значений ψ мы называем пространством признаков. После применения ψ данные можно легко объяснить с помощью полупространства $h(x) = \text{sign}(\langle \mathbf{w}, \psi(x) \rangle - b)$, где $\mathbf{w} = (0, 1)$ и $b = 5$.

Базовая парадигма формулируется следующим образом:

1) дана область образцов $\mathcal{X}$ и задача обучения. Выберем отображение $\psi : \mathcal{X} \to \mathcal{F}$, где $\mathcal{F}$ – некоторое *пространство признаков*, обычно $\mathbb{R}^n$ для некоторого n (однако областью значений такого отображения может быть произвольное *гильбертово пространство*, в т. ч. бесконечномерное, как мы покажем ниже);
2) по заданной последовательности помеченных примеров $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, создать последовательность образов $\hat{S} = (\psi(\mathbf{x}_1), y_1), \ldots, (\psi(\mathbf{x}_m), y_m)$;
3) обучить линейный предиктор h на $\hat{S}$;
4) в качестве метки тестовой точки $\mathbf{x}$ предсказать $h(\psi(\mathbf{x}))$.

Заметим, что для любого распределения вероятностей $\mathcal{D}$ на $\mathcal{X} \times \mathcal{Y}$ мы легко можем определить его образ $\mathcal{D}^\psi$ на $\mathcal{F} \times \mathcal{Y}$, положив для любого подмножества $A \subseteq \mathcal{F} \times \mathcal{Y}$, $\mathcal{D}^\psi(A) = \mathcal{D}(\psi^{-1}(A))$.[1] Отсюда следует, что для любого предиктора h на пространстве признаков $L_{\mathcal{D}^\psi}(h) = L_{\mathcal{D}}(h \circ \psi)$, где $h \circ \psi$ – композиция h и ψ.

Успех этой парадигмы обучения зависит от выбора хорошего отображения ψ для данной задачи обучения, т. е. такого ψ, при котором образ распределения данных является линейно разделимым в пространстве признаков (или близким к тому), так что результирующий алгоритм будет хорошим обучаемым для имеющейся задачи. Выбор такого погружения требует априорных знаний о задаче. Однако часто используются отображения общего вида, позволяющие обогатить класс полупространств и повысить его выразительность. Одно из них – полиномиальное отображение, обобщающее то ψ, которое мы видели в приведенном выше примере.

Напомним, что стандартный классификатор на базе полупространств предсказывает для образца $\mathbf{x}$ метку, основанную на линейном отображении $\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$. Мы можем сделать обобщение, заменив линейное отображение полиномиальным $\mathbf{x} \mapsto p(\mathbf{x})$, где p – полином от нескольких переменных степени k. Для простоты рассмотрим сначала случай, когда $\mathbf{x}$ – одномерный вектор. В таком случае $p(x) = \sum_{j=0}^{k} w_j x^j$, где $\mathbf{w} \in \mathbb{R}^{k+1}$ – вектор коэффициентов полинома, который мы должны обучить. Можно переписать $p(x)$ в виде $\langle \mathbf{w}, \psi(x) \rangle$, где $\psi : \mathbb{R} \to \mathbb{R}^{k+1}$ – отображение $x \mapsto (1, x, x^2, x^3, \ldots, x^k)$. Отсюда следует, что обучить полином степени k над $\mathbb{R}$ можно с помощью линейного отображения в $(k+1)$-мерное пространство признаков.

Вообще, полином степени k от нескольких переменных, отображающий $\mathbb{R}^n$ в $\mathbb{R}$, можно записать в виде

[1] Это отображение определено для любого A такого, что $\psi^{-1}(A)$ измеримо относительно $\mathcal{D}$.

$$p(\mathbf{x}) = \sum_{J \in [n]^r : r \le k} w_J \prod_{i=1}^{r} x_{J_i} . \tag{16.1}$$

Как и раньше, эту формулу можно переписать в виде $p(\mathbf{x}) = \langle \mathbf{w}, \psi(\mathbf{x}) \rangle$, только теперь $\psi : \mathbb{R}^n \to \mathbb{R}^d$ – такое отображение, что для любого $J \in [n]^r$, $r \le k$ координата $\psi(\mathbf{x})$, ассоциированная с J, – одночлен $\prod_{i=1}^{r} x_{J_i}$.

Естественно, полиномиальные классификаторы дают куда более богатые классы гипотез, чем полупространства. В начале этой главы мы видели пример, в котором обучающий пример нельзя было разделить полупространством в его исходной области определения ($\mathcal{X} = \mathbb{R}$), но это стало возможно после погружения $x \mapsto (x, x^2)$. Таким образом, классификатор, линейный в пространстве признаков, может вести себя совершенно нелинейно в исходном пространстве, из которого выбираются образцы.

В общем случае мы вольны выбирать любое отображение ψ исходных образцов в некоторое гильбертово пространство[1]. Евклидово пространство $\mathbb{R}^d$ является гильбертовым при любом конечном d. Но существуют также бесконечномерные гильбертовы пространства (как мы увидим ниже в этой главе).

Подытоживая это обсуждение, отметим, что мы можем обогатить класс полупространств, применив сначала нелинейное отображение ψ пространства образцов в некоторое пространство признаков, а затем обучив полупространство в пространстве признаков. Однако если область значений ψ – пространство высокой размерности, то возникают две проблемы. Во-первых, VC-размерность класса полупространств в $\mathbb{R}^n$ равна $n + 1$, а значит, если размерность области значений ψ очень велика, то для обучения в нем класса полупространств потребуется гораздо больше примеров. Во-вторых, выполнение вычислений в многомерном пространстве может обходиться слишком дорого. На самом деле, даже представить вектор $\mathbf{w}$ в пространстве признаков может оказаться нереально. Первую проблему можно решать с помощью парадигмы большого зазора (или предикторов с малой нормой), как обсуждалось в предыдущей главе в контексте алгоритма SVM. А в следующем разделе мы обсудим подходы к вычислительному аспекту.

16.2. Ядерный трюк

Мы видели, что погружение области образцов в многомерное пространство признаков делает обучение полупространства более выразительным. Но вычислительная сложность такого обучения может представлять серьезное пре-

[1] Гильбертовым пространством называется полное векторное пространство, в котором определено скалярное произведение. Пространство называется полным, если все последовательности Коши в нем сходятся. В нашем случае норма $\|\mathbf{w}\|$ определяется с помощью скалярного произведения как $\sqrt{\langle \mathbf{w}, \mathbf{w} \rangle}$. Требование, чтобы область значений ψ была гильбертовым пространством, связано с тем, что в гильбертовом пространстве корректно определена проекция. Точнее, если M — линейное подпространство гильбертова пространства, то любой вектор $\mathbf{x}$ этого гильбертова пространства можно представить в виде $\mathbf{x} = \mathbf{u} + \mathbf{v}$, где $\mathbf{u} \in M$ и $\langle \mathbf{v}, \mathbf{w} \rangle = 0$ для всех $\mathbf{w} \in M$. Мы воспользуемся этим фактом при доказательстве теоремы о представителе в следующем разделе.

пятствие – накладные расходы на вычисление линейных разделителей данных в пространстве очень высокой размерности могут оказаться непомерно велики. Эту проблему часто решают с помощью обучения на основе ядер. В этом контексте термин «ядра» обозначает скалярные произведения в пространстве признаков. Если ψ – погружение области определения $\mathcal{X}$ в некоторое гильбертово пространство, то мы можем определить ядерную функцию $K(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$. Можно считать, что K описывает сходство образцов, а ψ представлять себе как отображение области определения $\mathcal{X}$ в пространство, где это сходство реализуется в виде скалярного произведения. Оказывается, что многие алгоритмы обучения полупространств можно выразить, зная только значения ядерной функций от пары образцов. Основное достоинство таких алгоритмов состоит в том, что они реализуют линейные разделители в многомерном пространстве признаков, не нуждаясь ни в явном задании точек в этом пространстве, ни в явном выражении погружения ψ. Оставшуюся часть этого раздела мы посвятим построению таких алгоритмов.

В предыдущей главе мы видели, что регуляризация нормы $\mathbf{w}$ дает небольшую выборочную сложность даже при высокой размерности пространства признаков. Интересно – и ниже мы это докажем, – что регуляризация нормы $\mathbf{w}$ полезна также для преодоления вычислительных трудностей. Чтобы убедиться в этом, отметим сначала, что все варианты задачи оптимизации SVM, рассмотренные в предыдущей главе, являются частными случаями следующей общей задачи:

$$\min_{\mathbf{w}} (f(\langle \mathbf{w}, \psi(\mathbf{x}_1) \rangle, \ldots, \langle \mathbf{w}, \psi(\mathbf{x}_m) \rangle) + R(\|\mathbf{w}\|)), \tag{16.2}$$

где $f : \mathbb{R}^m \to \mathbb{R}$ – произвольная функция, а $R : \mathbb{R}_+ \to \mathbb{R}$ – монотонно неубывающая функция. Так, правило Soft-SVM для однородных полупространств (задача (15.6)) является частным случаем (16.2) при $R(a) = \lambda a^2$ и $f(a_1, \ldots, a_m) = (1/m)\sum_i \max\{0, 1 - y_i a_i\}$. Аналогично, правило Hard-SVM для неоднородных полупространств (задача (15.2)) является частным случаем (16.2) при $R(a) = a^2$ и $f(a_1, \ldots, a_m) = 0$, если существует такое b, что $y_i(a_i + b) \geq 1$ для всех i, и $f(a1, \ldots, am) = \infty$ в противном случае.

Следующая теорема показывает, что существует оптимальное решение задачи (16.2), принадлежащее линейной оболочке $\{\psi(\mathbf{x}_1), \ldots, \psi(\mathbf{x}_m)\}$.

Теорема 16.1 (теорема о представителе). *Предположим, что ψ – отображение из $\mathcal{X}$ в гильбертово пространство. Тогда существует вектор $\alpha \in \mathbb{R}^m$ такой, что $\mathbf{w} = \sum_{i=1}^m \alpha_i \psi(\mathbf{x}_i)$ – оптимальное решение задачи (16.2).*

Доказательство. Пусть $\mathbf{w}^*$ – оптимальное решение задачи(16.2). Поскольку $\mathbf{w}^*$ – элемент гильбертова пространства, мы можем записать $\mathbf{w}^*$ в виде

$$\mathbf{w}^* = \sum_{i=1}^m \alpha_i \psi(\mathbf{x}_i) + \mathbf{u},$$

где $\langle \mathbf{u}, \psi(\mathbf{x}_i) \rangle = 0$ для всех i. Положим $\mathbf{w} = \mathbf{w}^* - \mathbf{u}$. Очевидно, что $\|\mathbf{w}^*\|^2 = \|\mathbf{w}\|^2 + \|\mathbf{u}\|^2$, поэтому $\|\mathbf{w}\| \leq \|\mathbf{w}^*\|$. Так как функция R неубывающая, то $R(\|\mathbf{w}\|) \leq R(\|\mathbf{w}^*\|)$. Кроме того, для всех i имеем

$$y_i \langle \mathbf{w}, \psi(\mathbf{x}_i) \rangle = y_i \langle \mathbf{w}^* - \mathbf{u}, \psi(\mathbf{x}_i) \rangle = y_i \langle \mathbf{w}^*, \psi(\mathbf{x}_i) \rangle,$$

откуда

$$f(y_i\langle \mathbf{w}, \psi(\mathbf{x}_1)\rangle, \ldots, y_m\langle \mathbf{w}, \psi(\mathbf{x}_m)\rangle) = f(y_1\langle \mathbf{w}^*, \psi(\mathbf{x}_1)\rangle, \ldots, y_m\langle \mathbf{w}, \psi(\mathbf{x}_m)\rangle).$$

Мы показали, что целевая функция задачи (16.2) в точке $\mathbf{w}$ не может быть больше целевой функции в точке $\mathbf{w}^*$ и, следовательно, $\mathbf{w}$ также является оптимальным решением. Поскольку $\mathbf{w} = \sum_{i=1}^m \alpha_i \psi(\mathbf{x}_i)$, то наше доказательство завершено. □

Благодаря теореме о представителе мы можем следующим образом оптимизировать задачу (16.2) относительно всех коэффициентов α, а не коэффициентов $\mathbf{w}$. Записав $\mathbf{w} = \sum_{i=1}^m \alpha_j \psi(\mathbf{x}_j)$, мы получим, что для всех i

$$\langle \mathbf{w}, \psi(\mathbf{x}_i)\rangle = \left\langle \sum_j \alpha_j \psi(\mathbf{x}_j), \psi(\mathbf{x}_i) \right\rangle = \sum_{j=1}^m \alpha_j \langle \psi(\mathbf{x}_j), \psi(\mathbf{x}_i)\rangle.$$

Аналогично,

$$\|\mathbf{w}\|^2 = \left\langle \sum_j \alpha_j \psi(\mathbf{x}_j), \sum_j \alpha_j \psi(\mathbf{x}_j) \right\rangle = \sum_{i,j=1}^m \alpha_i \alpha_j \langle \psi(\mathbf{x}_i), \psi(\mathbf{x}_j)\rangle.$$

Пусть $K(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}')\rangle$ – функция, реализующая ядро относительно погружения ψ. Вместо задачи (16.2) мы можем решать эквивалентную задачу

$$\min_{\alpha \in \mathbb{R}^m} f\left(\sum_{j=1}^m \alpha_j K(\mathbf{x}_j, \mathbf{x}_1), \ldots, \sum_{j=1}^m \alpha_j K(\mathbf{x}_j, \mathbf{x}_m)\right) + R\left(\sqrt{\sum_{i,j=1}^m \alpha_i \alpha_j K(\mathbf{x}_j, \mathbf{x}_i)}\right). \tag{16.3}$$

Для решения задачи оптимизации (16.3) нам не нужен непосредственный доступ к элементам пространства признаков. Нужно лишь знать, как вычислять скалярное произведение в этом пространстве, или ядерную функцию. На самом деле, для решения задачи (16.3) нам нужно только знать значение матрицы G размера $m \times m$ такой, что $G_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$; ее часто называют *матрицей Грама*.

В частности, задачу Soft-SVM, описываемую формулой (15.6), можно переписать в виде

$$\min_{\alpha \in \mathbb{R}^m} \left(\lambda \boldsymbol{\alpha}^T G \boldsymbol{\alpha} + \frac{1}{m} \sum_{i=1}^m \max\{0, 1 - y_i (G\boldsymbol{\alpha})_i\} \right), \tag{16.4}$$

где $(G\boldsymbol{\alpha})_i$ – i-й элемент вектора, полученного умножением матрицы Грама G на вектор $\boldsymbol{\alpha}$. Заметим, что (16.4) можно записать в виде задачи квадратичного программирования, которая допускает эффективное решение. В следующем разделе мы опишем еще более простой алгоритм решения Soft-SVM с помощью ядер.

Обучив коэффициенты $\boldsymbol{\alpha}$, мы можем вычислить предсказание для нового образца:

$$\langle \mathbf{w}, \psi(\mathbf{x})\rangle = \sum_{j=1}^m \alpha_j \langle \psi(\mathbf{x}_j), \psi(\mathbf{x})\rangle = \sum_{j=1}^m \alpha_j K(\mathbf{x}_j, \mathbf{x}).$$

Преимущество работы с ядрами по сравнению с непосредственной оптимизацией $\mathbf{w}$ в пространстве признаков заключается в том, что в некоторых ситуациях размерность пространства признаков очень велика, тогда как реализовать ядерную функцию совсем просто. Ниже мы приводим несколько примеров.

Пример 16.1 (полиномиальные ядра). Полиномиальное ядро степени k определяется следующим образом:

$$K(\mathbf{x}, \mathbf{x}') = (1 + \langle \mathbf{x}, \mathbf{x}' \rangle)^k.$$

Покажем, что это действительно ядерная функция, т. е. существует отображение ψ исходного пространства в некоторое пространство большей размерности такое, что $K(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$. Для простоты обозначим $x_0 = x'_0 = 1$. Тогда имеем

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}') &= (1 + \langle \mathbf{x}, \mathbf{x}' \rangle)^k = (1 + \langle \mathbf{x}, \mathbf{x}' \rangle) \cdot \cdots \cdot (1 + \langle \mathbf{x}, \mathbf{x}' \rangle) \\ &= \left(\sum_{j=0}^{n} x_j x'_j \right) \cdot \cdots \cdot \left(\sum_{j=0}^{n} x_j x'_j \right) \\ &= \sum_{J \in \{0,1,\ldots,n\}^k} \prod_{i=1}^{k} x_{J_i} x'_{J_i} \\ &= \sum_{J \in \{0,1,\ldots,n\}^k} \prod_{i=1}^{k} x_{J_i} \prod_{i=1}^{k} x'_{J_i}. \end{aligned}$$

Теперь, если определить $\psi : \mathbb{R}^n \to \mathbb{R}^{(n+1)k}$ так, что для любого $J \in \{0, 1, \ldots, n\}^k$ существует элемент $\psi(\mathbf{x})$, равный $\prod_{i=1}^{k} x_{Ji}$, то получим, что

$$K(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle.$$

Поскольку ψ содержит все одночлены степени не выше k, то полупространство в области значений ψ соответствует полиномиальному предиктору степени k в исходном пространстве. Следовательно, обучение полупространства с помощью полиномиального ядра степени k позволяет обучать полиномиальные предикторы степени k в исходном пространстве.

Заметим, что здесь сложность реализации K равна $O(n)$, тогда как размерность пространства признаков имеет порядок n^k.

Пример 16.2 (гауссово ядро). Пусть исходное пространство образцов совпадает с $\mathbb{R}$ и рассмотрим отображение ψ такое, что для каждого неотрицательного целого числа $n \geq 0$ существует элемент $\psi(x)_n$, равный $\frac{1}{\sqrt{n!}} e^{-\frac{x^2}{2}} x^n$. Тогда

$$\begin{aligned} \langle \psi(x), \psi(x') \rangle &= \sum_{n=0}^{\infty} \left(\frac{1}{\sqrt{n!}} e^{-\frac{x^2}{2}} x^n \right) \left(\frac{1}{\sqrt{n!}} e^{-\frac{(x')^2}{2}} (x')^n \right) \\ &= e^{-\frac{x^2 + (x')^2}{2}} \sum_{n=0}^{\infty} \left(\frac{(xx')^n}{n!} \right) \\ &= e^{-\frac{\|x - x'\|^2}{2}}. \end{aligned}$$

Здесь пространство признаков бесконечномерно, но вычислить ядро очень просто. Вообще, если задан скаляр $\sigma > 0$, то гауссово ядро определяется следующим образом:

$$K(\mathbf{x}, \mathbf{x}') = e^{-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2\sigma}}.$$

Интуитивно понятно, что гауссово ядро определяет скалярное произведение $\mathbf{x}$ и $\mathbf{x}'$ в пространстве признаков таким образом, что оно близко к нулю, если образцы отстоят далеко друг от друга в исходном пространстве, и близко к 1, если образцы находятся рядом. Параметр σ управляет масштабом, определяющим, что мы понимаем под словом «близко». Легко проверить, что K реализует скалярное произведение в пространстве, где для любого n и одночлена порядка k существует элемент $\psi(\mathbf{x})$, равный $\frac{1}{\sqrt{n!}} e^{-\frac{||\mathbf{x}||^2}{2}} \prod_{i=1}^{n} x_{I_i}$. Поэтому мы можем обучить любой полиномиальный предиктор в исходном пространстве, используя гауссово ядро.

Напомним, что VC-размерность класса полиномиальных предикторов бесконечна (см. упражнение 16.12). Здесь нет никакого противоречия, потому что выборочная сложность, необходимая для обучения гауссовых ядер, зависит от зазора в пространстве признаков, который, если нам повезет, окажется большим, но в общем случае может быть сколь угодно малым.

Гауссово ядро называют также радиально-базисной функцией (Radial Basis Function – RBF).

16.2.1. Ядра как способ выразить априорное знание

Мы уже обсуждали выше, что отображение в пространство признаков ψ можно рассматривать как обогащение класса линейных классификаторов (до класса, соответствующего линейным классификаторам в пространстве признаков). Но до сих пор в этой книге мы говорили, что пригодность класса гипотез к данной задаче обучения зависит от характера задачи. Поэтому погружение ψ можно интерпретировать как способ выразить и использовать априорное знание о решаемой задаче. Например, если мы уверены, что положительные примеры можно отделить с помощью некоторого эллипса, то можем определить ψ как множество всех одночленов порядка не выше 2, т. е. использовать полиномиальное ядро степени 2.

В качестве более реалистичного примера рассмотрим задачу обучения предиктора, который ищет в файле последовательность символов (сигнатуру), определяющую, содержит этот файл вирус или нет. Положим $\mathcal{X}$ равным множеству всех конечных строк над некоторым алфавитом Σ, и пусть $\mathcal{X}_d$ – множество всех таких строк длины не больше d. Мы хотим обучить класс гипотез $\mathcal{H} = \{h_v : v \in \mathcal{X}_d\}$, в котором для любой строки $x \in \mathcal{X} h_v(x) = 1$ тогда и только тогда, когда v – подстрока x (и $h_v(x) = -1$ в противном случае). Покажем, как с помощью подходящего погружения этот класс можно реализовать в виде линейных классификаторов в результирующем пространстве признаков. Рассмотрим отображение ψ в пространство $\mathbb{R}^s$, где $s = |\mathcal{X}_d|$, такое, что каждая координата $\psi(x)$ соответствует какой-

то строке v и показывает, является ли v подстрокой x (т. е. для любого $x \in \mathcal{X}$ $\psi(x)$ – вектор в пространстве $\{0, 1\}^{|\mathcal{X}d|}$). Отметим, что размерность этого пространства признаков экспоненциально растет вместе с d. Нетрудно видеть, что любой член класса $\mathcal{H}$ можно реализовать в виде композиции линейного классификатора и $\psi(x)$, и, более того, таким полупространством с нормой 1, для которого достигается зазор 1 (см. упражнение 16.1). Кроме того, для любого $x \in \mathcal{X}$ $||\psi(x)|| = O(\sqrt{d})$. В итоге получаем, что класс допускает обучение с помощью алгоритма SVM с выборочной сложностью, полиномиально зависящей от d. Однако размерность пространства признаков зависит от d экспоненциально, поэтому реализовать SVM в лоб на пространстве признаков проблематично. По счастью, в пространстве признаков легко вычислить скалярное произведение (т. е. ядерную функцию) без явного отображения образцов в пространство признаков. Действительно, $K(x, x')$ – это просто количество общих подстрок x и x', которое легко вычислить за время, полиномиально зависящее от d.

Этот пример показывает также, как отображение в пространство признаков позволяет использовать полупространства для невекторных областей определения.

16.2.2. Характеристика ядерных функций*

В предыдущем разделе мы отметили, что задание ядерной матрицы можно рассматривать как способ выразить априорное знание. Рассмотрим функцию сходства, заданную в виде $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Является ли она допустимой ядерной функцией? То есть представляет ли она скалярное произведение $\psi(\mathbf{x})$ и $\psi(\mathbf{x}')$ для некоторого отображения ψ? Следующая лемма дает ответ на этот вопрос в форме необходимого и достаточного условия.

Лемма 16.2. *Симметричная функция $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ реализует скалярное произведение в некотором гильбертовом пространстве тогда и только тогда, когда она является положительно полуопределенной, т. е. для любых $\mathbf{x}_1, \ldots, \mathbf{x}_m$ матрица Грама $G_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$ положительно полуопределенная.*

Доказательство. Тот факт, что если K реализует скалярное произведение в некотором гильбертовом пространстве, то матрица Грама положительно полуопределенная, тривиален. Обратно, определим пространство функций на $\mathcal{X}$: $\mathbb{R}^{\mathcal{X}} = \{f : \mathcal{X} \to \mathbb{R}\}$. Для любого $\mathbf{x} \in \mathcal{X}$ пусть $\psi(\mathbf{x})$ будет функцией $\mathbf{x} \mapsto K(\cdot, \mathbf{x})$. Определим векторное пространство, взяв все линейные комбинации элементов вида $K(\cdot, \mathbf{x})$. Определим скалярное произведение в этом пространстве:

$$\left\langle \sum_i \alpha_i K(\cdot, \mathbf{x}_i), \sum_j \beta_j K(\cdot, \mathbf{x}'_j) \right\rangle = \sum_{i,j} \alpha_i \beta_j K(\mathbf{x}_i, \mathbf{x}'_j).$$

Это действительно скалярное произведение, поскольку оно симметрично (т. к. K симметрична), линейно (очевидно) и положительно полуопределено (легко видеть, что $K(\mathbf{x}, \mathbf{x}) \geq 0$, причем равенство достигается, только когда $\psi(\mathbf{x})$ – нулевая функция). Очевидно, что

$$\langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle = \langle K(\cdot, \mathbf{x}), K(\cdot, \mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}'),$$

что и требовалось доказать. □

16.3. Реализация Soft-SVM с ядрами

Далее мы обратимся к решению Soft-SVM с ядрами. Хотя можно было бы спроектировать алгоритм для решения задачи (16.4), существует еще более простой подход, при котором непосредственно решается задача оптимизации Soft-SVM в пространстве признаков

$$\min_{\mathbf{w}}\left(\frac{\lambda}{2}\|\mathbf{w}\|^2+\frac{1}{m}\sum_{i=1}^{m}\max\{0,1-y\langle\mathbf{w},\psi(\mathbf{x}_i)\rangle\}\right), \tag{16.5}$$

с использованием только ядерных вычислений. Основное наблюдение состоит в том, что вектор $\mathbf{w}^{(t)}$ в процедуре СГС, описанной в разделе 15.5, всегда принадлежит линейной оболочке $\{\psi(\mathbf{x}_1),\ldots,\psi(\mathbf{x}_m)\}$. Следовательно, вместо того чтобы хранить $\mathbf{w}^{(t)}$, мы можем хранить соответствующие коэффициенты α.

Формально, пусть K – ядерная функция, т. е. для любых $\mathbf{x}$, $\mathbf{x}'$, $K(\mathbf{x},\mathbf{x}')=\langle\psi(\mathbf{x}),\psi(\mathbf{x}')\rangle$. Мы будем хранить два вектора из $\mathbb{R}^m$, соответствующие векторам $\boldsymbol{\theta}^{(t)}$ и $\mathbf{w}^{(t)}$, определенным в процедуре СГС из раздела 15.5. Пусть $\boldsymbol{\beta}^{(t)}$ – такой вектор, что

$$\boldsymbol{\theta}^{(t)}=\sum_{j=1}^{m}\beta_j^{(t)}\psi(\mathbf{x}_j), \tag{16.6}$$

а $\boldsymbol{\alpha}^{(t)}$ – такой вектор, что

$$\mathbf{w}^{(t)}=\sum_{j=1}^{m}\alpha_j^{(t)}\psi(\mathbf{x}_j). \tag{16.7}$$

Векторы $\boldsymbol{\beta}$ и $\boldsymbol{\alpha}$ обновляются в соответствии со следующей процедурой.

Применение СГС для решения Soft-SVM я ядрами

цель: решить уравнение (16.5)
параметр: T
инициализация: $\beta^{(1)}=\mathbf{0}$
for $t=1,\ldots,T$
  положить $\alpha^{(t)}=(1/\lambda_t)\beta^{(t)}$
  случайным образом с равномерным распределением выбрать i из $[m]$
  для всех $j\neq i$ положить $\beta_j^{(t+1)}=\beta_j^{(t)}$
  если $(y_i\sum_{j=1}^{m}\alpha_j^{(t)}K(\mathbf{x}_j,\mathbf{x}_i)<1)$
    положить $\beta_i^{(t+1)}=\beta_i^{(t)}+y_i$
  иначе
    ноложить $\beta_i^{(t+1)}=\beta_i^{(t)}$
Выход: $\bar{\mathbf{w}}=\sum_{j=1}^{m}\bar{\alpha}_j\psi(\mathbf{x}_j)$, где $\bar{\alpha}=(1/T)\sum_{t=1}^{T}\alpha^{(t)}$

Следующая лемма показывает, что описанная реализация эквивалентна выполнению описанной в разделе 15.5. процедуры СГС в пространстве признаков.

Лемма 16.3. *Пусть* $\hat{\mathbf{w}}$ *– выход процедуры СГС, описанной в разделе 15.5, когда она применяется в пространстве признаков, и обозначим* $\bar{\mathbf{w}}=\sum_{j=1}^{m}\bar{\alpha}_j\psi(\mathbf{x}_j)$ *– выход в случае применения СГС с ядрами. Тогда* $\hat{\mathbf{w}}=\bar{\mathbf{w}}$.

Доказательство. Мы покажем, что для любого t имеет место равенство (16.6), где $\boldsymbol{\theta}^{(t)}$ – результат выполнения описанной в разделе 15.5 процедуры СГС в пространстве признаков. По определению $\boldsymbol{\alpha}^{(t)} = (1/\lambda t)\boldsymbol{\beta}^{(t)}$ и $\mathbf{w}^{(t)} = (1/\lambda t)\boldsymbol{\theta}^{(t)}$, из этого утверждения вытекает, что равенство (16.7) также имеет место, откуда следует справедливость леммы. Доказательство проведем по индукции. Для $t = 1$ утверждение, очевидно, выполняется. Предположим, что оно верно для всех $t \geq 1$. Тогда

$$y_i \langle \mathbf{w}^{(t)}, \psi(\mathbf{x}_i) \rangle = y_i \left\langle \sum_j \alpha_j^{(t)} \psi(\mathbf{x}_j), \psi(\mathbf{x}_i) \right\rangle = y_i \sum_{j=1}^{m} \alpha_j^{(t)} K(\mathbf{x}_j, \mathbf{x}_i).$$

Следовательно, условия в обоих алгоритмах эквивалентны и, если мы обновим $\boldsymbol{\theta}$, то будем иметь

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + y_i \psi(\mathbf{x}_i) = \sum_{j=1}^{m} \beta_j^{(t)} \psi(\mathbf{x}_j) + y_i \psi(\mathbf{x}_i) = \sum_{j=1}^{m} \beta_j^{(t+1)} \psi(\mathbf{x}_j),$$

что и завершает доказательство. □

16.4. Резюме

Отображение заданной области определения в пространство более высокой размерности, в котором затем используется линейный предиктор, может оказаться исключительно действенной техникой. Мы получаем все преимущества более богатого класса гипотез, однако должны решать задачу повышенной выборочной и вычислительной сложности. В главе 10 мы обсуждали алгоритм AdaBoost, в котором эти проблемы решаются путем использования слабого обучаемого: хотя мы имеем дело с пространством очень высокой размерности, существует «оракул», который возвещает нам одну хорошую координату, с которой следует работать на данной итерации. В этой главе мы описали другой подход, ядерный трюк. Идея в том, что для нахождения линейного предиктора в многомерном пространстве нам нужно знать не представление образцов в этом пространстве, а значения скалярных произведений их образов. Вычисление скалярных произведений в многомерном пространстве без использования представлений образцов в нем производится с помощью ядерных функций. Мы также показали, что алгоритм СГС можно реализовать с использованием ядер.

Идея отображения в пространство признаков и ядерный трюк позволяют использовать аппарат полупространств и линейных предикторов для невекторных данных. Мы продемонстрировали, как можно использовать ядра для обучения предикторов на множестве строк.

Мы убедились в том, что ядерный трюк применим к SVM. Но ядерный трюк можно применить и во многих других алгоритмах. Некоторые примеры приведены в упражнениях.

На этом заканчиваются главы, посвященные линейным предикторам и выпуклым задачам. В следующих двух главах мы будем иметь дело с классами гипотез совершенно другого типа.

16.5. Библиографические сведения

В контексте SVM ядерный трюк впервые был описан в работе Boser et al. (1992). См. также Aizerman et al. (1964). Тот факт, что ядерный трюк можно применять всегда, когда алгоритм зависит только от скалярных произведений, впервые был отмечен в работе Schölkopf et al. (1998). Теорема о представителе доказана в работах Schölkopf et al., 2000; Schölkopf et al., 2001. Условия, сформулированные в лемме 16.2, упрощены усилиями Мерсера. В литературе можно найти много полезных ядерных функций для различных приложений. Отсылаем читателя к работе Schölkopf & Smola (2002).

16.6. Упражнения

16.1. Рассмотрим задачу о нахождении последовательности символов в файле, описанную в разделе 16.2.1. Покажите, что любой член класса $\mathcal{H}$ можно реализовать в виде композиции с $\psi(x)$ линейного классификатора с нормой 1 и зазором 1.

16.2. Ядерный перцептрон. Покажите, как можно выполнить алгоритм перцептрона, обращаясь к образцам только посредством ядерной функции. *Указание.* Рассуждение аналогично выводу реализации СГС с ядрами.

16.3. Ядерная гребневая регрессия. Проблема гребневой регрессии с отображением в пространство признаков ψ – это проблема нахождения вектора $\mathbf{w}$, который минимизирует функцию

$$f(\mathbf{w}) = \lambda \|\mathbf{w}\|^2 + \frac{1}{2m}\sum_{i=1}^{m}(\langle \mathbf{w}, \psi(\mathbf{x}_i)\rangle - y_i)^2 \tag{16.8}$$

с последующим возвратом предиктора

$$h(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x}\rangle.$$

Покажите, как реализовать алгоритм гребневой регрессии с ядрами.

Указание. Согласно теореме о представителе, существует вектор $\boldsymbol{\alpha} \in \mathbb{R}^m$ такой, что $\sum_{i=1}^{m}\alpha_i\psi(\mathbf{x}_i)$ доставляет минимум функции (16.8).

1. Пусть G – матрица Грама относительно S и K, т. е. $G_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. Определим $g : \mathbb{R}^m \to \mathbb{R}$

$$g(\boldsymbol{\alpha}) = \lambda \cdot \boldsymbol{\alpha}^T G\boldsymbol{\alpha} + \frac{1}{2m}\sum_{i=1}^{m}(\langle \boldsymbol{\alpha}, G_{:,i}\rangle - y_i)^2, \tag{16.9}$$

   где $G_{:,i}$ – i-й столбец G. Покажите, что если $\boldsymbol{\alpha}^*$ минимизирует функцию (16.9), то $\mathbf{w}^* = \sum_{i=1}^{m}\alpha_i^*\psi(\mathbf{x}_i)$ доставляет минимум f.
2. Найдите выражение $\boldsymbol{\alpha}^*$ в замкнутом виде.

16.4. Пусть N – произвольное целое положительное число. Для любых $x, x' \in \{1, \ldots, N\}$ определим

$$K(x, x') = \min\{x, x'\}.$$

Докажите, что K – корректное ядро, т. е. найдите отображение $\psi : \{1, ..., N\} \to H$, где H – некоторое гильбертово пространство, такое, что

$$\forall x, x' \in \{1, ..., N\} K(x, x') = \langle \psi(x), \psi(x') \rangle.$$

16.5. Директор супермаркета хочет обучить систему узнавать, у кого из покупателей есть дети, на основе анализа содержимого их тележек. Точнее, он производит выборку независимых и одинаково распределенных покупателей, и для i-го покупателя $x_i \subset \{1, ..., d\}$ обозначает подмножество купленных им предметов, а $y_i \in \{\pm 1\}$ – метка, показывающая, есть ли у этого покупателя ребенок. Директор априорно знает, что существует k предметов таких, что метка заведомо равна 1 тогда и только тогда, когда покупатель купил хотя бы один из этих предметов. Разумеется, что это за предметы, неизвестно (иначе нечему было бы обучаться). Кроме того, по правилам магазина, каждый покупатель может купить не более s предметов. Помогите директору спроектировать алгоритм обучения, для которого вычислительная и выборочная сложность полиномиально зависят от s, k и $1/\epsilon$.

16.6. Пусть $\mathcal{X}$ – множество образцов, а ψ – отображение $\mathcal{X}$ в некоторое гильбертово пространство признаков V. Обозначим $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ ядерную функцию, которая реализует скалярное произведение в пространстве V.

Рассмотрим алгоритм бинарной классификации, который назначает не виденному ранее образцу такую же метку, как у класса с ближайшим средним. Формально, пусть задан обучающий набор $S = (\mathbf{x}_1, y_1), ..., (\mathbf{x}_m, y_m)$. Для каждого $y \in \{\pm 1\}$ определим

$$c_y = \frac{1}{m_y} \sum_{i: y_i = y} \psi(\mathbf{x}_i),$$

где $m_y = |\{i : y_i = y\}|$. Предполагается, что m_+ и m_- отличны от нуля. Тогда алгоритм выводит следующее решающее правило:

$$h(\mathbf{x}) = \begin{cases} 1, & \|\psi(\mathbf{x}) - c_+\| \le \|\psi(\mathbf{x}) - c_-\| \\ 0, & \text{в противном случае} \end{cases}.$$

1. Пусть $\mathbf{w} = c_+ - c_-$ и $b = \frac{1}{2}(\|c_-\|^2 - \|c_+\|^2)$. Покажите, что

$$h(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \psi(\mathbf{x}) \rangle + b).$$

2. Покажите, как выразить $h(\mathbf{x})$ через ядерную функцию и не обращаясь напрямую к отдельным элементам $\psi(\mathbf{x})$ или $\mathbf{w}$.

ЛАНЬ®

Глава 17

МНОГОКЛАССОВАЯ КАТЕГОРИЗАЦИЯ, РАНЖИРОВАНИЕ И СЛОЖНЫЕ ПРОБЛЕМЫ ПРЕДСКАЗАНИЯ

Многоклассовая категоризация – это проблема классификации образцов при наличии нескольких возможных целевых классов. То есть мы хотим обучить предиктор $h : \mathcal{X} \to \mathcal{Y}$, где $\mathcal{Y}$ – конечное множество категорий. Например, такая задача возникает, когда нужно классифицировать документы по темам ($\mathcal{X}$ – множество документов, а $\mathcal{Y}$ – множество возможных тем) или определить, встречается ли объект в изображении ($\mathcal{X}$ – множество изображений, $\mathcal{Y}$ – множество возможных объектов).

Важность проблемы многоклассового обучения подстегнула разработку различных подходов к этой задаче. Пожалуй, самым прямолинейным из них является сведение многоклассовой классификации к бинарной. В разделе 17.1 мы обсудим два самых распространенных способа такого сведения, а также главный недостаток этого подхода.

Затем мы перейдем к описанию семейства линейных предикторов для многоклассовых проблем. Опираясь на методы минимизации регуляризированной потери (RLM) и СГС из предыдущих глав, мы опишем несколько практических алгоритмов многоклассовой классификации.

В разделе 17.3 мы покажем, как использовать аппарат многоклассовой классификации для решения сложных проблем предсказания, когда множество $\mathcal{Y}$ очень велико, но имеет определенную структуру. Эта задача часто называется *обучением со структурированным выходом*. В частности, мы продемонстрируем этот подход к решению задачи о распознавании рукописных слов, когда $\mathcal{Y}$ – множество всех возможных строк ограниченной длины (следовательно, размер $\mathcal{Y}$ экспоненциально зависит от максимальной длины слова).

Наконец, в разделах 17.4 и 17.5 обсуждаются проблемы ранжирования, когда обучаемый должен расположить множество образцов в порядке их «релевантности». Типичное приложение – упорядочение результатов поиска по релевантности запросу. Мы опишем несколько показателей, применяемых для оценки качества ранжирующих предикторов, и покажем, как эффективно обучать линейные предикторы для проблем ранжирования.

17.1. Один против всех и все пары

Простейший подход к проблемам многоклассового предсказания – сведение к бинарной классификации. Напомним, что в многоклассовой постановке мы хотим обучить функцию $h : \mathcal{X} \to \mathcal{Y}$. Без ограничения общности обозначим $\mathcal{Y} = \{1, \ldots, k\}$.

В методе «один против всех» (или «один против остальных») мы обучаем k бинарных классификаторов, каждый из которых отличает один класс от всех остальных. То есть, имея обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, где каждый y_i принадлежит $\mathcal{Y}$, мы строим k бинарных обучающих наборов $S_1, \ldots, S_k$, где $S_i = (\mathbf{x}_1, (-1)^{\mathbb{1}[y_1 \neq i]}), \ldots, (\mathbf{x}_m, (-1)^{\mathbb{1}[y_m \neq i]})$. Иными словами, в наборе S_i образец помечен 1, если в исходном наборе S он был помечен меткой i, и −1 в противном случае. Для каждого $i \in [k]$ мы обучаем бинарный предиктор $h_i : X \to \{\pm 1\}$ на основе S_i в надежде, что $h_i(\mathbf{x})$ будет равно 1 тогда и только тогда, когда $\mathbf{x}$ принадлежит классу i. Затем, имея $h_1, \ldots, h_k$, мы строим многоклассовый предиктор по правилу

$$h(\mathbf{w}) \in \operatorname*{argmax}_{i \in [k]} h_i(\mathbf{x}). \tag{17.1}$$

Если сразу несколько бинарных гипотез предсказывают «1», то мы должны как-то решить, какой класс предсказывать (например, можно устранить неоднозначность, произвольно взяв минимальный индекс в множестве $\operatorname{argmax}_i h_i(\mathbf{x})$). Более разумный подход возможен, когда каждая h_i обладает дополнительной скрытой информацией, которую можно интерпретировать как уверенность в предсказании $y = i$. Например, так обстоит дело в случае полупространств, когда собственно предсказанием является $\operatorname{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$, а $\langle \mathbf{w}, \mathbf{x} \rangle$ можно интерпретировать как уверенность в этом предсказании. В таких случаях можно применить многоклассовое правило (17.1) к вещественным значениям предсказаний. Ниже приведен псевдокод алгоритма «один против всех».

Один против всех

вход:
обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
алгоритм бинарной классификации A
foreach $i \in \mathcal{Y}$
положить $S_i = (\mathbf{x}_1, (-1)^{\mathbb{1}[y_1 \neq i]}), \ldots, (\mathbf{x}_m, (-1)^{\mathbb{1}[y_m \neq i]})$
положить $h_i = A(S_i)$
выход:
многоклассовая гипотеза $h(\mathbf{x}) \in \operatorname{argmax}_{i \in \mathcal{Y}} h_i(\mathbf{x})$

Еще одним популярным способом сведения является подход «все пары», когда все пары классов сравниваются между собой. Формально, пусть дан обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, где каждая метка y_i принадлежит $[k]$. Для любых $1 \le i < j \le k$ мы строим бинарный обучающий набор $S_{i,j}$, содержащий те и только те примеры из S, для которых метка равна i или j. Каждому такому примеру назначается бинарная метка, равная +1, если в исходном наборе S он имел метку i, и −1, если он имел метку j. Затем мы обучаем алгоритм бинарной

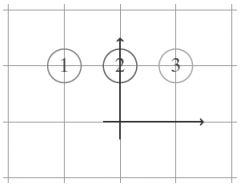
классификации на основе набора $S_{i,j}$ и получаем гипотезу $h_{i,j}$. Наконец, мы строим многоклассовый классификатор, который предсказывает класс, получивший максимальное число «побед». Ниже приведен псевдокод алгоритма «все пары».

Все пары

вход:
обучающий набор $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
алгоритм бинарной классификации A
foreach $i, j \in \mathcal{Y}$ таких, что $i < j$
инициализировать $S_{i,j}$ пустой последовательностью
for $t = 1, \ldots, m$
если $y_t = i$, добавить $(\mathbf{x}_t, 1)$ в $S_{i,j}$
если $y_t = j$, добавить $(\mathbf{x}_t, -1)$ в $S_{i,j}$
положить $h_{i,j} = A(S_{i,j})$
выход:
многоклассовая гипотеза $h(\mathbf{x}) \in \operatorname{argmax}_{i \in \mathcal{Y}}(\sum_{j \in \mathcal{Y}} \operatorname{sign}(j - i) h_{i,j}(\mathbf{x}))$

Хотя методы сведения, в т. ч. «один против всех» и «все пары» просты и легко строятся на основе существующих алгоритмов, у этой простоты есть цена. Бинарный обучаемый ничего не знает о том, что мы собираемся использовать его выходную гипотезу для построения многоклассового предиктора, поэтому результат может получиться неоптимальным, что доказывает следующий пример.

Пример 17.1. Рассмотрим проблему многоклассовой категоризации, в которой область образцов $\mathcal{X} = \mathbb{R}^2$, а область меток $\mathcal{Y} = \{1, 2, 3\}$. Предположим, что образцы различных классов находятся в непересекающихся шарах, как показано на рисунке ниже.



Пусть массы вероятности классов 1, 2, 3 составляют 40%, 20% и 40% соответственно. Рассмотрим применение метода «один против всех» к этой проблеме и предположим, что используемый в нем алгоритм бинарной классификации – ERM относительно класса полупространств. Заметим, что для проблемы различения класса 2 и всех остальных оптимальным полупространством был бы классификатор, назначающий всем примерам отрицательные метки. Поэтому многоклассовый предиктор, построенный алгоритмом «один против всех», ошибается на всех примерах из класса 2 (так будет, если неоднозначность при определении $h(\mathbf{x})$ разрешается выбором числового значения метки класса). С другой стороны, если выбрать $h_i(\mathbf{x}) = \langle \mathbf{w}_i, \mathbf{x} \rangle$, где $\mathbf{w}_1 = (-1/\sqrt{2}, 1/\sqrt{2})$, $\mathbf{w}_2 = (0, 1)$, $\mathbf{w}_3 = (1/\sqrt{2}, 1/\sqrt{2})$, то классификатор $h(\mathbf{x}) = \operatorname{argmax}_i h_i(\mathbf{x})$ точно предсказывает все без

исключения примеры. Мы видим, что хотя ошибка аппроксимации класса предикторов вида $h(\mathbf{x}) = \text{argmax}_i \langle \mathbf{w}_i, \mathbf{x} \rangle$ равна нулю, метод «один против всех» может и не найти хороший предиктор из этого класса.

17.2. Линейные многоклассовые предикторы

Поскольку методы сведения не дают адекватного результата, в этом разделе мы изучим более прямой подход к обучению многоклассовых предикторов. Мы опишем семейство линейных многоклассовых предикторов. Чтобы обосновать его построение, напомним, что линейный предиктор для бинарной классификации (например, полупространство) имеет вид

$$h(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle).$$

Эквивалентно это предсказание можно выразить в виде

$$h(\mathbf{x}) = \underset{y \in \{\pm 1\}}{\text{argmax}} \langle \mathbf{w}, y\mathbf{x} \rangle.$$

где $y\mathbf{x}$ – вектор, полученный умножением каждого элемента $\mathbf{x}$ на y.

Это представление ведет к следующему естественному обобщению полупространств на многоклассовые проблемы. Пусть $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ – *зависящее от класса отображение в пространство признаков*. Это значит, что Ψ получает пару $(\mathbf{x}, y)$ и отображает ее в d-мерный вектор признаков. Интуитивно мы можем рассматривать элементы вектора $\Psi(\mathbf{x}, y)$ как функции, оценивающие, насколько хорошо метка y соответствует образцу $\mathbf{x}$. Мы еще уточним детали Ψ ниже. Зная Ψ и вектор $\mathbf{w} \in \mathbb{R}^d$, мы можем определить многоклассовый предиктор $h : \mathcal{X} \to \mathcal{Y}$ следующим образом:

$$h(\mathbf{x}) = \underset{y \in \mathcal{Y}}{\text{argmax}} \langle \mathbf{w}, \Psi(\mathbf{x}, y) \rangle.$$

То есть для входа $\mathbf{x}$ предиктор h возвращает метку, на которой достигается наибольшая взвешенная оценка, причем веса определяются вектором $\mathbf{w}$.

Пусть W – множество векторов в $\mathbb{R}^d$, например, $W = \{\mathbf{w} \in \mathbb{R}^d : ||\mathbf{w}|| \leq B\}$ для некоторого скаляра $B > 0$. Каждая пара (Ψ, W) определяет класс гипотез многоклассовых предикторов:

$$\mathcal{H}_{\Psi, W} = \{\mathbf{x} \mapsto \underset{y \in \mathcal{Y}}{\text{argmax}} \langle \mathbf{w}, \Psi(\mathbf{x}, y) \rangle : \mathbf{w} \in W\}.$$

Конечно, сразу возникает вопрос, как построить хорошее отображение Ψ, и мы обязательно обсудим его. А пока заметим, что если $\mathcal{Y} = \{\pm 1\}$ и мы полагаем $\Psi(\mathbf{x}, y) = y\mathbf{x}$ и $W = \mathbb{R}^d$, то $\mathcal{H}_{\Psi, W}$ оказывается классом однородных полупространств для бинарной классификации.

17.2.1. Как построить Ψ

Как уже отмечалось, мы можем рассматривать элементы $\Psi(\mathbf{x}, y)$ как функция, оценивающие, насколько хорошо метка yсоответствует образцу $\mathbf{x}$. Естественно, проектирование хорошего отображения Ψ напоминает задачу проектирования

хорошего отображения в пространство признаков (мы обсуждали ее в главе 16 и еще вернемся в главе 25). Ниже приведено два примера полезных конструкций.

Многовекторное построение

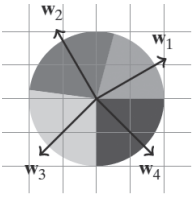
Пусть $\mathcal{Y}$= {1, ..., k} и $\mathcal{X} = \mathbb{R}^n$. Определим $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$, где $d = nk$, следующим образом:

$$\Psi(\mathbf{x}, y) = [\underbrace{0, \ldots, 0}_{\in \mathbb{R}^{(y-1)n}}, \underbrace{x_1, \ldots, x_n}_{\in \mathbb{R}^n}, \underbrace{0, \ldots, 0}_{\in \mathbb{R}^{(k-y)n}}]. \tag{17.2}$$

Иначе говоря, $\Psi(\mathbf{x}, y)$ составлено из k векторов размерности n каждый, и все эти векторы нулевые, кроме y-го, который равен $\mathbf{x}$. Поэтому $\mathbf{w} \in \mathbb{R}^{nk}$ можно рассматривать как k весовых векторов в $\mathbb{R}^n$, т. е. $\mathbf{w} = [\mathbf{w}_1; \ldots; \mathbf{w}_k]$, отсюда и название *многовекторное построение*. По построению, имеем $\langle \mathbf{w}, \Psi(\mathbf{x}, y)\rangle = \langle \mathbf{w}_y, \mathbf{x}\rangle$, и потому многоклассовое предсказание принимает вид:

$$h(\mathbf{x}) = \operatorname*{argmax}_{y \in \mathcal{Y}} \langle \mathbf{w}_y, \mathbf{x}\rangle.$$

На рисунке ниже приведена геометрическая интерпретация многоклассового предсказания в $\mathcal{X} = \mathbb{R}^2$.



TF-IDF

Приведенное выше определение $\Psi(\mathbf{x}, y)$ не содержит априорного знания о задаче. Далее мы опишем пример функции Ψ, которая включает такое знание. Пусть $\mathcal{X}$ – множество текстовых документов, $\mathcal{Y}$ – множество возможных тем. Пусть d – размер словаря. Для каждого слова в словаре с индексом j обозначим $TF(j, \mathbf{x})$ количество вхождений j-го слова в документ $\mathbf{x}$. Эта величина называется *частотой терма*. Кроме того, обозначим $DF(j, y)$ – количество вхождений j-го слова в документы, не относящиеся к теме y. Эта величина называется *частотой документа* и измеряет, часто ли слово j встречается в других темах. Теперь определим Ψ: $\mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ следующим образом

$$\Psi_j(\mathbf{x}, y) = TF(j, \mathbf{x}) \log\left(\frac{m}{DF(j, y)}\right),$$

где m – количество документов в обучающем наборе. Эта величина называется частотой терма-обратной частотой документа, или коротко TF-IDF. Интуитивно понятно, что $\Psi_j(\mathbf{x}, y)$ должно быть велико, если j-е слово часто встречается в документе $\mathbf{x}$, но вообще не встречается в документах, не относящихся к теме

y. Если дело обстоит именно так, что мы склонны поверить, что документ $\mathbf{x}$ относится к теме y. Заметим, что в отличие от описанного выше многовекторного построения, в этой конструкции размерность Ψ не зависит от количества тем (т. е. размера $\mathcal{Y}$).

17.2.2. Стоимостная классификация

До сих пор мы использовали в роли показателя качества $h(\mathbf{x})$ бинарную потерю. В некоторых ситуациях более осмысленно по-разному штрафовать за ошибки разных видов. Например, в задачах распознавания объектов предсказание, сообщающее, что изображение тигра содержит кошку, не так плохо, как утверждение, будто изображен кит. Это можно смоделировать, задав функцию потерь $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$, так что для каждой пары меток y', y потеря в случае предсказания метки y', когда правильной является метка y, равна $\Delta(y', y)$. Предполагается, что $\Delta(y, y) = 0$. Заметим, что бинарную потерю легко смоделировать, положив $\Delta(y', y) = \mathbb{1}_{[y' \neq y]}$.

17.2.3. ERM

Мы определили класс гипотез $\mathcal{H}_{\Psi,W}$ и функцию потерь Δ. Чтобы обучить класс с заданной функцией потерь, мы можем применить правило ERM относительно этого класса. То есть мы ищем многоклассовую гипотезу $h \in \mathcal{H}_{\Psi,W}$, параметризованную вектором $\mathbf{w}$, которая минимизирует эмпирический риск относительно Δ:

$$L_S(h) = \frac{1}{m} \sum_{i=1}^{m} \Delta(h(\mathbf{x}_i), y_i).$$

Покажем, что если $W = \mathbb{R}^d$ и выполнено предположение о реализуемости, то проблему ERM можно решить эффективно методами линейного программирования. Действительно, в реализуемом случае нам нужно найти вектор $\mathbf{w} \in \mathbb{R}^d$, который удовлетворяет условию

$$\forall i \in [m], \quad y_i = \operatorname*{argmax}_{y \in \mathcal{Y}} \langle \mathbf{w}, \Psi(\mathbf{x}_i, y) \rangle.$$

Или, что то же самое, требуется, чтобы вектор удовлетворял такой системе линейных неравенств:

$$\forall i \in [m], \quad \forall y \in \mathcal{Y} \setminus \{y_i\}, \quad \langle \mathbf{w}, \Psi(\mathbf{x}_i, y_i) \rangle > \langle \mathbf{w}, \Psi(\mathbf{x}_i, y) \rangle.$$

Нахождение такого вектора $\mathbf{w}$ сводится к решению линейной программы.

Как и в случае бинарной классификации, для решения проблемы ERM можно также воспользоваться обобщением алгоритма перцептрона. См. упражнение 17.2.

В нереализуемом случае решение проблемы ERM, вообще говоря, вычислительно трудная задача. Эта трудность обходится путем использования выпуклой суррогатной функции потерь (см. раздел 12.3). В частности, мы обобщим кусочно-линейную потерю на многоклассовые проблемы.

17.2.4. Обобщенная кусочно-линейная потеря

Напомним, что в случае бинарной классификации кусочно-линейная функция потерь определяется как $\max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}$. Мы собираемся обобщить эту функцию на многоклассовые предикторы вида

$$h_{\mathbf{w}}(\mathbf{x}) = \operatorname*{argmax}_{y \in \mathcal{Y}} \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}')\rangle.$$

Напомним, что суррогатная выпуклая потеря должна ограничивать сверху исходную невыпуклую потерю, каковой в нашем случае является $\Delta(h_{\mathbf{w}}(\mathbf{x}), y)$. Чтобы вывести верхнюю границу для $\Delta(h_{\mathbf{w}}(\mathbf{x}), y)$, заметим сначала, что из определения $h_{\mathbf{w}}(\mathbf{x})$ следует, что

$$\langle \mathbf{w}, \Psi(\mathbf{x}, y)\rangle \le \langle \mathbf{w}, \Psi(\mathbf{x}, h_{\mathbf{w}}(\mathbf{x}))\rangle.$$

Поэтому

$$\Delta(h_{\mathbf{w}}(\mathbf{x}), y) \le \Delta(h_{\mathbf{w}}(\mathbf{x}), y) \le \langle \mathbf{w}, \Psi(\mathbf{x}, h_{\mathbf{w}}(\mathbf{x})) - \Psi(\mathbf{x}, y)\rangle.$$

Поскольку $h_{\mathbf{w}}(\mathbf{x}) \in \mathcal{Y}$, мы можем ограничить правую часть сверху выражением

$$\max_{y' \in \mathcal{Y}} (\Delta(y', y) + \langle \mathbf{w}, \Psi(\mathbf{x}, y') - \Psi(\mathbf{x}, y)\rangle) \stackrel{\text{def}}{=} \ell(\mathbf{w}, (\mathbf{x}, y)). \tag{17.3}$$

Это выражение мы называем «обобщенной кусочно-линейной потерей». Как мы только что показали, $\ell(\mathbf{w}, (\mathbf{x}, y)) \ge \Delta(h_{\mathbf{w}}(\mathbf{x}), y)$. При этом равенство имеет место, когда оценка правильной метки больше, чем оценка любой другой метки y', по меньшей мере на $\Delta(y', y)$, т. е.

$$\forall y' \in \mathcal{Y} \setminus \{y\}, \quad \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y})\rangle \ge \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}')\rangle + \Delta(y', y).$$

Также сразу видно, что функция $\ell(\mathbf{w}, (\mathbf{x}, y))$ выпукла относительно $\mathbf{w}$, поскольку это максимум линейных функций от $\mathbf{w}$ (см. утверждение 12.5 в главе 12), и что $\ell(\mathbf{w}, (\mathbf{x}, y))$ является ρ-липшицевой с $\rho = \max_{y' \in \mathcal{Y}} \|\Psi(\mathbf{x}, y') - \Psi(\mathbf{x}, y)\|$.

Замечание 17.2. Мы пользуемся термином «обобщенная кусочно-линейная потеря», потому что в бинарном случае, когда $\mathcal{Y} = \{\pm 1\}$, если положить $\Psi(\mathbf{x}, y) = y\mathbf{x}/2$, то эта обобщенная потеря превращается в обычную кусочно-линейную потерю для бинарной классификации

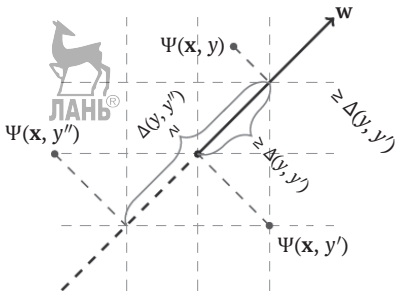
$$\ell(\mathbf{w}, (\mathbf{x}, y)) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}.$$

Геометрическая интерпретация

Функция $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ отображает каждый образец $\mathbf{x}$ в $|\mathcal{Y}|$ векторов в $\mathbb{R}^d$. Значение $\ell(\mathbf{w}, (\mathbf{x}, y))$ будет равно нулю, если существует направление $\mathbf{w}$ такое, что при проецировании на него этих $|\mathcal{Y}|$ векторов получается, что каждый вектор представлен скаляром $\langle \mathbf{w}, \Psi(\mathbf{x}, y)\rangle$, и мы можем ранжировать точки на основе этих скаляров, так что:

- точка, соответствующая правильной метке y, имеет наивысший ранг;
- для любого $y' \ne y$ разность между $\langle \mathbf{w}, \Psi(\mathbf{x}, y)\rangle$ и $\langle \mathbf{w}, \Psi(\mathbf{x}, y')\rangle$ больше, чем потеря при предсказании y' вместо y. Разность $\langle \mathbf{w}, \Psi(\mathbf{x}, y)\rangle - \langle \mathbf{w}, \Psi(\mathbf{x}, y')\rangle$ называют также «зазором» (см. раздел 15.1).

Это показано на следующем рисунке:



17.2.5. SVM и СГС в многоклассовом случае

Определив обобщенную кусочно-линейную функцию потерь, мы получили выпуклую–липшицеву проблему обучения и можем применить к ней общие методы решения таких проблем. В частности, изученная в главе 13 техника RLM приводит к многоклассовому правилу SVM:

Многоклассовый SVM

вход: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
параметры:
параметр регуляризации $\lambda > 0$
функция потерь $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$
зависящее от класса отображение в пространство признаков $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$
решить:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \left(\lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \max_{y' \in \mathcal{Y}} \left(\Delta(y', y_i) + \langle \mathbf{w}, \Psi(\mathbf{x}_i, y') - \Psi(\mathbf{x}_i, y_i) \rangle \right) \right)$$

выход: предиктор $h_{\mathbf{w}}(\mathbf{x}) = \operatorname{argmax}_{y \in \mathcal{Y}} \langle \mathbf{w}, \Psi(\mathbf{x}, y) \rangle$

Мы можем решить задачу оптимизации, возникающую в связи с многоклассовым SVM, применяя общие алгоритмы выпуклой оптимизации (или с помощью метода, описанного в разделе 15.5). Проанализируем риск результирующих гипотез. Наш подход естественно вытекает из общего анализа выпуклых–липшицевых проблем, проведенного в главе 13. В частности, применяя следствие 13.8 и используя тот факт, что обобщенная кусочно-линейная потеря ограничивает сверху потерю Δ, мы сразу получаем аналог следствия 15.7:

Следствие 17.1. *Пусть $\mathcal{D}$ – распределение на $\mathcal{X} \times \mathcal{Y}$, $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$, и предположим, что для всех $\mathbf{x} \in \mathcal{X}$ и $y \in \mathcal{Y}$ имеет место неравенство $\|\Psi(\mathbf{x}, y)\| \le \rho/2$. Положим $B > 0$. Рассмотрим выполнение SVM с $\lambda = \sqrt{\frac{2\rho^2}{B^2 m}}$ на обучающем наборе $S \sim \mathcal{D}^m$, и пусть $h_{\mathbf{w}}$ – выход многоклассового SVM. Тогда*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L^{\Delta}_{\mathcal{D}}(h_{\mathbf{w}})] \le \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L^{\text{g-hinge}}_{\mathcal{D}}(\mathbf{w})] \le \min_{\mathbf{u}:\|\mathbf{u}\|\le B} L^{\text{g-hinge}}_{\mathcal{D}}(\mathbf{u}) + \sqrt{\frac{8\rho^2 B^2}{m}},$$

где $L^{\Delta}_{\mathcal{D}}(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\Delta(h(\mathbf{x}), y)]$ *и* $L^{\text{g-hinge}}_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\ell(\mathbf{w}, (\mathbf{x}, y))]$, *а* ℓ *– обобщенная кусочно-линейная потеря, определенная в (17.3).*

Мы также можем применить схему обучения с помощью СГС к минимизации $L^{\text{g-hinge}}_{\mathcal{D}}(\mathbf{w})$, как описано в главе 14. Напомним утверждение 14.6, в котором речь шла о субградиентах функций max. Согласно этому утверждению, чтобы найти субградиент обобщенной кусочно-линейной потери, нам всего-то и нужно, что найти такое $y \in \mathcal{Y}$, на котором достигается максимум, указанный в определении обобщенной кусочно-линейной потери. Таким образом, мы приходим к следующему алгоритму.

Применение СГС для многоклассового обучения

параметры:
скаляр $\eta > 0$, целое число $T > 0$
функция потерь $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$
зависящее от класса отображение в пространство признаков $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$
инициализация: $\mathbf{w}^{(1)} = \mathbf{0} \in \mathbb{R}^d$
for $t = 1, 2, \ldots, T$
выбрать $(\mathbf{x}, y) \sim \mathcal{D}$
найти $\hat{y} \in \operatorname{argmax}_{y' \in \mathcal{Y}}(\Delta(y', y) + \langle \mathbf{w}^{(t)}, \Psi(\mathbf{x}, y') - \Psi(\mathbf{x}, y)\rangle)$
положить $\mathbf{v}_t = \Psi(\mathbf{x}, \hat{y}) - \Psi(\mathbf{x}, y)$
обновить $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
выход: $\bar{\mathbf{w}} = (1/T)\sum_{t=1}^{T} \mathbf{w}^{(t)}$

Из общего анализа СГС, резюмированного в следствии 14.12, сразу вытекает:

Следствие 17.2. *Пусть* $\mathcal{D}$ *– распределение на* $\mathcal{X} \times \mathcal{Y}$, Ψ: $\mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$, *и предположим, что для всех* $\mathbf{x} \in \mathcal{X}$ *и* $y \in \mathcal{Y}$ *имеет место неравенство* $\|\Psi(\mathbf{x}, y)\| \le \rho/2$. *Положим* $B > 0$. *Тогда для любого* $\epsilon > 0$ *справедливо следующее утверждение: если выполнить* T *итераций (т. е. взять столько примеров) алгоритма СГС для многоклассового обучения, где*

$$T \ge \frac{B^2\rho^2}{\epsilon^2}$$

и $\eta = \sqrt{\dfrac{B^2}{\rho^2 T}}$, *то на выходе будет получен вектор* $\bar{\mathbf{w}}$, *удовлетворяющий следующим условиям:*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L^{\Delta}_{\mathcal{D}}(h_{\bar{\mathbf{w}}})] \le \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L^{\text{g-hinge}}_{\mathcal{D}}(\bar{\mathbf{w}})] \le \min_{\mathbf{u}:\|\mathbf{u}\|\le B} L^{\text{g-hinge}}_{\mathcal{D}}(\mathbf{u}) + \epsilon.$$

Замечание 17.3. Интересно отметить, что границы риска, указанные в следствиях 17.1 и 17.2, явно не зависят от размера множества меток $\mathcal{Y}$ – мы восполь-

зуемся этим фактом в следующем разделе. Однако они могут неявно зависеть от размера $\mathcal{Y}$ через норму $\Psi(\mathbf{x}, y)$ и вследствие того факта, что границы имеют смысл, только когда существует некоторый вектор $\mathbf{u}$, $||\mathbf{u}|| \leq B$, для которого $L_D^{\text{g-hinge}}(\mathbf{u})$ не слишком велико.

17.3. Предсказание структурированного выхода

Проблемы предсказания структурированного выхода – это многоклассовые проблемы, в которых множество $\mathcal{Y}$ очень велико, но наделено предопределенной структурой. Структура играет ключевую роль в построении эффективных алгоритмов. Чтобы объяснить, зачем нужно изучать такие проблемы, рассмотрим задачу оптического распознавания символов (OCR). Предположим, что мы получили изображение написанного от руки слова и хотим предсказать, какое слово изображено. Чтобы упростить постановку задачи, допустим, что мы знаем, как сегментировать изображение, выделив последовательность частей, содержащих фрагменты, соответствующие одной букве. То есть $\mathcal{X}$ будет множеством последовательностей изображений, а $\mathcal{Y}$ – множеством последовательностей букв. Заметим, что размер $\mathcal{Y}$ экспоненциально зависит от максимальной длины слова. Пример изображения $\mathbf{x}$, соответствующего метке $\mathbf{y}$ = «workable», приведен на рисунке ниже.

Чтобы подойти к решению проблемы структурного предсказания, мы можем опереться на семейство линейных предикторов, описанное в предыдущем разделе. В частности, нужно определить разумную функцию потерь Δ, а также хорошее зависящее от класса отображение в пространство признаков Ψ. Под «хорошим» мы понимаем изображение, которое дает малую ошибку аппроксимации на классе линейных предикторов относительно Δ и Ψ. Сделав это, мы сможем взять за основу, например, алгоритм обучения СГС, определенный в предыдущем разделе.

Однако в связи с гигантским размером $\mathcal{Y}$ возникают некоторые трудности.

1. Чтобы применить многоклассовое предсказание, мы должны решить задачу максимизации на $\mathcal{Y}$. Как обеспечить эффективность предсказания при столь большом размере $\mathcal{Y}$?
2. Как эффективно обучить $\mathbf{w}$? В частности, чтобы применить правило СГС, нам опять-таки необходимо решить задачу максимизации на $\mathcal{Y}$.
3. Как избежать переобучения?

В предыдущем разделе мы уже показали, что выборочная сложность обучения линейного многоклассового предиктора явно не зависит от количества классов.

Нужно только убедиться, что норма области значений Ψ не слишком велика. Это решит проблему переобучения. Чтобы справиться с вычислительными трудностями, мы примем во внимание структуру проблемы и определим функции Ψ и Δ, так чтобы задачи максимизации в определении $h_{\mathbf{w}}$ и в алгоритме СГС можно было решить эффективно. Далее мы продемонстрируем один из способов достижения этих целей для задачи OCR.

Чтобы упростить изложение, предположим, что все слова в $\mathcal{Y}$ имеют длину r и что количество букв в алфавите равно q. Пусть $\mathbf{y}$ и $\mathbf{y}'$ – два слова (т. е. последовательности букв) в $\mathcal{Y}$. Определим функцию $\Delta(\mathbf{y}', \mathbf{y})$ как среднее количество букв, различающихся в $\mathbf{y}'$ и $\mathbf{y}$, т. е. $(1/r)\sum_{i=1}^{r} \mathbb{1}_{[y_i \neq y'_i]}$.

Далее определим зависящее от класса отображение в пространство признаков $\Psi(\mathbf{x}, \mathbf{y})$. Удобно представлять себе $\mathbf{x}$ как матрицу размера $n \times r$, где n – число пикселей в изображении, а r – число изображений в последовательности. j-й столбец $\mathbf{x}$ соответствует j-му изображению (представленному в полутоновом формате). Размерность области значений Ψ будет равна $d = nq + q^2$.

Первые nq признаков имеют «тип 1» и выглядят так:

$$\Psi_{i,j,1}(\mathbf{x}, \mathbf{y}) = \frac{1}{r} \sum_{t=1}^{r} x_{i,t} \mathbb{1}_{[y_t = j]}.$$

То есть мы суммируем значения i-го пикселя только по изображениям, которым $\mathbf{y}$ назначает букву j. Тройной индекс $(i, j, 1)$ означает, что мы имеем дело с признаком (i, j) типа 1. Интуитивно понятно, что такие признаки могут уловить пиксели, яркость которых свидетельствует о наличии определенной буквы. Признаки второго типа имеют вид

$$\Psi_{i,j,2}(\mathbf{x}, \mathbf{y}) = \frac{1}{r} \sum_{t=2}^{r} \mathbb{1}_{[y_t = i]} \mathbb{1}_{[y_{t-1} = j]}.$$

То есть мы подсчитываем, сколько раз буква i следует сразу за буквой j. Интуитивно эти признаки улавливают закономерности вида «вероятно встретить в слове пару ‘qu’» или «невероятно встретить в слове пару ‘rz’». Конечно, не все такие признаки полезны, поэтому задача обучения – назначить признакам веса, обучив вектор $\mathbf{w}$, так чтобы показанная ниже взвешенная оценка давала хорошее предсказание

$$h_{\mathbf{w}}(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}) \rangle.$$

Осталось показать, как эффективно решить задачу оптимизации, поставленную в определении $h_{\mathbf{w}}(\mathbf{x})$, и задачу оптимизации в определении $\hat{y}$ в алгоритме СГС. Это будет сделано с помощью динамического программирования. Мы опишем процедуру решения первой задачи и оставим в качестве упражнения решение второй.

Чтобы вывести процедуру динамического программирования, заметим сначала, что можно записать

$$\Psi(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{r} \boldsymbol{\varphi}(\mathbf{x}, y_t, y_{t-1}),$$

при подходящей функции $\varphi : \mathcal{X} \times [q] \times [q] \cup \{0\} \to \mathbb{R}^d$ и для простоты будем предполагать, что y_0 равно 0. В самом деле, каждый признак $\Psi_{i,j,1}$ можно выразить через

$$\varphi_{i,j,1}(\mathbf{x}, y_t, y_{t-1}) = x_{i,1} \mathbb{1}_{[y_t = j]},$$

а признак $\Psi_{i,j,1}$ – через

$$\varphi_{i,j,2}(\mathbf{x}, y_t, y_{t-1}) = \mathbb{1}_{[y_t = i]} \mathbb{1}_{[y_{t-1} = j]}.$$

Поэтому предсказание можно записать в виде

$$h_{\mathbf{w}}(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \sum_{t=1}^{r} \langle \mathbf{w}, \boldsymbol{\varphi}(\mathbf{x}, y_t, y_{t-1}) \rangle. \tag{17.4}$$

Далее мы выведем процедуру динамического программирования, которая решает любую проблему вида (17.4). Мы будем хранить матрицу $M \in \mathbb{R}^{q,r}$ такую, что

$$M_{s,\tau} = \max_{(y_1, \ldots, y_\tau) : y_\tau = s} \sum_{t=1}^{\tau} \langle \mathbf{w}, \boldsymbol{\varphi}(\mathbf{x}, y_t, y_{t-1}) \rangle.$$

Очевидно, что максимум $\langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}) \rangle$ равен $\max_s M_{s,r}$. Матрицу M можно вычислять рекурсивно:

$$M_{s,r} = \max_{s'} (M_{s',\tau-1}) + \langle \mathbf{w}, \boldsymbol{\varphi}(\mathbf{x}, s, s') \rangle). \tag{17.5}$$

В результате приходим к такой процедуре.

Вычисление $h_{\mathbf{w}}(\mathbf{x})$ из формулы (17.4) методом динамического программирования

вход: матрица $\mathbf{x} \in \mathbb{R}^{n,r}$ и вектор $\mathbf{w}$
инициализация:
  foreach $s \in [q]$
    $M_{s,1} = \langle \mathbf{w}, \varphi(\mathbf{x}, s, -1) \rangle$
for $\tau = 2, \ldots, r$
  foreach $s \in [q]$
    положить $M_{s,\tau}$ как в выражении (17.5)
    положить $I_{s,\tau}$ равным такому s', который максимизирует (17.5)
положить $y_t = \operatorname{argmax}_s M_{s,r}$
for $\tau = r, r-1, \ldots, 2$
  положить $y_{\tau-1} = I_{y_\tau,\tau}$
выход: $\mathbf{y} = (y_1, \ldots, y_r)$

17.4. Ранжирование

Ранжирование – это задача об упорядочении множества образцов по «релевантности». Типичное приложение – сортировка результатов поиска по их релевантности запросу. Другой пример – система, который наблюдает за электронными транзакциями и поднимает тревогу при обнаружении мошеннических

транзакций. Такая система должна упорядочивать транзакции по степени подозрительности.

Формально, обозначим $\mathcal{X}^* = \bigcup_{n=1}^{\infty} \mathcal{X}^n$ множество всех последовательностей образцов из $\mathcal{X}$ произвольной длины. Ранжирующая гипотеза h – это функция, которая получает последовательность образцов $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_r) \in \mathcal{X}^*$ и возвращает перестановку $[r]$. Удобнее считать, что выходом h является вектор $\mathbf{y} \in \mathbb{R}^r$, а в результате сортировки элементов $\mathbf{y}$ по возрастанию получается перестановка $[r]$. Будем обозначать $\pi(\mathbf{y})$ перестановку $[r]$, индуцированную вектором $\mathbf{y}$. Например, если $r = 5$, то вектор $\mathbf{y} = (2, 1, 6, -1, 0{,}5)$ индуцирует перестановку $\pi(\mathbf{y}) = (4, 3, 5, 1, 2)$. Действительно, отсортировав $\mathbf{y}$ в порядке возрастания, мы получаем вектор $(-1, 0{,}5, 1, 2, 6)$. Теперь $\pi(\mathbf{y})_i$ – позиция y_i в отсортированном векторе $(-1, 0{,}5, 1, 2, 6)$. Такая нотация отражает тот факт, что образцы с наивысшим рангом – те, которым соответствуют наибольшие значения в $\pi(\mathbf{y})$.

В обозначениях нашей модели PAC-обучения множество примеров $Z = \bigcup_{r=1}^{\infty}(\mathcal{X}^r \times \mathbb{R}^r)$, а класс гипотез $\mathcal{H}$ – некоторое множество ранжирующих гипотез. Теперь обратимся к описанию функций потерь для ранжирования. Определить их можно разными способами, мы приведем несколько примеров. Во всех примерах определяется $\ell(h, (\bar{\mathbf{x}}, \mathbf{y})) = \Delta(h(\bar{\mathbf{x}}), \mathbf{y})$ для некоторой функции $\Delta : \bigcup_{r=1}^{\infty}(\mathbb{R}^r \times \mathbb{R}^r) \to \mathbb{R}_+$.

- **Бинарная потеря ранжирования:** $\Delta(\mathbf{y}', \mathbf{y}) = 0$, если $\mathbf{y}$ и $\mathbf{y}'$ индуцируют в точности одинаковое ранжирование, и $\Delta(\mathbf{y}', \mathbf{y}) = 1$ в противном случае. Иными словами, $\Delta(\mathbf{y}', \mathbf{y}) = \mathbb{1}_{[\pi(\mathbf{y}') \neq \pi(\mathbf{y})]}$. Эта функция потерь почти никогда не используется на практике, потому что она не отличает случай, когда $\pi(\mathbf{y}')$ почти равна $\pi(\mathbf{y})$, от случая, когда $\pi(\mathbf{y}')$ ничем не похожа на $\pi(\mathbf{y})$.
- **Потеря Кендалла тау:** мы подсчитываем, сколько пар (i, j) встречаются в двух перестановках в разном порядке. Это можно записать в виде такого выражения

$$\Delta(\mathbf{y}', \mathbf{y}) = \frac{2}{r(r-1)} \sum_{i=1}^{r-1} \sum_{j=i+1}^{r} \mathbb{1}_{[\mathrm{sign}(y'_i - y'_j) \neq \mathrm{sign}(y_i - y_j)]}.$$

  Такая функция потерь полезнее бинарной потери, поскольку отражает степень сходства двух ранжирований.
- **Нормированная приведенная суммарная эффективность** (Normalized Discounted Cumulative Gain – NDCG): эта метрика отдает предпочтение правильности верхней части списка путем использования монотонно убывающей функции дисконтирования $D : \mathbb{N} \to \mathbb{R}_+$. Сначала определим приведенную суммарную эффективность:

$$G(\mathbf{y}', \mathbf{y}) = \sum_{i=1}^{r} D(\pi(\mathbf{y}')_i) y_i.$$

  Иными словами, если интерпретировать y_i как оценку «истинной релевантности» элемента i, то мы берем взвешенную сумму релевантностей элементов, где вес y_i определяется, исходя из позиции i в $\pi(\mathbf{y}')$. В предположении, что все элементы $\mathbf{y}$ неотрицательны, легко проверить, что $0 \leq G(\mathbf{y}', \mathbf{y}) \leq G(\mathbf{y}, \mathbf{y})$. Поэтому мы можем определить нормированную при-

веденную суммарную эффективность, поделив на $G(\mathbf{y}, \mathbf{y})$, и соответствующая функция потерь будет иметь вид

$$\Delta(\mathbf{y}', \mathbf{y}) = 1 - \frac{G(\mathbf{y}', \mathbf{y})}{G(\mathbf{y}, \mathbf{y})} = \frac{1}{G(\mathbf{y}, \mathbf{y})} \sum_{i=1}^{r} \left(D(\pi(\mathbf{y})_i) - D(\pi(\mathbf{y}')_i) \right) y_i.$$

Легко видеть, что $\Delta(\mathbf{y}', \mathbf{y}) \in [0, 1]$ и что $\Delta(\mathbf{y}', \mathbf{y}) = 0$, когда $\pi(\mathbf{y}') = \pi(\mathbf{y})$.

$$D(i) = \begin{cases} \dfrac{1}{\log_2(r - i + 2)}, & \text{если } i \in \{r - k + 1, \ldots, r\} \\ 0, & \text{в противном случае} \end{cases},$$

где $k < r$ – параметр. Это означает, что мы уделяем больше внимания элементам с более высоким рангом и полностью игнорируем элементы, не попавшие в первые k. Метрика NDCG часть применяется для оценки качества поисковых систем, поскольку в таких приложениях имеет смысл полностью игнорировать элементы с недостаточно высоким рангом.

Имея класс гипотез и функцию потерь ранжирования, мы можем обучить функцию ранжирования по правилу ERM. Но с вычислительной точки зрения получающаяся задача оптимизации может оказаться трудной. Далее мы обсудим, как обучить линейные предикторы для ранжирования.

17.4.1. Линейные предикторы для ранжирования

Функцию ранжирования можно естественно определить, спроецировав образцы на некоторый вектор $\mathbf{w}$ и отдавая на выходе результирующие скаляры. Иначе говоря, впредположении, что $\mathcal{X} \subset \mathbb{R}^d$, для любого $\mathbf{w} \in \mathbb{R}^d$ определяем функцию ранжирования следующим образом

$$h_{\mathbf{w}}((\mathbf{x}_1, \ldots, \mathbf{x}_r)) = (\langle \mathbf{w}, \mathbf{x}_1 \rangle, \ldots, \langle \mathbf{w}, \mathbf{x}_r \rangle). \tag{17.6}$$

В главе 16 отмечалось, что можно также применить отображение образцов в некоторое пространство признаков, а затем вычислять скалярные произведения с $\mathbf{w}$ в этом пространстве. Для простоты мы ограничимся более простой формой (17.6).

Если дано множество $W \subset \mathbb{R}^d$, то мы теперь можем определить класс гипотез $\mathcal{H}_W = \{h_{\mathbf{w}} : \mathbf{w} \in W\}$. Определив этот класс гипотез и выбрав функцию потерь ранжирования, мы можем применить правило ERM следующим образом: для данного обучающего набор $S = (\bar{\mathbf{x}}_1, \mathbf{y}_1), \ldots, (\bar{\mathbf{x}}_m, \mathbf{y}_m)$, где каждая пара $(\bar{\mathbf{x}}_i, \mathbf{y}_i)$ принадлежит $(\mathcal{X} \times \mathbb{R})^{r_i}$ для некоторого $r_i \in \mathrm{N}$, мы должны найти такой вектор $\mathbf{w} \in W$, который доставляет минимум эмпирической потере $\sum_{i=1}^{m} \Delta(h_{\mathbf{w}}(\bar{\mathbf{x}}_i), \mathbf{y}_i)$. Как и в случае бинарной классификации, для многих функций потерь эта проблема вычислительно трудна, поэтому мы обратимся к выпуклым суррогатным функциям потерь. Мы опишем такие суррогаты для потери Кендалла тау и NDCG.

Кусочно-линейная потеря для функции потерь Кендалла тау

Потерю Кендалла тау можно рассматривать как среднее бинарных потерь для каждой пары. Для каждой пары (i, j) мы можем переписать

$$\mathbb{1}_{[\text{sign}(y'_i - y'_j) \neq \text{sign}(y_i - y_j)]} = \mathbb{1}_{[\text{sign}(y_i - y_j)(y'_i - y'_j) \leq 0]}.$$

В нашем случае $y'_i - y'_j = \langle \mathbf{w}, \mathbf{x}_i - \mathbf{x}_j \rangle$. Следовательно, мы можем получить верхнюю границу, используя кусочно-линейную потерю:

$$\mathbb{1}_{[\text{sign}(y_i - y_j)(y'_i - y'_j) \leq 0]} \leq \max\{0, 1 - \text{sign}(y_i - y_j)\langle \mathbf{w}, \mathbf{x}_i - \mathbf{x}_j \rangle\}.$$

Усредняя по всем парам, получаем такую выпуклую суррогатную потерю для функции потерь Кендалла тау:

$$\Delta(h_{\mathbf{w}}(\bar{\mathbf{x}}), \mathbf{y}) \leq \frac{2}{r(r-1)} \sum_{i=1}^{r-1} \sum_{j=i+1}^{r} \max\{0, 1 - \text{sign}(y_i - y_j)\langle \mathbf{w}, \mathbf{x}_i - \mathbf{x}_j \rangle\}.$$

Правая часть выпукла относительно $\mathbf{w}$ и ограничивает сверху потерю Кендалла тау. Она также является ρ-липшицевой с параметром $\rho \leq \max_{i,j} \|\mathbf{x}_i - \mathbf{x}_j\|$.

Кусочно-линейная потеря для функции потерь NDCG

Функция потерь NDCG зависит от предсказанного вектора ранжирования $\mathbf{y}' \in \mathbb{R}^r$ через индуцированную им перестановку. Чтобы вывести суррогатную функцию потерь, мы сначала сделаем следующее наблюдение. Пусть V – множество всех перестановок $[r]$, представленных в виде векторов, т. е. каждая $\mathbf{v} \in V$ – это вектор в $[r]^r$ такой, что для всех $i \neq j$ имеем $v_i \neq v_j$. Тогда (см. упражнение 17.4)

$$\pi(\mathbf{y}') = \operatorname*{argmax}_{\mathbf{v} \in V} \sum_{i=1}^{r} v_i y'_i. \tag{17.7}$$

Обозначим $\Psi(\bar{\mathbf{x}}, \mathbf{v}) = \sum_{i=1}^{r} v_i \mathbf{x}_i$, тогда

$$\begin{aligned} \pi(h_{\mathbf{w}}(\bar{\mathbf{x}})) &= \operatorname*{argmax}_{\mathbf{v} \in V} \sum_{i=1}^{r} v_i \langle \mathbf{w}, \mathbf{x}_i \rangle \\ &= \operatorname*{argmax}_{\mathbf{v} \in V} \left\langle \mathbf{w}, \sum_{i=1}^{r} v_i \mathbf{x}_i \right\rangle \\ &= \operatorname*{argmax}_{\mathbf{v} \in V} \langle \mathbf{w}, \Psi(\bar{\mathbf{x}}, \mathbf{v}) \rangle. \end{aligned}$$

Принимая во внимание это наблюдение, мы можем использовать обобщенную кусочно-линейную функцию потерь для стоимостной многоклассовой классификации в качестве суррогатной функции потери для потери NDCG:

$$\begin{aligned} \Delta(h_{\mathbf{w}}(\bar{\mathbf{x}}), \mathbf{y}) &\leq \Delta(h_{\mathbf{w}}(\bar{\mathbf{x}}), \mathbf{y}) + \langle \mathbf{w}, \Psi(\bar{\mathbf{x}}, \pi(h_{\mathbf{w}}(\bar{\mathbf{x}}))) \rangle - \langle \mathbf{w}, \Psi(\bar{\mathbf{x}}, \pi(\mathbf{y})) \rangle \\ &\leq \max_{\mathbf{v} \in V} [\Delta(\mathbf{v}, \mathbf{y}) + \langle \mathbf{w}, \Psi(\bar{\mathbf{x}}, \mathbf{v}) \rangle - \langle \mathbf{w}, \Psi(\bar{\mathbf{x}}, \pi(\mathbf{y})) \rangle \\ &= \max_{\mathbf{v} \in V} \left[\Delta(\mathbf{v}, \mathbf{y}) + \sum_{i=1}^{r} (v_i - \pi(\mathbf{y})_i) \langle \mathbf{w}, \mathbf{x}_i \rangle \right]. \end{aligned} \tag{17.8}$$

Правая часть – выпуклая функция относительно $\mathbf{w}$.

Теперь проблему обучения можно решить методом СГС, как описано в разделе 17.2.5. Основное узкое место – вычисление субградиента функции потерь

или, что то же самое, нахождение вектора **v**, доставляющего максимум выражению (17.8) (см. утверждение 14.6). По определению потери NDCG, это эквивалентно решению задачи

$$\operatorname*{argmax}_{\mathbf{v}\in V} \sum_{i=1}^{r} (\alpha_i v_i + \beta_i D(v_i)),$$

где $\alpha_i = -\langle \mathbf{w}, \mathbf{x}_i \rangle$ и $\beta_i = y_i / G(\mathbf{y}, \mathbf{y})$. Мы можем рассматривать эту задачу немного иначе, определив матрицу $A \in \mathbb{R}^{r,r}$, где

$$A_{i,j} = j\alpha_i + D(j)\beta_i.$$

Будем интерпретировать каждое j как «исполнителя», каждое i – как «задачу», а $A_{i,j}$ – как стоимость назначения задачи i исполнителю j. При таком подходе нахождение **v** сводится к отысканию распределения задач между исполнителями, имеющего минимальную стоимость. Эта проблема, называемая «задачей о назначениях», имеет эффективное решение, в частности, с помощью «венгерского алгоритма» (Kuhn, 1955). По-другому задачу о назначениях можно решить методами линейного программирования. Для этого сначала перепишем ее в виде

$$\begin{aligned} \operatorname*{argmin}_{B\in\mathbb{R}_+^{r,r}} & \sum_{i,j=1}^{r} A_{i,j} B_{i,j} \\ \text{s.t. } & \forall i \in [r],\ \sum_{j=1}^{r} B_{i,j} = 1 \\ & \forall j \in [r],\ \sum_{i=1}^{r} B_{i,j} = 1 \\ & \forall i, j,\ B_{i,j} \in \{0, 1\}. \end{aligned} \tag{17.9}$$

Матрица B, удовлетворяющая ограничениям этой задачи оптимизации, называется *матрицей перестановок*. Такое название она получила, потому что ограничения гарантируют, что в каждой строке существует не более одного элемента, равного 1, а в каждом столбце ровно один элемент, равный 1. Поэтому матрице B соответствует перестановка $\mathbf{v} \in V$ такая, что $v_i = j$ для того единственного индекса j, который удовлетворяет условию $B_{i,j} = 1$.

Эта задача оптимизации еще не является линейной программой из-за комбинаторного ограничения $B_{i,j} \in \{0, 1\}$. Однако оказывается, что это ограничение избыточно – если просто опустить комбинаторное ограничение, то по-прежнему гарантируется существование оптимального решения, удовлетворяющего ему. Ниже мы формализуем эту мысль.

Обозначим $\langle A, B \rangle = \sum_{i,j} A_{i,j} B_{i,j}$. Тогда (17.9) становится задачей минимизации $\langle A, B \rangle$, в которой B – матрица перестановок.

Матрица $B \in \mathbb{R}^{r,r}$ называется *дважды стохастической*, если все ее элементы неотрицательны, сумма элементов в каждой строке равна 1 и сумма элементов в каждом столбце равна 1. Поэтому задача (17.9) без ограничений $B_{i,j} \in \{0, 1\}$ эквивалентна задаче

$$\operatorname*{argmin}_{B\in\mathbb{R}^{r,r}} \langle A, B \rangle \text{ такое, что } B \text{ – дважды стохастическая матрица.} \tag{17.10}$$

Следующее утверждение говорит, что любая дважды стохастическая матрица является выпуклой комбинацией матриц перестановки.

Утверждение 17.3 (Birkhoff, 1946; von Neumann, 1953). *Множество дважды стохастических матриц в* $\mathbb{R}^{r,\,r}$ *является выпуклой оболочкой множества матриц перестановки в* $\mathbb{R}^{r,\,r}$.

Из этого утверждения легко получается следующая лемма.

Лемма 17.4. *Существует оптимальное решение задачи (17.10), являющееся одновременно оптимальным решением задачи (17.9).*

Доказательство. Пусть B – решение задачи (17.10). Тогда, в силу утверждения 17.3, можно записать $B = \sum_i \gamma_i C_i$, где для всех iC_i – матрица перестановок, $\gamma_i > 0$ и $\sum_i \gamma_i = 1$. Так как все матрицы C_i дважды стохастические, то очевидно, что $\langle A, B\rangle \le \langle A, C_i\rangle$ для любого i. Мы утверждаем, что существует такое i, для которого $\langle A, B\rangle = \langle A, C_i\rangle$. Это должно быть так, поскольку иначе, если для всех i $\langle A, B\rangle < \langle A, C_i\rangle$, то мы имели бы

$$\langle A, B\rangle = \left\langle A, \sum_i \gamma_i C_i \right\rangle = \sum_i \gamma_i \langle A, C_i\rangle > \sum_i \gamma_i \langle A, B\rangle = \langle A, B\rangle,$$

что невозможно. Таким образом, мы показали, что некоторая матрица перестановок C_i удовлетворяет условию $\langle A, B\rangle = \langle A, C_i\rangle$. Но поскольку для любой другой матрицы перестановок C имеет место неравенство $\langle A, B\rangle \le \langle A, C_i\rangle$, то мы заключаем, что C_i – оптимальное решение обеих задач (17.9) и (17.10). □

17.5. Двудольное ранжирование и многомерные показатели качества

В предыдущем разделе мы описали проблему ранжирования. Мы использовали вектор $\mathbf{y} \in \mathbb{R}^r$ для представления отношения порядка на элементах $\mathbf{x}_1, \ldots, \mathbf{x}_r$. Если все элементы $\mathbf{y}$ различны, то $\mathbf{y}$ задает отношение полного порядка на $[r]$. Но если какие-то два элемента $\mathbf{y}$ совпадают, т. е. $y_i = y_j$ при $i \ne j$, то $\mathbf{y}$ задает только отношение частичного порядка на $[r]$. В таком случае говорят, что $\mathbf{x}_i$ и $\mathbf{x}_j$ обладают одинаковой релевантностью согласно $\mathbf{y}$. В экстремальном случае $\mathbf{y} \in \{\pm 1\}^r$, т. е. каждый $\mathbf{x}_i$ либо релевантен, либо нерелевантен. Такую конфигурацию часто называют «двудольным ранжированием». Например, в упомянутом выше приложении для обнаружения мошенничества каждая транзакция помечается либо как мошенническая ($y_i = 1$), либо как честная ($y_i = -1$).

На первый взгляд кажется, что проблему двудольного ранжирования можно решить, если обучить бинарный классификатор, применить его к каждому образцу и поместить положительные образцы в начало ранжированного списка. Однако при этом могут получиться плохие результаты, потому что цель бинарного обучаемого обычно состоит в том, чтобы минимизировать бинарную потерю (или какой-то ее суррогат), тогда как у ранжировщика цель может быть совершенно другой. Для иллюстрации снова рассмотрим проблему обнаружения мошенничества. Обычно большинство транзакций честные (скажем, 99,9%).

Поэтому бинарный классификатор, который предсказывает «честная» для всех транзакций, при бинарной функции потерь будет иметь ошибку 0,1 %. Это очень мало, но получающийся предиктор совершенно бесполезен для обнаружения мошеннических транзакций. Корень зла в том, что бинарная потеря не подходит для решаемой задачи. Более адекватный показатель качества должен был бы принимать во внимание предсказания на всем множестве образцов. Например, в предыдущем разделе мы определили потерю NDCG, которая ставит на первое место корректность элементов с высоким рангом. А сейчас мы опишем дополнительные функции потерь, специально «заточенные» под проблемы двудольного ранжирования.

Как и ранее, нам дана последовательность образцов $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_r)$, и мы предсказываем ранжирующий вектор $\mathbf{y}' \in \mathbb{R}^r$. Вектор обратной связи имеет вид $\mathbf{y} \in \{\pm 1\}^r$. Мы определим потерю, которая зависит от $\mathbf{y}'$, $\mathbf{y}$ и от порогового значения $\theta \in \mathbb{R}$. Этот порог преобразует вектор $\mathbf{y}' \in \mathbb{R}^r$ в вектор $(\text{sign}(y'_i - \theta), \ldots, \text{sign}(y'_r - \theta)) \in \{\pm 1\}^r$. Обычно θ полагают равным 0. Но, как мы увидим, иногда θ выбирают с учетом дополнительных ограничений, присутствующих в задаче.

Определяемые ниже функции потерь зависят от следующих 4 чисел:

$$\begin{array}{ll} \text{Истинно положительные результаты:} & a = |\{i : y_i = +1 \wedge \text{sign}(y'_i - \theta) = +1\}| \\ \text{Ложноположительные результаты:} & a = |\{i : y_i = -1 \wedge \text{sign}(y'_i - \theta) = +1\}| \\ \text{Ложноотрицательные результаты:} & a = |\{i : y_i = +1 \wedge \text{sign}(y'_i - \theta) = -1\}| \\ \text{Истинно отрицательные результаты:} & a = |\{i : y_i = -1 \wedge \text{sign}(y'_i - \theta) = -1\}| \end{array} \quad (17.11)$$

Полнотой (recall) (или **чувствительностью**) вектора предсказаний называется доля «улавливаемых» $\mathbf{y}'$ истинно положительных результатов, т. е. $a/(a + c)$. **Точностью** (precision) называется доля правильных предсказаний среди всех предсказанных положительных меток, т. е. $a/(a + b)$. **Специфичностью** (specificity) называется доля «улавливаемых» предиктором истинно отрицательных результатов, т. е. $d/(d + b)$.

Отметим, что с уменьшением θ полнота увеличивается (и достигает значения 1, когда $\theta = -\infty$). С другой стороны, точность и специфичность обычно уменьшаются с уменьшением θ. Поэтому между точностью и полнотой существует компромисс, управляемый параметром θ. В определенных ниже функциях потерь используются различные способы комбинирования точности и полноты.

- **Усреднение чувствительности и специфичности.** Эта метрика определяется формулой $\frac{1}{2}(a/(a+c) + d/(d+b))$. Иначе говоря, это среднее арифметическое верности на положительных примерах и верности на отрицательных примерах. В данном случае мы полагаем $\theta = 0$, а соответствующая функция потерь имеет вид $\Delta(\mathbf{y}', \mathbf{y}) = 1 - \frac{1}{2}(a/(a+c) + d/(d+b))$.
- **Мера F_1.** Это среднее гармоническое точности и полноты: $\dfrac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$.

  Она достигает максимального значения 1, когда точность и полнота равны 1, а минимального значения 0, когда хотя бы одна из этих метрик равна 0 (пусть даже вторая при этом равна 1). Меру F_1 можно выразить через a, b, c следующим образом: $F_1 = 2a/(2a + b + c)$. И снова мы полагаем $\theta = 0$, так что функция потерь принимает вид $\Delta(\mathbf{y}', \mathbf{y}) = 1 - F_1$.

- **Мера F_β.** Похожа на меру F_1, но мы приписываем полноте в β^2 раз большую важность, чем точности, т. е. $\frac{1+\beta^2}{\frac{1}{\text{Precision}}+\beta^2\frac{1}{\text{Recall}}}$. Ее можно записать также в виде $F_\beta = \frac{(1+\beta^2)a}{(1+\beta^2)a+b+\beta^2 c}$. Параметр θ снова равен 0, а функция потерь принимает вид $\Delta(\mathbf{y}', \mathbf{y}) = 1 - F_\beta$.
- **Полнота при k.** Мы измеряем полноту при условии, что предсказание содержит не более k положительных меток. То есть мы должны установить θ так, чтобы $a + b \le k$. Это удобно, например, в приложении для обнаружения мошенничества, чтобы сотрудник банка обрабатывал небольшое количество подозрительных транзакций.
- **Точность при k.** Мы измеряем точность при условии, что предсказание содержит не менее k положительных меток. То есть мы должны установить θ, так чтобы $a + b \ge k$.

ЛАНЬ®

Определенные выше метрики часто называют *многомерными показателями качества*. Отметим, что эти метрики сильно отличаются от средней бинарной потери, которая в принятых обозначениях равна $(b + d)/(a + b + c + d)$. В вышеупомянутом примере обнаружения мошенничества в случае, когда 99,9% примеров помечены отрицательно, бинарная потеря предсказания, согласно которому все вообще примеры отрицательны, составляет всего 0,1%. Напротив, полнота такого предсказания равна нулю, а значит, мера F_1 тоже равна 0, и, следовательно, соответствующая потеря равна 1.

17.5.1. Линейные предикторы для двудольного ранжирования

ЛАНЬ®

Опишем далее, как обучить линейные предикторы для двудольного ранжирования. Как и в предыдущем разделе, линейный предиктор для ранжирования определяется в виде

$$h_{\mathbf{w}}(\bar{\mathbf{x}}) = (\langle \mathbf{w}, \mathbf{x}_1 \rangle, \ldots, \langle \mathbf{w}, \mathbf{x}_r \rangle).$$

Соответствующая функция потерь – один из описанных выше многомерных показателей качества. Функция потерь зависит от вектора $\mathbf{y}' = h_{\mathbf{w}}(\bar{\mathbf{x}})$ через индуцируемый им двоичный вектор, который мы обозначим

$$\mathbf{b}(\mathbf{y}') = (\text{sign}(y'_1 - \theta), \ldots, \text{sign}(y'_r - \theta)) \in \{\pm 1\}^r. \tag{17.12}$$

Как и в предыдущем разделе, для обеспечения эффективности алгоритма мы подберем выпуклую суррогатную потерю для Δ. Рассуждение будет примерно таким же, как при выводе обобщенной кусочно-линейной потери для потери ранжирования NDCG.

Прежде всего, заметим, что для любого определенного выше значения θ существует $V \subseteq \{\pm 1\}^r$ такое, что $\mathbf{b}(\mathbf{y}')$ можно переписать в виде

$$\mathbf{b}(\mathbf{y}') = \operatorname*{argmax}_{\mathbf{v} \in V} \sum_{i=1}^{r} v_i y'_j. \tag{17.13}$$

Это очевидно так в случае $\theta = 0$, если взять $V = \{\pm 1\}^r$. Есть две метрики, для которых берется θ, не равное 0: точность при k и полнота при k. В случае точности при k мы можем взять в качестве V множество $V_{\geq k}$, содержащие все векторы из $\{\pm 1\}^r$, в которых количество единиц не меньше k. В случае полноты при k в качестве V множество $V_{\leq k}$, определенное аналогично. См. упражнение 17.5.

Определив $\mathbf{b}$, как в (17.13), мы можем построить выпуклую суррогатную потерю следующим образом. В предположении, что $\mathbf{y} \in V$, имеем

$$\begin{aligned}\Delta(h_{\mathbf{w}}(\bar{\mathbf{x}}), \mathbf{y}) &= \Delta(\mathbf{b}(h_{\mathbf{w}}(\bar{\mathbf{x}})), \mathbf{y}) \\ &\leq \Delta(\mathbf{b}(h_{\mathbf{w}}(\bar{\mathbf{x}})), \mathbf{y}) + \sum_{i=1}^{r} (b_i(h_{\mathbf{w}}(\bar{\mathbf{x}})) - y_i)\langle \mathbf{w}, \mathbf{x}_i \rangle \\ &\leq \max_{\mathbf{v} \in V} \left[\Delta(\mathbf{v}, \mathbf{y}) + \sum_{i=1}^{r} (v_i - y_i)\langle \mathbf{w}, \mathbf{x}_i \rangle \right]. \end{aligned} \tag{17.14}$$

Правая часть – выпуклая функция относительно $\mathbf{w}$.

Теперь мы можем решить проблему обучения с помощью СГС, как описано в разделе 17.2.5. Главная трудность – вычисление субградиента функции потерь, эквивалентное нахождению вектора $\mathbf{v}$, который доставляет максимум функции (17.14) (см. утверждение 14.6).

Ниже мы опишем, как эффективно найти такой вектор для любого показателя качества, который можно записать в виде функции от чисел a, b, c, d, определенных в (17.11), и для которого множество V содержит все элементы $\{\pm 1\}^r$, для которых значения a и b удовлетворяют некоторым ограничениям. Например, в случае «полноты при k» множество V состоит из всех векторов, для которых $a + b \leq k$.

Идея заключается в следующем. Для любых $a, b \in [r]$ положим

$$\bar{\mathcal{Y}}_{a,b} = \{\mathbf{v} : |\{i : v_i = 1 \wedge y_i = 1\}| = a \wedge |\{i : v_i = 1 \wedge y_i = -1\}| = b\}.$$

Любой вектор $\mathbf{v} \in V$ попадает в $\bar{\mathcal{Y}}_{a,b}$ для некоторых $a, b \in [r]$. Кроме того, если $\bar{\mathcal{Y}}_{a,b} \cap V$ непусто для некоторых $a, b \in [r]$, то $\bar{\mathcal{Y}}_{a,b} \cap V = \bar{\mathcal{Y}}_{a,b}$. Следовательно, мы можем искать внутри всех $\bar{\mathcal{Y}}_{a,b}$, имеющих непустое пересечение с V, по отдельности, а затем взять оптимальное значение. Ключевое наблюдение состоит в том, что при поиске только внутри $\bar{\mathcal{Y}}_{a,b}$ значение Δ фиксировано, поэтому остается лишь максимизировать выражение

$$\max_{\mathbf{v} \in \bar{\mathcal{Y}}_{a,b}} \sum_{i=1}^{r} v_i \langle \mathbf{w}, \mathbf{x}_i \rangle.$$

Предположим, что примеры отсортированы таким образом, что $\langle \mathbf{w}, \mathbf{x}_1 \rangle \geq \ldots \geq \langle \mathbf{w}, \mathbf{x}_r \rangle$. Тогда легко проверить, что нам выгодно сделать v_i положительными для наименьших индексов i. С учетом ограничений на a и b это означает, что нужно положить $v_i = 1$ для a положительных примеров с наивысшим рангом и для b отрицательных примеров с наивысшим рангом. В результате мы приходим к такой процедуре.

Решение задачи (17.14)

вход:
$(\mathbf{x}_1, \ldots, \mathbf{x}_r)$, $(y_1, \ldots, y_r)$, $\mathbf{w}$, V, Δ
предположения:
Δ является функцией от a, b, c, d
V содержит все векторы, для которых $f(a, b) = 1$ для некоторой функции f
инициализация:
$P = |\{i: y_i = 1\}|$, $N = |\{i: y_i = -1\}|$
$\boldsymbol{\mu} = (\langle \mathbf{w}, \mathbf{x}_1 \rangle, \ldots, \langle \mathbf{w}, \mathbf{x}_r \rangle)$, $\alpha^* = -\infty$
отсортировать примеры, так чтобы $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_r$
пусть $i_1, \ldots, i_P$ – индексы положительных примеров (после сортировки)
пусть $j_1, \ldots, j_N$ – индексы отрицательных примеров (после сортировки)
for $a = 0, 1, \ldots, P$
  $c = P - a$
  for $b = 0, 1, \ldots, N$ таких, что $f(a, b) = 1$
    $d = N - b$
    вычислить Δ по a, b, c, d
    положить $v_1, \ldots, v_r$, так что $v_{i_1} = \ldots = v_{i_a} = v_{j_1} = \ldots = v_{j_b} = 1$,
      а остальные элементы $\mathbf{v}$ положить равными -1
    положить $\alpha = \Delta + \sum_{i=1}^{r} v_i \mu_i$
    if $\alpha \geq \alpha^*$
      $\alpha^* = \alpha$, $\mathbf{v}^* = \mathbf{v}$
выход: $\mathbf{v}^*$

17.6. Резюме

Многие практические проблемы обучения с учителем можно сформулировать как обучение многоклассового предиктора. Мы начали эту главу с описания того, как можно свести многоклассовое обучение к бинарному. Затем мы описали и проанализировали семейство линейных предикторов для многоклассового обучения. Мы показали, что это семейство можно использовать, даже если количество классов очень велико, при условии, что задача имеет подходящую структуру. Наконец, мы описали проблемы ранжирования. В главе 29 мы изучим выборочную сложность многоклассового обучения более детально.

17.7. Библиографические сведения

Сведение методами «один против всех» и «все пары» было унифицировано в рамках системы ECOC (Error Correction Output Codes – выходные коды, исправляющие ошибки) (Dietterich & Bakiri, 1995; Allwein, Schapire & Singer, 2000). Существуют и другие способы сведения, в частности, древовидные классифи-

каторы (см., например, Beygelzimer, Langford & Ravikumar, 2007). Ограничения методов сведения были изучены в работе Daniely et al., 2011; Daniely et al., 2012. См. также главу 29, в которой мы анализируем выборочную сложность многоклассового обучения.

Прямые подходы к многоклассовому обучению на основе линейных предикторов изучались в работах Vapnik, 1998; Weston & Watkins, 1999; Crammer & Singer, 2001. В частности, многовекторное построение описано в работе Crammer and Singer (2001).

В работе Collins (2000) показано, как применить алгоритм перцептрона к проблемам со структурированным выходом. См. также Collins (2002). С этим подходом тесно связано дискриминантное обучение условных случайных полей, см. Lafferty et al. (2001). Алгоритм SVM со структурированным выходом изучался в работах Collins, 2002; Taskar et al., 2003,; Tsochantaridis et al., 2004.

Динамическая процедура вычисления предсказания $h_{\mathbf{w}}(\mathbf{x})$, изложенная нами в разделе о структурированном выходе, похожа на прямые–обратные переменные, вычисляемые процедурой Витерби в HMM (см., например, Rabiner & Juang, 1986). Вообще, решение задачи максимизации в постановке со структурированным выходом тесно связано с проблемой вывода в графических моделях (см. например, Koller & Friedman, 2009a).

В работе Chapelle, Le and Smola (2007) предложено обучать функцию ранжирования относительно потери NDCG с использованием идей, заимствованных из обучения со структурированным выходом. Авторы также подметили, что задача максимизации в определении обобщенной кусочно-линейной потери эквивалентна задаче о назначениях.

В работе Agarwal and Roth (2005) проанализирована выборочная сложность двудольного ранжирования. Joachims (2005) изучал применимость алгоритма SVM со структурированным выходом к двудольному ранжированию с многомерными показателями качества.

17.8. Упражнения

17.1. Рассмотрим множество S примеров в $\mathbb{R}^n \times [k]$, для которых существуют векторы $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ такие, что любой пример $(\mathbf{x}, y) \in S$ принадлежит шару с центром в точке $\boldsymbol{\mu}_y$ и радиусом $r \geq 1$. Предположим также, что для любых $i \neq j \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\| \geq 4r$. Рассмотрим дополнение каждого образца константой 1 и последующее многовекторное построение:

$$\Psi(\mathbf{x}, y) = [\underbrace{0, \ldots, 0}_{\in \mathbb{R}^{(y-1)(n+1)}}, \underbrace{x_1, \ldots, x_n, 1}_{\in \mathbb{R}^{(n+1)}}, \underbrace{0, \ldots, 0}_{\in \mathbb{R}^{(k-y)(n+1)}}].$$

Покажите, что существует вектор $\mathbf{w} \in \mathbb{R}^{k(n+1)}$ такой, что $\ell(\mathbf{w}, (\mathbf{x}, y)) = 0$ для всех $(\mathbf{x}, y) \in S$.

Указание. Заметим, что любой пример $(\mathbf{x}, y) \in S$ можно записать в виде $\mathbf{x} = \boldsymbol{\mu}_y + \mathbf{v}$ для некоторого вектора $\mathbf{v}$ с нормой $\|\mathbf{v}\| \leq r$. Теперь возьмем $\mathbf{w} = [\mathbf{w}_1, \ldots, \mathbf{w}_k]$, где $\mathbf{w}_i = [\boldsymbol{\mu}_i, -\|\boldsymbol{\mu}_i\|^2/2]$.

17.2. Многоклассовый перцептрон. Рассмотрим следующий алгоритм:

Многоклассовый пакетный перцептрон

вход:
обучающий набор $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
зависящее от класса отображение в пространство признаков
$\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$
инициализация: $\mathbf{w}^{(1)} = (0, \ldots, 0) \in \mathbb{R}^d$
for $t = 1, 2, \ldots$
if ($\exists i$ и $y \neq y_i$ такой, что $\langle \mathbf{w}^{(t)}, \Psi(\mathbf{x}_i, y_i)\rangle \leq \langle \mathbf{w}^{(t)}, \Psi(\mathbf{x}_i, y)\rangle$) **then**
$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \Psi(\mathrm{x}_i, y_i) - \Psi(\mathrm{x}_i, y)$
else
вывести $\mathbf{w}^{(t)}$

Докажите следующую теорему.

Теорема 17.5. *Предположим, что существует вектор* $\mathbf{w}^*$ *такой, что для всех* i *и для всех* $y \neq y_i$ *имеет место неравенство* $\langle \mathbf{w}^*, \Psi(\mathbf{x}_i, y_i)\rangle \geq \langle \mathbf{w}^*, \Psi(\mathbf{x}_i, y)\rangle + 1$. *Обозначим* $R = \max_{i,y} ||\Psi(\mathbf{x}_i, y_i) - \Psi(\mathbf{x}_i, y)||$. *Тогда алгоритм многоклассового перцептрона останавливается после самое большее* $(R||\mathbf{w}^*||)^2$ *итераций, и в момент остановки справедливо утверждение:* $\forall i \in [m] y_i = \mathrm{argmax}_y \langle \mathbf{w}^{(t)}, \Psi(\mathbf{x}_i, y_i)\rangle$.

17.3. Обобщите процедуру динамического программирования из раздела 17.3 на решение задачи максимизации, встречающейся в определении $\hat{h}$ в процедуре СГС для многоклассового предсказния. Можете предполагать, что $\Delta(\mathbf{y}', \mathbf{y}) = \sum_{t=1}^{r} \delta(y'_t, y_t)$ для некоторой произвольной функции δ.

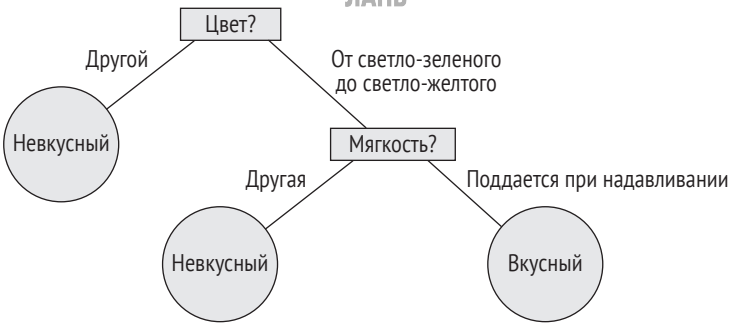
17.4. Докажите справедливость равенства (17.7).

17.5. Покажите, что два определения π – (17.12) и (17.13) – действительно эквивалентны для всех многомерных показателей качества.

Глава 18

РЕШАЮЩИЕ ДЕРЕВЬЯ

Решающее дерево – это предиктор $h : \mathcal{X} \to \mathcal{Y}$, который предсказывает метку, ассоциированную с образцом **x**, проходя от корня дерева к узлу. Для простоты мы ограничимся бинарной классификацией, т. е. $\mathcal{Y} = \{0, 1\}$, но вообще решающие деревья применимы и к другим проблемам предсказания. В каждом узле на пути от корня к листу следующий узел выбирается, исходя из разбиения пространства входов. Обычно разбиение производится по одному из признаков **x** или на основе предопределенного набора правил. В каждом узле хранится конкретная метка. На рисунке ниже приведено решающее дерево для примера о вкусе плодов папайи (см. главу 2):



Чтобы определить будет ли данный плод вкусным, решающее дерево сначала проверяет цвет папайи. Если цвет не находится в диапазоне от светло-зеленого до светло-желтого, то дерево сразу предсказывает, что плод невкусный, не выполняя дополнительных проверок. В противном случае дерево переходит к проверке мягкости папайи. Если она мягкая настолько, что немного поддается при надавливании, то дерево предсказывает, что плод вкусный, иначе что невкусный. В этом примере важно обратить внимание на основное достоинство решающих деревьев – получающийся классификатор очень просто понять и интерпретировать.

18.1. Выборочная сложность

Для разбиения во внутренних узлах часто выбирают правило, основанное на пороговом значении одного признака. То есть мы переходим к правому или левому дочернему узлу, исходя из правила $\mathbb{1}_{[x_i<\theta]}$, где $i \in [d]$ – индекс релевантного признака, а $\theta \in \mathbb{R}$ – пороговое значение. В таких случаях мы можем считать, что решающее дерево разбивает пространство образцов $\mathcal{X} = \mathbb{R}^d$ на ячейки, так что каждый листовый узел соответствует одной ячейке. Отсюда следует, что решающее дерево с k листьями может разделить множество, насчитывающее k образцов. Поэтому если допустить решающие деревья произвольного размера, то мы получим класс гипотез бесконечной VC-размерности. При таком подходе дело легко может закончиться переобучением.

Чтобы избежать переобучения, мы можем воспользоваться принципом минимальной длины описания (MDL), описанным в главе 7, и поставить целью обучить такое решающее дерево, которое, с одной стороны, хорошо аппроксимирует данные, а с другой стороны – не слишком велико.

Для простоты будем предполагать, что $\mathcal{X} = \{0, 1\}^d$. Иными словами, каждый образец – это вектор длиной d бит. В таком случае бинаризация значений одного признака соответствует правилу разбиения вида $\mathbb{1}_{[x_i=1]}$ для некоторого $i \in [d]$. Например, мы можем смоделировать «решающее дерево для папайи», считая, что папайя параметризована двумерным битовым вектором $\mathbf{x} \in \{0, 1\}^2$, где бит x_1 представляет цвет: в диапазоне от светло-зеленого до светло-желтого или нет, а бит x_2 – мягкость: поддается легкому надавливанию или нет. При таком представлении узел «Цвет?» можно заменить на $\mathbb{1}_{[x_1=1]}$, а узел «Мягкость?» – на $\mathbb{1}_{[x_2=1]}$. Хотя это заметное упрощение, алгоритмы и их анализ, представленные ниже, допускают обобщение.

С вышеупомянутым упрощающим предположением, класс гипотез становится конечным, но все равно очень большим. В частности, любой классификатор, отображающий $\{0, 1\}^d$ в $\{0, 1\}$, можно представить решающим деревом с 2^d листьями глубиной $d + 1$ (см. упражнение 18.1). Поэтому VC-размерность класса равна 2^d, а это означает, что количество примеров, необходимых для его PAC-обучения, растет как 2^d. То есть при сколько-нибудь большом d требуется очень много примеров.

Чтобы преодолеть это препятствие, мы обратимся к схеме MDL, описанной в главе 7. Априорным знанием в данном случае будет тот факт, что мы предпочитаем маленькие деревья большим. Чтобы формализовать эту интуитивную идею, нам сначала понадобится определить префиксно-свободный язык описания решающих деревьев, который требует тем меньше бит, чем меньше дерево. Вот один из возможных способов. Дерево с n узлами будет описываться $n + 1$ блоками размера $\log_2(d + 3)$ каждый. Первые n блоков кодируют узлы дерева в прямом порядке обхода в глубину, а последний блок отмечает конец кода. Каждый блок содержит следующую информацию о текущем узле:

- это внутренний узел вида $\mathbb{1}_{[x_i=1]}$ для некоторого $i \in [d]$;
- это листовый узел с значением 1;
- это листовый узел с значением 0;
- это конец кода.

Всего существует $d + 3$ вариантов, поэтому для описания каждого блока требуется $\log 2(d + 3)$ бит.

В предположении, что у каждого внутреннего узла есть два дочерних узла[1], нетрудно показать, что это префиксно-свободный код и что длина описания дерева с n узлами равна $(n + 1)\log_2(d + 3)$.

По теореме 7.7 с вероятностью не менее $1 - \delta$ для случайной выборки размера m для любого n и любого решающего дерева $h \in \mathcal{H}$ с n узлами

$$L_{\mathcal{D}}(h) \leq L_S(h) + \sqrt{\frac{(n+1)\log_2(d+3) + \log(2/\delta)}{2m}}. \tag{18.1}$$

Эта граница знаменует компромисс: с одной стороны, мы ожидаем, что для большого и более сложного дерева риск обучения $L_S(h)$ меньше, но соответствующее значение n больше. С другой стороны, у небольшого решающего дерева меньше значение n, но больше $L_S(h)$. Наши надежды (или априорное знание) связаны с тем, что мы сможем найти решающее дерево, для которого и эмпирический риск $L_S(h)$ мал, и количество узлов n не слишком велико. Приведенная выше граница показывает, что у такого дерева будет мал истинный риск $L_{\mathcal{D}}(h)$.

18.2. Алгоритмы на решающих деревьях

Граница $L_{\mathcal{D}}(h)$ в неравенстве (18.1) подсказывает правило обучения решающих деревьев – искать дерево, которое минимизирует правую часть (18.1). К сожалению, оказывается, что эта проблема вычислительно трудна[2]. Поэтому практические алгоритмы обучения решающих деревьев основываются на различных эвристиках, например, применяется жадный подход, когда дерево строится постепенно, и при построении каждого узла принимаются локально оптимальные решения. Такой алгоритм не может гарантировать, что будет построено глобально оптимальное дерево, но на практике обычно дает достаточно хорошие результаты.

Общая схема «выращивания» решающего дерева такова. Мы начинаем с дерева, содержащего всего один лист (корень) и назначаем этому листу метку, получившую большинство голосов среди всех меток в обучающем наборе. После этого выполняется ряд итераций. На каждой итерации проверяется эффект разбиения в одном узле. Мы определяем некоторую меру «выигрыша», которая количественно выражает улучшение, достигнутое благодаря такому разбиению. Затем из всех возможных разбиений мы либо выбираем и выполняем то, которое максимизирует выигрыш, либо не производим разбиение в этом листе вовсе.

Ниже предлагается одна из возможных реализаций. Она основана на популярном алгоритме построения решающего дерева ID3 (аббревиатура «Iterative

[1] Такое предположение можно сделать без ограничения общности, потому что если некоторый узел имеет только один дочерний узел, то его можно заменить дочерним узлом, и предсказание дерева при этом не изменится.

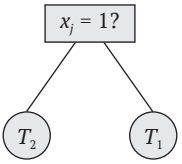
[2] Точнее, если NP ≠ P, то никакой алгоритм не сможет решить задачу (18.1) за время, полиномиально зависящее от n, d и m.

Dichotomizer 3»). Мы опишем этот алгоритм для случая бинарных признаков, т. е. $\mathcal{X} = \{0, 1\}^d$, поэтому все правила разбиения будут иметь вид $\mathbb{1}_{[x_i=1]}$ для некоторого $i \in [d]$. Случай вещественных признаков мы опишем в разделе 18.2.3.

Алгоритм работает рекурсивно, он начинается с вызова ID3(S, $[d]$) и возвращает решающее дерево. В показанном ниже псевдокоде встречаются обращения к процедуре Gain(S, i), которая получает обучающий набор S и индекс i и вычисляет выигрыш от разбиения дерева по i-му признаку. В разделе 18.2.1 мы опишем несколько вариантов выигрыша.

ID3(S, A)

вход: обучающий набор S, подмножество признаков $A \subseteq [d]$
if все примеры в S помечены меткой 1, вернуть лист 1
if все примеры в S помечены меткой 0, вернуть лист 0
if $A = \emptyset$, вернуть лист, значение в котором равно большинству меток в S
else :
  пусть $j = \text{argmax}_{i \in A}$ Gain(S, i)
  if все примеры в S имеют одну и ту же метку
    вернуть лист, значение в котором равно большинству меток в S
  else
    пусть T_1 – дерево, возвращенное вызовом ID3($\{(\mathbf{x}, y) \in S : x_j = 1\}$, $A \backslash \{j\}$)
    пусть T_2 – дерево, возвращенное вызовом ID3($\{(\mathbf{x}, y) \in S : x_j = 0\}$, $A \backslash \{j\}$)
    вернуть дерево:



18.2.1. Реализации меры выигрыша

В разных алгоритмах используются различные реализации Gain(S, i). Мы представим три из них. Будем использовать нотацию $\mathbb{P}_S[F]$ для обозначения вероятности события при равномерном распределении на S.

Ошибка обучения. Простейшее определение выигрыша – уменьшение ошибки обучения. Формально, обозначим $C(a) = \min\{a, 1 - a\}$. Заметим, что ошибка обучения до разбиения по признаку i равна $C(\mathbb{P}_S[y = 1])$, поскольку мы проводим голосование большинством меток. Аналогично ошибка после разбиения по признаку i равна

$$\mathop{\mathbb{P}}_{S}[x_i = 1] C(\mathop{\mathbb{P}}_{S}[y = 1 | x_i = 1]) + \mathop{\mathbb{P}}_{S}[x_i = 0] C(\mathop{\mathbb{P}}_{S}[y = 1 | x_i = 0]).$$

Мы можем определить Gain как разность между этими величинами:

$$\text{Gain}(S, i) := C(\mathop{\mathbb{P}}_{S}[y = 1]) - (\mathop{\mathbb{P}}_{S}[x_i = 1] C(\mathop{\mathbb{P}}_{S}[y = 1 | x_i = 1]) + \mathop{\mathbb{P}}_{S}[x_i = 0] C(\mathop{\mathbb{P}}_{S}[y = 1 | x_i = 0])).$$

Информационный выигрыш. Еще одна популярная мера выигрыша, используемая в алгоритме ID3 и в алгоритме C4.5 из работы Quinlan (1993) – информационный выигрыш, т. е. разность между энтропией метки до и после разбиения. Для этого нужно заменить функцию C в предыдущем выражении функцией

$$C(a) = -a\log(a) - (1 - a)\log(1 - a).$$

Индекс Джини. Еще одно определение выигрыша, используемое в алгоритме CART из работы Breiman, Friedman, Olshen, and Stone (1984), – индекс Джини

$$C(a) = 2a(1 - a).$$

И информационный выигрыш, и индекс Джини – гладкие и выпуклые верхние границы для ошибки обучения. В некоторых ситуациях эти свойства оказываются полезными (см., например, Kearns & Mansour (1996)).

18.2.2. Редукция

Описанный выше алгоритм ID3 все еще подвержен серьезной проблеме: возвращенное им дерево обычно очень велико. У таких деревьев может быть низкий эмпирический риск, но их истинный риск, как правило, высок – это показывает и теоретический анализ, и практическое применение. Одно из решений – ограничить количество итераций ID3, оставив дерево с ограниченным количеством узлов. Другое часто применяемое решение – *редуцировать* (prune) дерево после построения в надежде получить гораздо меньшее дерево, характеризуемое приблизительно такой же эмпирической ошибкой. Теоретически, согласно верхней границе (18.1), если мы сможем сделать n гораздо меньше, не увеличивая намного $L_S(h)$, то, вероятно, получим решающее дерево с меньшим истинным риском.

Обычно редукция выполняется путем обхода дерева снизу вверх. Каждый узел можно заменить одним из его поддеревьев или листом в зависимости от некоторой границы или оценки $L_D(h)$ (например, оценки (18.1)). Ниже приведен псевдокод типичного шаблонного алгоритма.

Общая процедура редукции дерева

вход:
  функция $f(T, m)$ (граница или оценка ошибки обобщения решающего дерева T при размере выборки m)
  дерево T
foreach узел j в порядке обхода дерева T снизу вверх (от листьев к корню)
  найти T', минимизирующее $f(T', m)$, где T' – любое из:
    текущее дерево после замены узла j узлом 1
    текущее дерево после замены узла j узлом 0
    текущее дерево после замены узла j его левым поддеревом
    текущее дерево после замены узла j его правым поддеревом
    текущее дерево
  положить $T := T'$

18.2.3. Пороговые правила разбиения для вещественных признаков

В предыдущем разделе мы описали алгоритм «выращивания» решающего дерева в предположении, что все признаки бинарные, а правила разбиения имеют вид $\mathbb{1}_{[x_i=1]}$. Теперь мы обобщим этот результат на случай вещественных признаков и пороговых правилах разделения вида $\mathbb{1}_{[x_i<\theta]}$. Такие правила порождают решающие пни, мы изучали их в главе 10.

Основная идея заключается в том, чтобы свести задачу к случаю бинарных признаков, и делается это следующим образом. Пусть $\mathbf{x}_1, \ldots, \mathbf{x}_m$ – образцы из обучающего набора. Для каждого вещественного признака i отсортируем образцы, так чтобы $x_{1,i} \le \ldots \le x_{m,i}$. Определим множество порогов $\theta_{0,i}, \ldots, \theta_{m+1,i}$, так чтобы $\theta_{j,i} \in (x_{j,i}, x_{j+1,i})$ (где по соглашению $x_{0,i} = -\infty$ и $x_{m+1,i} = \infty$). Наконец, для каждого i и j определим бинарный признак $\mathbb{1}_{[x_i<\theta_{j,i}]}$. Сконструировав эти бинарные признаки, мы можем выполнить процедуру ID3, описанную в предыдущем разделе. Легко проверить, что для любого решающего дерева с пороговыми правилами разбиения и исходными вещественными признаками существует решающее дерево со сконструированными бинарными признаками, с таким же количеством узлов и с такой же ошибкой обучения.

Если исходное количество вещественных признаков равно d, а количество примеров равно m, то количество сконструированных бинарных признаков будет равно dm. Поэтому вычисление выигрыша каждого признака может потребовать $O(dm^2)$ операций. Но если использовать более изобретательную реализацию, то время работы можно сократить до $O(dm\log(m))$. Идея напоминает реализацию ERM для решающих пней, описанную в разделе 10.1.1.

18.3. Случайные леса

Как уже отмечалось, класс решающих деревьев произвольного размера имеет бесконечную VC-размерность. Поэтому мы ограничили размер решающего дерева. Другой способ уменьшить опасность переобучения заключается в построении ансамбля деревьев. В частности, ниже мы опишем метод *случайных лесов*, введенный в работе Breiman (2001).

Случайный лес – это классификатор, состоящий из коллекции решающих деревьев, в которой каждое дерево строится применением алгоритма A к обучающему набору S, и дополнительного случайного вектора θ, элементы которого независимо выбираются из некоторого распределения. Предсказание случайного леса – это результат, за который проголосовало большинство отдельных деревьев.

Чтобы задать конкретный случайный лес, мы должны определить алгоритм A и распределение θ. Это можно сделать разными способами, мы опишем только один. Вектор θ генерируется следующим образом. Сначала производим случайный выбор из S с возвращением, т. е. получаем новый обучающий набор S' размера m', применяя равномерное распределение к S. Затем строим последовательность $I_1, I_2, \ldots$, где каждое I_t – подмножество $[d]$ размера k, сгенерированное путем случайной равномерной выборки элементов из $[d]$. Эти случайные вели-

чины и образуют вектор θ. Далее алгоритм A выращивает решающее дерево (например, с помощью ID3), основываясь на выборке S', причем на каждом шаге разбиения алгоритм вправе выбирать только признак из I_t, максимизирующий выгрыш. Интуитивно очевидно, что если k мало, то это ограничение может предотвратить переобучение.

ЛАНЬ®

18.4. Резюме

Решающие деревья – интуитивно очень понятные предикторы. Если предиктор создается программистом-человеком, то он, скорее всего, будет напоминать решающее дерево. Мы показали, что VC-размерность решающих деревьев с k листьями равна k, и предложили парадигму MDL для обучения деревьев. Основная проблема заключается в том, что обучение решающих деревьев – вычислительно трудная задача, поэтому мы описали несколько эвристических процедур обучения.

18.5. Библиографические сведения

Многие алгоритмы обучения решающих деревьев (в частности, ID3 и C4.5) были описаны в работе Quinlan (1986). Алгоритм CART представлен в работе Breiman, Friedman, Olshen & Stone (1984). Случайные леса предложены Брейманом в работе Breiman (2001). За дополнительной информацией отсылаем читателя к работам Hastie, Tibshirani & Friedman, 2001; Rokach, 2007.

Доказательство трудности обучения решающих деревьев приведено в работе Hyafil and Rivest (1976).

18.6. Упражнения

18.1. 1. Покажите, что любой бинарный классификатор $h : \{0, 1\}^d \mapsto \{0, 1\}$ можно реализовать как решающее дерево высоты не более $d + 1$ с внутренними узлами вида $(x_i = 0?)$ для некоторого $i \in \{1, \ldots, d\}$.

2. Выведите отсюда, что VC-размерность класса решающих деревьев над областью определения $\{0, 1\}^d$ равна 2^d.

ЛАНЬ®

18.2. (**Субоптимальность ID3**) Рассмотрим следующий обучающий набор, где $\mathcal{X} = \{0, 1\}^3$, $\mathcal{Y} = \{0, 1\}$:

((1, 1, 1), 1)
((1, 0, 0), 1)
((1, 1, 0), 0)
((0, 0, 1), 0)

Предположим, что мы хотим использовать этот обучающий набор для построения решающего дерева глубины 2 (т. е. для каждого входного образца нам разрешено задать только два вопроса вида $(x_i = 0?)$ и затем нужно определить метку).

1. Предположим, что мы выполняем алгоритм ID3 до глубины 2 (т. е. выбираем корень и его дочерние узлы в соответствии с алгоритмом, но вместо того чтобы продолжать рекурсию, останавливаемся и выбираем листья согласно мажоритарной метке в каждом поддереве). Предположим, что подпрограмма для измерения качества каждого признака основана на функции энтропии (т. е. мы измеряем информационный выигрыш) и что если два признака дают одинаковую оценку, то мы произвольно выбираем один из них. Покажите, что ошибка обучения получающегося решающего дерева не меньше 1/4.
2. Найдите решающее дерево глубины 2, на котором достигается нулевая ошибка обучения.

ЛАНЬ®

ЛАНЬ®

Глава 19

БЛИЖАЙШИЕ СОСЕДИ

Алгоритмы ближайших соседей относятся к числу простейших алгоритмов машинного обучения. Идея в том, чтобы запомнить весь обучающий набор, а затем предсказывать метку нового образца, ориентируясь на метки ближайших к нему элементов обучающего набора. В обоснование этого метода выдвигается предположение о том, что признаки, использованные для описания точек из области определения, соотносятся с их метками таким образом, что близкие точки с большой вероятностью будут иметь одинаковые метки. Кроме того, в некоторых ситуациях, даже если обучающий набор огромен, найти ближайшего соседа можно очень быстро (например, если обучающим набором является весь веб, а расстояние измеряется количеством ссылок на пути из одного узла в другой).

Отметим, что в отличие от ранее рассмотренных алгоритмических парадигм – ERM, SRM, MDL и RLM, – определяемых некоторым классом гипотез $\mathcal{H}$, метод ближайшего соседа вычисляет метку любой тестовой точки, не прибегая к поиску предиктора в каком-то предопределенном классе функций.

В этой главе мы опишем методы ближайших соседей для проблем классификации и регрессии. Мы проанализируем их качество в простом случае бинарной классификации и обсудим эффективность реализации.

19.1. Метод *k* ближайших соседей

В этой главе мы будем предполагать, что область определения $\mathcal{X}$ наделена метрикой ρ, т. е. на ней определена функция $\rho : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, которая возвращает расстояние между любыми двумя элементами $\mathcal{X}$. Например, если $\mathcal{X} = \mathbb{R}^d$, то ρ может быть евклидовым расстоянием $\rho(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'|| = \sqrt{\sum_{i=1}^{d} (x_i - x_i')^2}$.

Пусть $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ – последовательность обучающих примеров. Для любого $\mathbf{x} \in \mathcal{X}$ пусть $\pi_1(\mathbf{x}), \ldots, \pi_m(\mathbf{x})$ – перестановка множества $\{1, \ldots, m\}$ в соответствии с расстоянием до $\mathbf{x}$, $\rho(\mathbf{x}, \mathbf{x}_i)$, т. е. для любого $i < m$

$$\rho(\mathbf{x}, \mathbf{x}_{\pi_i(\mathbf{x})}) \leq \rho(\mathbf{x}, \mathbf{x}_{\pi_{i+1}(\mathbf{x})}).$$

Если дано число k, то правило k ближайших соседей, k-NN, для бинарной классификации определяется следующим образом:

k-NN

вход: обучающая выборка $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
выход: для любой точки $\mathbf{x} \in \mathcal{X}$
вернуть метку, чаще всего встречающуюся в множестве $\{y_{\pi_i(\mathbf{x})} : i \leq k\}$

При $k = 1$ получаем правило 1-NN:

$$h_S(\mathbf{x}) = y_{\pi_1(\mathbf{x})}.$$

На рис. 19.1 приведена геометрическаяиллюстрация правила 1-NN.

Рис. 19.1. Решающие границы в правиле 1-NN. Изображены выборочные точки, а в качестве метки любой новой точки предсказывается метка выборочной точки, находящейся в центре ячейки, которой новая точка принадлежит. Совокупность этих ячеек называется диаграммой Вороного

Для проблем регрессии, когда $\mathcal{Y} = \mathbb{R}$, в качестве предсказания можно брать среднее арифметическое меток k ближайших соседей, т. е. $h_S(\mathbf{x}) = (1/k)\sum_{i=1}^{k} y_{\pi_i(\mathbf{x})}$. Вообще, если дана некоторая функция $\varphi : (\mathcal{X} \times \mathcal{Y})^k \to \mathcal{Y}$, то правило k-NN относительно φ имеет вид:

$$h_S(\mathbf{x}) = \varphi((\mathbf{x}_{\pi_1(\mathbf{x})}, y_{\pi_1(\mathbf{x})}), \ldots, (\mathbf{x}_{\pi_k(\mathbf{x})}, y_{\pi_k(\mathbf{x})})). \tag{19.1}$$

Легко проверить, что предсказание по большинству меток (в случае классификации) и по усреднению меток (в случае регрессии) являются частными случаями (19.1) при соответствующем выборе φ. Но благодаря общности формулировки возможны и другие правила; например, если $\mathcal{Y} = \mathbb{R}$, то можно взять взвешенное среднее меток точек, выбираемых в соответствии с расстоянием до $\mathbf{x}$:

$$h_S(\mathbf{x}) = \sum_{i=1}^{k} \frac{\rho(\mathbf{x}, \mathbf{x}_{\pi_i(\mathbf{x})})}{\sum_{j=1}^{k} \rho(\mathbf{x}, \mathbf{x}_{\pi_j(\mathbf{x})})} y_{\pi_i(\mathbf{x})}.$$

19.2. Анализ

Поскольку правила ближайших соседей (NN) – настолько естественные методы обучения, их свойства обобщаемости были изучены очень подробно. Но большинство предшествующих результатов касаются асимптотической согласованности; в них анализируется качество правила NN, когда размер выборки m

стремится к бесконечности, а скорость сходимости зависит от истинного распределения. Как было отмечено в разделе 7.4, анализ такого вида нельзя признать удовлетворительным. Нам хотелось бы обучить систему на конечной обучающей выборке и представить свойства обобщаемости в виде функции от размера выборки и четко сформулированных априорных предположений о распределении данных. Поэтому мы приведем анализ правила 1-NN на конечных выборках и продемонстрируем характер убывания ошибки в виде функции от m и зависимость ошибки от свойств распределения. Мы также объясним, как этот анализ можно обобщить на правило k-NN при любом k. В частности, анализ позволяет определить, сколько примеров необходимо для достижения истинной ошибки $2L_{\mathcal{D}}(h^*) + \epsilon$, где h^* – оптимальная байесовская гипотеза, в предположении, что правила пометки «хорошо себя ведут» (что это значит, будет определено ниже).

19.2.1. Граница обобщаемости для правила 1-NN

Проанализируем истинную ошибку правила 1-NN для бинарной классификации с бинарной потерей, т. е. $\mathcal{Y} = \{0, 1\}$ и $\ell(h, (\mathbf{x}, y)) = \mathbb{1}_{[h(\mathbf{x})\neq y]}$. В процессе анализа мы будем предполагать, что $\mathcal{X} = [0, 1]^d$, а ρ – евклидово расстояние.

Начнем с обозначений. Пусть $\mathcal{D}$ – распределение на $\mathcal{X} \times \mathcal{Y}$. Обозначим $\mathcal{D}_{\mathcal{X}}$ – индуцированное маргинальное распределение на $\mathcal{X}$ и пусть $\eta : \mathbb{R}^d \to \mathbb{R}$ – условная вероятность меток[1], т. е.

$$\eta(\mathbf{x}) = \mathbb{P}[y = 1|\mathbf{x}].$$

Напомним, что оптимальное байесовское правило (т. е. гипотеза, которая минимизирует $L_{\mathcal{D}}(h)$ на множестве всех функций) имеет вид

$$h^*(\mathbf{x}) = \mathbb{1}_{[\eta(\mathbf{x})>1/2]}.$$

Мы предполагаем, что функция условной вероятности η является c-липшицевой для некоторого $c > 0$, т. е. для любых $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ $|\eta(\mathbf{x}) - \eta(\mathbf{x}')| \leq c\, \|\mathbf{x} - \mathbf{x}'\|$. Иными словами, это предположение означает, что если два вектора близки, то их метки, скорее всего, будут одинаковы.

В следующей лемме липшицевость функции условной вероятности применяется для получения верхней границы истинной ошибки правила 1-NN в виде функции от ожидаемого расстояния между тестовым образцом и его ближайшим соседом в обучающем наборе.

Лемма 19.1. *Пусть $\mathcal{X} = [0, 1]^d$, $\mathcal{Y} = \{0, 1\}$ и $\mathcal{D}$ – распределение на $\mathcal{X} \times \mathcal{Y}$, для которого функция условной вероятности η является c-липшицевой. Пусть $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ – независимая и одинаково распределенная выборка, а h_S – соответствующая ей гипотеза 1-NN. Обозначим h^* – оптимальное байесовское правило для η. Тогда*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \leq 2L_{\mathcal{D}}(h^*) + c \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m,\, \mathbf{x}\sim\mathcal{D}}[\|\mathbf{x} - \mathbf{x}_{\pi_1(\mathbf{x})}\|].$$

[1] Формально $\mathbb{P}[y = 1 \,|\, \mathbf{x}] = \lim_{\delta\to 0} \dfrac{\mathcal{D}(\{(\mathbf{x}', 1) : \mathbf{x}' \in B(\mathbf{x}, \delta)\})}{\mathcal{D}(\{(\mathbf{x}', y) : \mathbf{x}' \in B(\mathbf{x}, \delta),\, y \in \mathcal{Y}\})}$, где $B(x, \delta)$ — шар радиуса δ с центром в точке x.

Доказательство. Так как $L_{\mathcal{D}}(h_S) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\mathbb{1}_{[h_S(\mathbf{x})\neq y]}]$, то $\mathbb{E}_S[L_{\mathcal{D}}(h_S)]$ – вероятность выбрать обучающий набор S и дополнительный пример $(\mathbf{x}, y)$, так что метка $\pi_1(\mathbf{x})$ отлична от y. Иными словами, мы можем сначала выбрать m непомеченных примеров $S_x = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$ из распределения $\mathcal{D}_X$ и дополнительный непомеченный пример $\mathbf{x} \sim \mathcal{D}_X$, затем найти $\pi_1(\mathbf{x})$ – ближайшего соседа $\mathbf{x}$ в S_x – и, наконец, выбрать $y \sim \eta(\mathbf{x})$ и $y_{\pi 1}(\mathbf{x}) \sim \eta(\pi_1(\mathbf{x}))$. Отсюда следует, что

$$\begin{aligned}\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(h_S)] &= \mathop{\mathbb{E}}_{S_x\sim\mathcal{D}_X^m,\mathbf{x}\sim\mathcal{D}_X,y\sim\eta(\mathbf{x}),y'\sim\eta(\pi_1(\mathbf{x}))}[\mathbb{1}_{[y\neq y']}] \\ &= \mathop{\mathbb{E}}_{S_x\sim\mathcal{D}_X^m,\mathbf{x}\sim\mathcal{D}_X}\left[\mathop{\mathbb{P}}_{y\sim\eta(\mathbf{x}),y'\sim\eta(\pi_1(\mathbf{x}))}[y\neq y']\right]. \qquad (19.2)\end{aligned}$$

Далее мы ограничим сверху величину $\mathbb{P}_{y\sim\eta(\mathbf{x}),y'\sim\eta(\mathbf{x}')}[y \neq y']$ для любых двух точек $\mathbf{x}$, $\mathbf{x}'$ из области определения:

$$\begin{aligned}\mathop{\mathbb{P}}_{y\sim\eta(\mathbf{x}),y'\sim\eta(\mathbf{x}')}[y\neq y'] &= \eta(\mathbf{x}')(1-\eta(\mathbf{x})) + (1-\eta(\mathbf{x}'))\eta(\mathbf{x}) \\ &= (\eta(\mathbf{x}) - \eta(\mathbf{x}) + \eta(\mathbf{x}'))(1-\eta(\mathbf{x})) \\ &\quad + (1-\eta(\mathbf{x}) + \eta(\mathbf{x}) - \eta(\mathbf{x}'))\eta(\mathbf{x}) \\ &= 2\eta(\mathbf{x})(1-\eta(\mathbf{x})) + (\eta(\mathbf{x}) - \eta(\mathbf{x}'))(2\eta(\mathbf{x}) - 1).\end{aligned}$$

Используя неравенство $|2\eta(\mathbf{x}) - 1| \leq 1$ и предположение о c-липшицевости η, получаем, что эта вероятность не превосходит

$$\mathop{\mathbb{P}}_{y\sim\eta(\mathbf{x}),y'\sim\eta(\mathbf{x}')}[y\neq y'] \leq 2\eta(\mathbf{x})(1-\eta(\mathbf{x})) + c||\mathbf{x}-\mathbf{x}'||.$$

Подставляя в (19.2), заключаем, что

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(h_S)] \leq \mathop{\mathbb{E}}_{\mathbf{x}}[2\eta(\mathbf{x})(1-\eta(\mathbf{x}))] + c\mathop{\mathbb{E}}_{S,\mathbf{x}}[||\mathbf{x}-\mathbf{x}_{\pi_1(\mathbf{x})}||].$$

Наконец, ошибка оптимального байесовского классификатора

$$L_{\mathcal{D}}(h^*) = \mathop{\mathbb{E}}_{\mathbf{x}}[\min\{\eta(\mathbf{x}), 1-\eta(\mathbf{x})\}] \geq \mathop{\mathbb{E}}_{\mathbf{x}}[\eta(\mathbf{x})(1-\eta(\mathbf{x}))].$$

Объединяя последние два неравенства, мы завершаем доказательство леммы. □

Следующий шаг – получить оценку сверху математического ожидания расстояния между случайной точкой $\mathbf{x}$ и ближайшим к ней элементом S. Сначала нам потребуется следующая общая лемма из теории вероятностей. Она дает верхнюю границу массы вероятности подмножеств, в которые не попадает случайная выборка, в виде функции от размера выборки.

Лемма 19.2. *Пусть $C_1, \ldots, C_r$ – набор подмножеств некоторого множества $\mathcal{X}$. Пусть S – последовательность m точек, случайно и независимо выбранных из некоторого распределения $\mathcal{D}$ на $\mathcal{X}$. Тогда*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\sum_{i:C_i\cap S=\emptyset}\mathbb{P}[C_i]\right] \leq \frac{r}{me}.$$

Доказательство. В силу линейности математического ожидания

$$\mathop{\mathbb{E}}_{S}\left[\sum_{i:C_i\cap S=\emptyset}\mathbb{P}[C_i]\right] = \sum_{i=1}^{r}\mathbb{P}[C_i]\mathop{\mathbb{E}}_{S}[\mathbb{1}_{[C_i\cap S=\emptyset]}].$$

Далее для любого i имеем

$$\mathop{\mathbb{E}}_{S}[\mathbb{1}_{[C_i \cap S=\emptyset]}] = \mathop{\mathbb{P}}_{S}[C_i \cap S = \emptyset] = (1 - \mathop{\mathbb{P}}_{S}[C_i])^m \le e^{-\mathbb{P}[C_i]m}.$$

Объединяя два предыдущих выражения, получаем

$$\mathop{\mathbb{E}}_{S}\left[\sum_{i:C_i \cap S=\emptyset} \mathbb{P}[C_i]\right] \le \sum_{i=1}^{r} \mathbb{P}[C_i] e^{-\mathbb{P}[C_i]m} \le r \max_i \mathbb{P}[C_i] e^{-\mathbb{P}[C_i]m}.$$

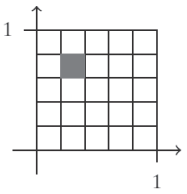
Наконец, из курса математического анализа известно, что $\max_a ae^{-ma} \le 1/me$, что и завершает доказательство. □

Вооружившись этими леммами, мы теперь готовы сформулировать и доказать главный результат этого раздела – верхнюю границу математического ожидания ошибки правила обучения 1-NN.

Теорема 19.3. *Пусть $\mathcal{X} = [0, 1]^d$, $\mathcal{Y} = \{0, 1\}$ и $\mathcal{D}$ – распределение на $\mathcal{X} \times \mathcal{Y}$, для которого функция условной вероятности η является c-липшицевой. Обозначим h_S результат применения правила 1-NN к выборке $S \sim \mathcal{D}^m$. Тогда*

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \le 2L_{\mathcal{D}}(h^*) + 4c\sqrt{d}\, m^{-\frac{1}{d+1}}.$$

Доказательство. Зафиксируем $\epsilon = 1/T$ для некоторого целого T, и пусть $r = T^d$, а $C_1, \ldots, C_r$ – покрытие множества $\mathcal{X}$ прямоугольниками длины ϵ, т. е. для любых $(\alpha_1, \ldots, \alpha_d) \in [T]^d$ существует множество C_i вида $\{\mathbf{x} : \forall j,\, x_j \in [(\alpha_j - 1)/T,\, \alpha_j/T]\}$. На рисунке ниже приведена иллюстрация для $d = 2$, $T = 5$ и множества, соответствующего $\alpha = (2,4)$.



Для любых $\mathbf{x}$, $\mathbf{x}'$, принадлежащих одному и тому же прямоугольнику, имеем $||\mathbf{x} - \mathbf{x}'|| \le \sqrt{d}\epsilon$. В противном случае $||\mathbf{x} - \mathbf{x}'|| \le \sqrt{d}$. Поэтому

$$\mathop{\mathbb{E}}_{\mathbf{x},S}[\|\mathbf{x} - \mathbf{x}_{\pi_i(\mathbf{x})}\|] \le \mathop{\mathbb{E}}_{S}\left[\mathbb{P}\left[\bigcup_{i:C_i \cap S=\emptyset} C_i\right]\sqrt{d} + \mathbb{P}\left[\bigcup_{i:C_i \cap S=\emptyset} C_i\right]\epsilon\sqrt{d}\right],$$

и, объединяя лемму 19.2 с тривиальным неравенством $\mathbb{P}[\bigcup_{i:C_i \cap S \neq \emptyset} C_i] \le 1$, получаем, что

$$\mathop{\mathbb{E}}_{\mathbf{x},S}[\|\mathbf{x} - \mathbf{x}_{\pi_1(\mathbf{x})}\|] \le \sqrt{d}\left(\frac{r}{me} + \epsilon\right).$$

Поскольку количество прямоугольников $r = (1/\epsilon)^d$, то получаем

$$\mathop{\mathbb{E}}_{\mathbf{x},S}[\|\mathbf{x}-\mathbf{x}_{\pi_1(\mathbf{x})}\|] \le \sqrt{d}\left(\frac{2^d\epsilon^{-d}}{me}+\epsilon\right).$$

Объединяем это с леммой 19.1:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(h_S)] \le 2L_{\mathcal{D}}(h^*)+c\sqrt{d}\left(\frac{2^d\epsilon^{-d}}{me}+\epsilon\right).$$

Наконец, полагая $\epsilon = 2m^{-1/(d+1)}$ и замечая, что

$$\begin{aligned}\frac{2^d\epsilon^{-d}}{me}+\epsilon &= \frac{2^d 2^{-d} m^{d/(d+1)}}{me}+2m^{-1/(d+1)}\\ &= m^{-1/(d+1)}(1/e+2) \le 4m^{-1/(d+1)},\end{aligned}$$

мы завершаем доказательство теоремы. □

Из этой теоремы следует, что если сначала зафиксировать распределение, порождающее данные, а затем устремить m к бесконечности, то ошибка правила 1-NN сходится к удвоенной байесовской ошибке. Проведенный анализ можно обобщить на большие значения k и показать, что математическое ожидание ошибки правила k-NN сходится к величине, которая в $(1 + \sqrt{8/k})$ раз больше ошибки байесовского классификатора. Это формализовано в теореме 19.5, доказательство которой мы оставляем в качестве упражнения (сопровождаемого указаниями).

19.2.2. Проклятие размерности

Верхняя граница из теоремы 19.3 растет вместе с c (липшицевым коэффициентом для η) и d, евклидовой размерностью области определения $\mathcal{X}$. На самом деле легко видеть, что необходимым условием для того, чтобы последний член в теореме 19.3 был меньше ϵ, является соблюдение неравенства $m \ge (4c\sqrt{d}/\epsilon)^{d+1}$. То есть размер обучающего набора должен расти экспоненциально вместе с увеличением размерности. Следующая теорема показывает, что это не просто частное свойство нашей верхней границы, но для некоторых распределений описывает количество примеров, действительно необходимых для обучения по правилу NN.

Теорема 19.4. *Для любого $c > 1$ и любого правила обучения L существует распределение на $[0, 1]^d \times \{0, 1\}$ такое, что $\eta(\mathbf{x})$ является c-липшицевой функцией, байесовская ошибка распределения равна 0, но для выборок размера $m \le (c + 1)^d/2$ истинная ошибка правила L больше 1/4.*

Доказательство. Зафиксируем произвольные значения c и d. Пусть G_c^d – решетка на $[0, 1]^d$ с расстоянием между узлами $1/c$. Тогда каждый узел решетки имеет вид $(a_1/c, \ldots, a_d/c)$, где a_i принадлежит множеству $\{0, \ldots, c - 1, c\}$. Заметим, что, поскольку любые два узла решетки находятся на расстоянии не меньше $1/c$ друг от друга, то любая функция $\eta : G_C^D \to [0, 1]$ будет c-липшицевой. Отсюда следует, что множество всех c-липшицевых функций на G_c^d содержит множество *всех* бинарных функций в этой области определения. Поэтому мы можем сослаться на

теорему об отсутствии бесплатных завтраков (теорема 5.1) и получить нижнюю границу размера выборки, необходимой для обучения этого класса. Количество узлов решетки равно $(c+1)^d$, поэтому если $m < (c+1)^d/2$, то из теоремы 5.1 вытекает доказываемая нижняя граница. □

Экспоненциальная зависимость от размерности известна под названием *проклятия размерности*. Как мы видели, правило 1-NN может потерпеть неудачу, если количество примеров меньше $\Omega((c+1)^d)$. Поэтому, хотя правило 1-NN не ограничивается предопределенным набором гипотез, оно все же опирается на некое априорное знание – его успешность зависит от предположения о том, что размерность и липшицева константа порождающего распределения η не слишком велики.

19.3. Эффективная реализация*

Метод ближайших соседей – это правило типа «обучение путем запоминания». Для его применения необходимо хранить весь обучающий набор, а во время проверки просматривать его для поиска соседей. Поэтому время применения правила NN составляет $\Theta(dm)$, а значит, вычисления на этапе проверки обходятся дорого.

При небольших d можно воспользоваться структурами данных из области вычислительной геометрии, которые позволяют снизить время до $o(d^{O(1)}\log(m))$. Однако эти структуры потребляют память размера $m^{O(d)}$, что делает их практически бесполезными при больших d.

Для преодоления этой трудности было предложено заменить точный поиск *приближенным*. Формально, r-аппроксимирующая процедура поиска гарантированно находит точку, отстоящую на расстояние, которое превышает расстояние до ближайшего соседа не более чем в r раз. Три наиболее популярных алгоритма поиска квазиближайшего соседа – kd-деревья, шаровые деревья (balltree) и локально-чувствительное хеширование (locality-sensitive hashing – LSH). Отсылаем читателя, например, к работе Shakhnarovich, Darrell & Indyk (2006).

19.4. Резюме

Правило k-NN – очень простой алгоритм обучения, опирающийся на предположение, что «предметы, которые выглядят, похоже, должны быть действительно похожи». Мы формализовали это интуитивное представление с помощью свойства липшицевости функции условной вероятности. Мы показали, что при достаточно большом обучающем наборе риск правила 1-NN ограничен сверху удвоенным риском оптимального байесовского правила. Мы также вывели нижнюю границу, демонстрирующую «проклятие размерности» – требуемый размер выборки экспоненциально растет при увеличении размерности. Поэтому на практике метод ближайших соседей обычно используется после шага предобработки, на котором понижается размерность. Методы понижения размерности мы будем обсуждать в главе 23.

19.5. Библиографические сведения

В работе Cover and Hart (1967) впервые проведен анализ алгоритма 1-NN и показано, что его риск сходится к удвоенной оптимальной байесовской ошибке при довольно мягких условиях. Пользуясь леммой из работы Stone (1977), Devroye and Györfi (1985) показали, что правило k-NN согласовано (относительно класса гипотез, состоящего из всех функций из $\mathbb{R}^d$ в {0, 1}). Хорошее изложение этого анализа имеется в книге Devroye et al. (1996). Здесь мы привели гарантию для конечной выборки, в которой явно используется априорное предположение о распределении. О результатах, относящихся к согласованности, см. раздел 7.4. Наконец, в работе Gottlieb, Kontorovich, and Krauthgamer (2010) доказана еще одна граница конечной выборки для NN, больше напоминающая границы VC-размерности.

19.6. Упражнения

В этом упражнении мы докажем следующую теорему для правила **k-NN**.

Теорема 19.5. *Пусть $\mathcal{X} = [0, 1]^d$, $\mathcal{Y} = \{0, 1\}$, а $\mathcal{D}$ – распределение на $\mathcal{X} \times \mathcal{Y}$, для которого функция условной вероятности η является c-липшицевой. Обозначим h_S результат применения правила k-NN к выборке $S \sim \mathcal{D}^m$, где $k \geq 10$. Пусть h^* – оптимальная байесовская гипотеза. Тогда*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(h_S)] \leq \left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^*) + \left(6c\sqrt{d} + k\right) m^{-1/(d+1)}.$$

19.1. Докажите следующую лемму

Лемма 19.6. *Пусть $C_1, \ldots, C_r$ – коллекция подмножеств некоторого множества $\mathcal{X}$. Пусть S – последовательность m точек, независимо выбранных из некоторого распределения вероятностей $\mathcal{D}$ на $\mathcal{X}$. Тогда для любого $k \geq 2$*

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m}\left[\sum_{i:|C_i \cap S|<k} \mathbb{P}[C_i]\right] \leq \frac{2rk}{m}.$$

Указания:

- Покажите, что

$$\mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S|<k} \mathbb{P}[C_i]\right] = \sum_{i=1}^{r} \mathbb{P}[C_i]\, \mathop{\mathbb{P}}_{S}[|C_i \cap S| < k].$$

- Зафиксируем некоторое i и предположим, что $k < \mathbb{P}[C_i]m/2$. Воспользовавшись границей Чернова, покажите, что

$$\mathop{\mathbb{P}}_{S}[|C_i \cap S| < k] \leq \mathop{\mathbb{P}}_{S}[|C_i \cap S| < \mathbb{P}[C_i]m/2] \leq e^{-\mathbb{P}[C_i]m/8}.$$

- Воспользовавшись неравенством $\max_a ae^{-ma} \leq 1/(me)$, покажите, что для такого i имеем

$$\mathbb{P}[C_i]\mathbb{P}_S[|C_i \cap S| < k] \le \mathbb{P}[C_i]e^{-\mathbb{P}[C_i]m/8} \le \frac{8}{me}.$$

- Завершите доказательство, воспользовавшись тем фактом, что в случае $k \ge \mathbb{P}[C_i]m/2$, очевидно, справедливы неравенства

$$\mathbb{P}[C_i]\mathbb{P}_S[|C_i \cap S| < k] \le \mathbb{P}[C_i] \le \frac{2k}{m}.$$

19.2. Будем использовать обозначение $y \sim p$ как сокращение фразы «y – случайная величина, имеющая распределение Бернулли с математическим ожиданием p». Докажите следующую лемму.

Лемма 19.7. *Пусть $k \ge 10$ и $Z_1, \ldots, Z_k$ – независимые случайные величины с распределением Бернулли, для которых $\mathbb{P}[Z_i = 1] = p_i$. Обозначим $p = (1/k)\sum_i p_i$ и $p' = (1/k)\sum_{i=1}^k Z_i$. Покажите, что*

$$\mathop{\mathbb{E}}_{Z_1,\ldots,Z_k} \mathop{\mathbb{P}}_{y\sim p}[y \ne \mathbb{1}_{[p'>1/2]}] \le \left(1 + \sqrt{\frac{8}{k}}\right) \mathop{\mathbb{P}}_{y\sim p}[y \ne \mathbb{1}_{[p>1/2]}].$$

Указания:

Без ограничения общности можно предположить, что $p \le 1/2$. Тогда $\mathbb{P}_{y\sim p}[y \ne \mathbb{1}_{[p>1/2]}] = p$. Положим $y' = \mathbb{1}_{[p'>1/2]}$.

- Покажите, что

$$\mathop{\mathbb{E}}_{Z_1,\ldots,Z_k} \mathop{\mathbb{P}}_{y\sim p}[y \ne y'] - p = \mathop{\mathbb{P}}_{Z_1,\ldots,Z_k}[p' > 1/2](1-2p).$$

- Воспользовавшись границей Чернова (лемма B.3), покажите, что

$$\mathbb{P}[p' > 1/2] \le e^{-kph\left(\frac{1}{2p}-1\right)},$$

где

$$h(a) = (1 + a)\log(1 + a) - a.$$

- Чтобы завершить доказательство леммы, можете воспользоваться следующим неравенством (не доказывая его): для любого $p \in [0,1/2]$ и $k \ge 10$:

$$(1-2p)e^{-kp+\frac{k}{2}(\log(2p)+1)} \le \sqrt{\frac{8}{k}}p.$$

19.3. Зафиксируем некоторые $p, p' \in [0, 1]$ и $y' \in \{0, 1\}$. Покажите, что

$$\mathop{\mathbb{P}}_{y\sim p}[y \ne y'] \le \mathop{\mathbb{P}}_{y\sim p}[y \ne y'] + |p - p'|.$$

19.4. Завершите доказательство теоремы следующими шагами.

- Как и при доказательстве теоремы 19.3, зафиксируем некоторое $\epsilon > 0$ и пусть $C_1, \ldots, C_r$ – покрытие множества $\mathcal{X}$ прямоугольниками длины ϵ. Для

любых $\mathbf{x}$, $\mathbf{x}'$, принадлежащих одному и тому же прямоугольнику имеем $\|\mathbf{x} - \mathbf{x}'\| \le \sqrt{d}\epsilon$. В остальных случаях $\|\mathbf{x} - \mathbf{x}'\| \le 2\sqrt{d}$. Покажите, что

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(h_S)] \le \mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S|<k} \mathbb{P}[C_i]\right] + \max_i \mathop{\mathbb{P}}_{S,(\mathbf{x},y)}\left[h_S(\mathbf{x}) \neq y \mid \forall j \in [k], \|\mathbf{x} - \mathbf{x}_{\pi_j(\mathbf{x})}\| \le \epsilon\sqrt{d}\right]. \tag{19.3}$$

- Ограничьте сверху первое слагаемое, применив лемму 19.6.
- Чтобы ограничить сверху второе слагаемое, зафиксируем $S|_x$ и $\mathbf{x}$ так, чтобы все k соседей $\mathbf{x}$ в $S|_x$ находились от $\mathbf{x}$ на расстоянии не более $\sqrt{d}$. Без ограничения общности предположим, что k ближайших соседей – это $\mathbf{x}_1, \ldots, \mathbf{x}_k$. Обозначим $p_i = \eta(\mathbf{x}_i)$ и пусть $p = (1/k)\sum_i p_i$. Воспользуйтесь упражнением 19.3, чтобы показать, что

$$\mathop{\mathbb{E}}_{y_1,\ldots,y_j} \mathop{\mathbb{P}}_{y\sim\eta(\mathbf{x})}[h_S(\mathbf{x}) \neq y] \le \mathop{\mathbb{E}}_{y_1,\ldots,y_j} \mathop{\mathbb{P}}_{y\sim p}[h_S(\mathbf{x}) \neq y] + |p - \eta(\mathbf{x})|.$$

Без ограничения общности предположим, что $p \le 1/2$. Теперь с помощью леммы 19.7 покажите, что

$$\mathop{\mathbb{P}}_{y_1,\ldots,y_j} \mathop{\mathbb{P}}_{y\sim p}[h_S(\mathbf{x}) \neq y] \le \left(1 + \sqrt{\frac{8}{k}}\right) \mathop{\mathbb{P}}_{y\sim p}[\mathbb{1}_{[p>1/2]} \neq y].$$

- Покажите, что

$$\mathop{\mathbb{P}}_{y\sim p}[\mathbb{1}_{[p>1/2]} \neq y] = p = \min\{p, 1-p\} \le \min\{\eta(\mathbf{x}), 1-\eta(\mathbf{x})\} + |p - \eta(\mathbf{x})|.$$

- Объединив все полученные результаты, покажите, что второе слагаемое в выражении (19.3) ограничено сверху величиной

$$\left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^*) + 3c\epsilon\sqrt{d}.$$

- Взяв $r = (2/\epsilon)^d$, получаем:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(h_S)] \le \left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^*) + 3c\epsilon\sqrt{d} + \frac{2(2/\epsilon)^d k}{m}.$$

Положив $\epsilon = 2m^{-1/(d+1)}$ и, воспользовавшись неравенством

$$6cm^{-1/(d+1)}\sqrt{d} + \frac{2k}{e}m^{-1/(d+1)} \le \left(6c\sqrt{d} + k\right)m^{-1/(d+1)},$$

завершите доказательство.

Глава 20

НЕЙРОННЫЕ СЕТИ

Искусственная нейронная сеть – это модель вычислений, организованная по образцу структуры нейронных сетей в мозге. Упрощенные модели мозга представляют его как собрание большого числа вычислительных устройств (нейронов), соединенных друг с другом и образующих сложную вычислительную сеть, благодаря которой мозг способен выполнять сложнейшие вычисления. Искусственные нейронные сети – это формальные конструкции, моделирующие такую парадигму вычислений.

Обучение нейронных сетей было предложено в середине двадцатого столетия. Это эффективная парадигма, а недавно была продемонстрирована ее высочайшая производительность и качество на нескольких задачах обучения.

Нейронную сеть можно описать как ориентированный граф, вершины которого соответствуют нейронам, а ребра – связям между ними. Каждый нейрон получает на выходе взвешенную сумму выходов нейронов, соединенных с его входящими ребрами. Нас будут интересовать сети *прямого распространения*, граф которых не содержит циклов.

В контексте обучения мы можем определить класс гипотез, состоящих из нейросетевых предикторов, в котором все гипотезы разделяют общую графовую структуру сети, а различаются весами ребер. В разделе 20.3 мы увидим, что любой предиктор с n переменными, который можно реализовать за время $T(n)$, можно также выразить в виде нейросетевого предиктора размера $O(T(n)^2)$, где под размером сети понимается количество вершин в ней. Отсюда следует, что семейства классов гипотез, имеющих вид нейронной сети полиномиального размера, может оказаться достаточно для всех практических задач обучения, цель которых – обучить предикторы, допускающие эффективную реализацию. Далее, в разделе 20.4, мы покажем, что выборочная сложность обучения таких классов гипотез также ограничена в терминах размера сети. Поэтому складывается впечатление, что это окончательная парадигма обучения в том смысле, что и выборочная сложность растет полиномально, и ошибка аппроксимации минимальна среди всех классов гипотез, состоящих из эффективно реализуемых предикторов. И, стало быть, больше нечего желать.

Подвох, однако, в том, что задача обучения классов гипотез, состоящих из нейросетевых предикторов, вычислительно трудна. Мы формализуем эту мысль в разделе 20.5. Широко распространенная эвристика для обучения нейронных сетей опирается на метод СГС, который мы изучали в главе 14. Там было показано, что СГС является успешным обучаемым, если функция потерь выпукла.

Но в нейронных сетях функция потерь очень далека от выпуклости. Тем не менее мы можем реализовать алгоритм СГС и надеяться, что он найдет разумное решение (как то имеет место в нескольких практически важных задачах). В разделе 20.6 мы опишем, как реализовать СГС для нейронных сетей. В частности, самой сложной операцией оказывается вычисление градиента функции потерь по параметрам сети. Мы представим алгоритм *обратного распространения*, который эффективно вычисляет градиент.

20.1. Нейронные сети прямого распространения

Идея нейронной сети состоит в том, что много нейронов можно соединить коммуникационными связями для выполнения сложных вычислений. Принято описывать структуру нейронной сети в виде графа, вершинами (узлами) которого являются нейроны, а каждое (ориентированное) ребро связывает выход одного нейрона с входом другого. Мы ограничимся сетями прямого распространения, для которых соответствующий граф не содержит циклов.

Нейронная сеть прямого распространения описывается ориентированным ациклическим графом $G = (V, E)$ и весовой функций на ребрах $w : E \to \mathbb{R}$. Вершины графа соответствуют нейронам. Каждый нейрон моделируется как простая скалярная функция $\sigma : \mathbb{R} \to \mathbb{R}$. Мы ограничимся рассмотрением трех возможных функций σ: знаковая функция $\sigma(a) = \text{sign}(a)$, ступенчатая функция $\sigma(a) = \mathbb{1}_{[a>0]}$ и сигмоида $\sigma(a) = 1/(1 + \exp(-a))$, являющаяся гладкой аппроксимацией ступенчатой функции. Функция σ называется функцией «активации» нейрона. Каждое ребро графа связывает выход одного нейрона с входом другого. Суммарный вход нейрона равен взвешенной сумме выходов соединенных с ним нейронов, а схема взвешивания определяется функцией w.

Чтобы упростить описание вычислений, выполняемых сетью, мы будем далее предполагать, что сеть организована в виде *слоев*, т. е. множество вершин можно представить в виде объединения непустых непересекающихся подмножеств, $V = \biguplus_{t=0}^{T} V_t$, так что каждое ребро из E соединяет вершину из V_{t-1} с вершиной из V_t для некоторого $t \in [T]$. Нижний слой V_0 называется входным. Он содержит $n + 1$ нейронов, где n – размерность пространства входов. Для любого $i \in [n]$ выход нейрона i слоя V_0 – это просто x_i. Последний нейрон слоя V_0 – «постоянный», он всегда выводит 1. Будем обозначать $v_{t,i}$ i-й нейрон t-го слоя, а $o_{t,i}(\mathbf{x})$ – выход $v_{t,i}$, когда на вход сети подается вектор $\mathbf{x}$. Таким образом, для $i \in [n]$ имеем $o_{0,i}(\mathbf{x}) = x_i$, а для $i = n + 1$ имеем $o_{0,i}(\mathbf{x}) = 1$. Теперь перейдем к послойным вычислениям. Предположим, что уже вычислены выходы нейронов слоя t. Тогда выходы нейронов слоя $t + 1$ можно вычислить следующим образом. Зафиксируем некоторый нейрон $v_{t+1,j} \in V_{t+1}$. Обозначим $a_{t+1,j}(\mathbf{x})$ вход $v_{t+1,j}$, когда на вход сети подан вектор $\mathbf{x}$. Тогда

$$a_{t+1,j}(\mathbf{x}) = \sum_{r:(v_{t,r},\, v_{t+1,j}) \in E} w((v_{t,r}, v_{t+1,j}))\, o_{t,r}(\mathbf{x})$$

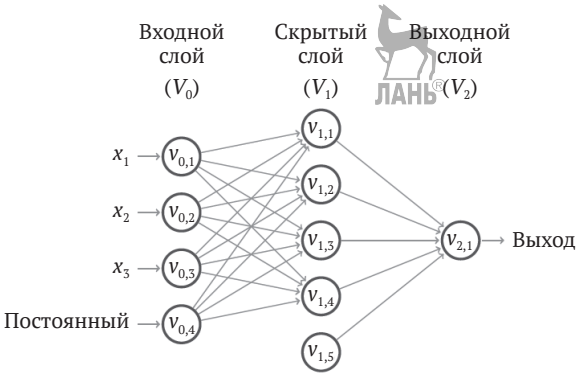
и

$$o_{t+1,j}(\mathbf{x}) = \sigma(a_{t+1,j}(\mathbf{x})).$$

Это означает, что входом $v_{t+1,j}$ является взвешенная сумма нейронов слоя V_t, соединенных с $v_{t+1,j}$, в которой веса определяются функцией w, а выходом $v_{t+1,j}$ является просто результат применения функции активации σ к его входу.

Слои V_1, ..., V_{T-1} часто называют *скрытыми*. Верхний слой V_T называется выходным. В простых проблемах предсказания выходной слой содержит один нейрон, и его выход является выходом всей сети.

Мы называем T числом слоев сети (V_0 исключается) или ее глубиной. Размер сети обозначается $|V|$. Шириной сети называется величина $\max_t |V_t|$. На рисунке ниже приведена иллюстрация многослойной нейронной сети прямого распространения глубины 2, размера 10 и ширины 5. Заметим, что у одного нейрона в скрытом слое нет входящих ребер. Этот нейрон выводит константу $\sigma(0)$.



20.2. Обучение нейронных сетей

Определив нейронную сеть четверкой (V, E, σ, w), мы получаем функцию $h_{V,E,\sigma,w} : \mathbb{R}^{|V_0|-1} \to \mathbb{R}^{|V_T|}$. Любое множество таких функций может выступать в роли класса гипотез для обучения. Обычно мы определяем класс гипотез нейросетевых предикторов, зафиксировав граф (V, E) и функцию активации σи включая в класс все функции вида $h_{V,E,\sigma,w} : E \to \mathbb{R}$. Тройку (V, E, σ) часто называют *архитектурой* сети. Мы обозначаем класс гипотез

$$\mathcal{H}_{V,E,\sigma} = \{h_{V,E,\sigma,w} : w - \text{отображение } E \text{ в } \mathbb{R}\}. \tag{20.1}$$

Таким образом, параметры, описывающие гипотезу, принадлежащую классу гипотез, – это веса ребер сети.

Теперь мы можем изучать ошибку аппроксимации, ошибку оценивания и ошибку оптимизации таких классов гипотез. В разделе 20.3 мы изучим ошибку аппроксимации $\mathcal{H}_{V,E,\sigma}$, исследовав, какие типы функций могут реализовать гипотезы из $\mathcal{H}_{V,E,\sigma}$ в терминах графа сети. В разделе 20.4 мы изучим ошибку оценивания $\mathcal{H}_{V,E,\sigma}$ для случая бинарной классификации (т. е. $V_T = 1$, а σ – функция sign), для чего проанализируем VC-размерность класса. Наконец, в разделе 20.5 мы покажем, что обучение класса $\mathcal{H}_{V,E,\sigma}$ – вычислительно трудная задача, даже если граф сети мал, а в разделе 20.6 представим наиболее распространенную эвристику для обучения $\mathcal{H}_{V,E,\sigma}$.

20.3. Выразительная способность нейронных сетей

В этом разделе мы изучим выразительную способность нейронных сетей, т. е. выясним, какие типы функций можно реализовать с помощью нейронной сети. Точнее, мы зафиксируем некоторую архитектуру V, E, σ и попробуем описать, какие функции могут быть реализованы гипотезами из $\mathcal{H}_{V,E,\sigma}$, в виде функции от размера V.

Мы начнем обсуждение с изучения того, какие типы булевых функций (т. е. функций из $\{\pm 1\}^n$ в $\{\pm 1\}$) можно реализовать с помощью $\mathcal{H}_{V,E,\text{sign}}$. Заметим, что для любого компьютера, в котором вещественные числа хранятся в виде b бит, вычисление функции $f : \mathbb{R}^n \to \mathbb{R}$ на самом деле подменяется вычислением функции $g : \{\pm 1\}^{nb} \to \{\pm 1\}^b$. Поэтому, изучив, какие булевы функции допускают реализацию с помощью $\mathcal{H}_{V,E,\text{sign}}$, мы сможем что-то сказать о том, какие функции могут быть реализованы на компьютере, который представляет вещественные числа b битами.

Для начала докажем простое утверждение, которое показывает, что если не ограничивать размер сети, то любую булеву функцию можно реализовать нейронной сетью глубины 2.

Утверждение 20.1. *Для любого n существует граф (V, E) глубины 2 такой, что $\mathcal{H}_{V,E,\text{sign}}$ содержит все функции из $\{\pm 1\}^n$ в $\{\pm 1\}$.*

Доказательство. Мы построим граф, для которого $|V_0| = n + 1$, $|V_1| = 2^n + 1$ и $|V_2| = 1$. В качестве E возьмем множество всех возможных ребер между соседними слоями. Пусть теперь $f : \{\pm 1\}^n \to \{\pm 1\}$ – некоторая булева функция. Мы должны показать, что можем подобрать веса так, что сеть будет реализовывать f. Пусть $\mathbf{u}_1, \ldots, \mathbf{u}_k$ – все векторы в $\{\pm 1\}^n$, для которых f принимает значение 1. Заметим, что для любого i и любого $\mathbf{x} \in \{\pm 1\}^n$, если $\mathbf{x} \neq \mathbf{u}_i$, то $\langle \mathbf{x}, \mathbf{u}_i \rangle \leq n - 2$, а если $\mathbf{x} = \mathbf{u}_i$, то $\langle \mathbf{x}, \mathbf{u}_i \rangle = n$. Отсюда следует, что функция $g_i(\mathbf{x}) = \text{sign}(\langle \mathbf{x}, \mathbf{u}_i \rangle - n + 1)$ равна 1 тогда и только тогда, когда $\mathbf{x} = \mathbf{u}_i$. А это значит, что мы можем подобрать веса ребер между V_0 и V_1, так что для любого $i \in [k]$ нейрон $v_{1,i}$ будет реализовывать функцию $g_i(\mathbf{x})$. Далее заметим, что $f(\mathbf{x})$ является дизъюнкцией функций $g_i(\mathbf{x})$ и, следовательно, может быть записана в виде

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{k} g_i(\mathbf{x}) + k - 1\right),$$

что и требовалось доказать. □

Это утверждение показывает, что с помощью нейронных сетей можно реализовать любую булеву функцию. Однако это очень слабое свойство, потому что размер результирующей сети может расти экспоненциально. В конструкции, приведенной в доказательстве утверждения 20.1, количестве вершин в скрытом слое экспоненциально велико. И это не случайная особенность нашего доказательства, как вытекает из следующей теоремы.

Теорема 20.2. *Для любого n обозначим $s(n)$ такое минимальное целое число, что существует граф (V, E) с $|V| = s(n)$, для которого класс гипотез $\mathcal{H}_{V,E,\text{sign}}$ содержит все функции из $\{0, 1\}^n$ в $\{0, 1\}$. Тогда $s(n)$ растет экспоненциально с ростом n. Аналогичный результат имеет место для класса $\mathcal{H}_{V,E,\sigma}$, где σ – сигмоидная функция.*

Доказательство. Предположим, что для некоторого графа (V, E) класс $\mathcal{H}_{V,E,\text{sign}}$ содержит все функции из $\{0, 1\}^n$ в $\{0, 1\}$. Тогда он может разбить множество $m = 2^n$ векторов из $\{0, 1\}^n$ и, следовательно, VC-размерность $\mathcal{H}_{V,E,\text{sign}}$ равна 2^n. С другой стороны, VC-размерность $\mathcal{H}_{V,E,\text{sign}}$ ограничена сверху величиной порядка $O(|E| \log(|E|)) \le O(|V|^3)$, как мы покажем в следующем разделе. Отсюда следует, что $|V| \ge \Omega(2^{n/3})$, что и завершает доказательство для случая сетей с функцией активации sign. Доказательство для случая сигмоиды аналогично. □

Замечание 20.1. Можно доказать аналогичную теорему для класса $\mathcal{H}_{V,E,\sigma}$ с любой функцией σ, если только ограничить веса, так чтобы любой вес можно было выразить числом бит, ограниченным сверху универсальной константой. Мы даже можем рассмотреть классы гипотез, в которых разные нейроны выполняют разные функции активации при условии, что количество допустимых функций активации тоже конечно.

Какие функции можно выразить с помощью сети полиномиального размера? Доказанное выше утверждение говорит, что невозможно выразить все булевы функции сетью полиномиального размера. С другой стороны, ниже мы покажем, что все булевы функции, вычислимые за время $O(T(n))$, можно выразить сетью размера $O(T(n)^2)$.

Теорема 20.3. *Пусть $T : \mathrm{N} \to \mathrm{N}$, и для любого n пусть $\mathcal{F}_n$ – множество функций, которые можно реализовать машиной Тьюринга за время не более $T(n)$. Тогда существуют константы $b, c \in \mathbb{R}_+$ такие, что для любого n существует граф (V_n, E_n) размера не более $cT(n)^2 + b$, для которого $\mathcal{H}_{V_n,E_n,\text{sign}}$ содержит $\mathcal{F}_n$.*

Доказательство этой теоремы опирается на связь между временной и схемной сложностью программ (см., например, Sipser, 2006). В двух словах, булевой схемой называется такой тип сети, в котором отдельные нейроны реализуют конъюнкции, дизъюнкции и отрицания своих входов. Схемная сложность измеряет размер булевых схем, необходимых для вычисления функций. Связь между временной и схемной сложностью на интуитивном уровне можно описать следующим образом. Мы можем смоделировать каждый шаг выполнения компьютерной программы как простую операцию над ее памятью. Следовательно, нейроны каждого слоя сети отражают состояние памяти компьютера в соответствующий момент времени, а переход к следующему слою сопровождается простым вычислением, которое может выполнить сеть. Чтобы соотнести булевы схемы с сетями с функцией активации sign, мы должны показать, что можем реализовать операции конъюнкции, дизъюнкции и отрицания с помощью этой функции активации. Следующая лемма показывает, то функция активации sign может также реализовать конъюнкции и дизъюнкции своих входов.

Лемма 20.4. *Предположим, что нейрон v, реализующий функцию активации sign, имеет k входящих ребер, соединяющих его с нейронами, выдающими на выходе $\{\pm 1\}$. Тогда, добавив еще одно ребро, связывающее «постоянный» нейрон с v, и подобрав веса на ребрах, входящих в v, мы можем сделать так, что выход v будет реализовывать конъюнкцию или дизъюнкцию своих входов.*

Доказательство. Просто заметим, что если $f : \{\pm 1\}^k \to \{\pm 1\}$ – конъюнкция, т. е. $f(\mathbf{x}) = \wedge_i x_i$, то ее можно записать в виде $f(\mathbf{x}) = \text{sign}(1 - k + \sum_{i=1}^k x_i)$. Аналогично, дизъюнкцию $f(\mathbf{x}) = \vee_i x_i$ можно записать в виде $f(\mathbf{x}) = \text{sign}(k - 1 + \sum_{i=1}^k x_i)$. □

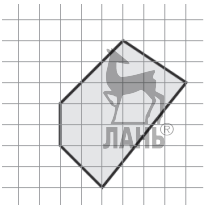
До сих пор мы обсуждали булевы функции. В упражнении 20.1 мы покажем, что нейронные сети являются *универсальными аппроксиматорами*. Это означает, что для любого фиксированного значения точности $\epsilon > 0$ и для любой липшицевой функции $f : [-1, 1]^n \to [-1, 1]$ можно построить такую сеть, которая для любого входа $\mathbf{x} \in [-1, 1]^n$ будет выводить число между $f(\mathbf{x}) - \epsilon$ и $f(\mathbf{x}) + \epsilon$. Однако, как и в случае булевых функций, размер сети растет быстрее, чем полиномиально, с ростом n. Формально это выражено в следующей теореме, которая является прямым следствием теоремы 20.2 и доказательство которой мы оставляем в качестве упражнения.

Теорема 20.5. *Зафиксируем некоторое $\epsilon \in (0, 1)$. Для любого n обозначим $s(n)$ такое минимальное целое число, что существует граф (V, E) с $|V| = s(n)$, для которого класс гипотез $\mathcal{H}_{V,E,\sigma}$, где σ – сигмоидная функция, может с любой точностью ϵ аппроксимировать произвольную 1-липшицеву функцию $f : [-1, 1]^n \to [-1, 1]$. Тогда $s(n)$ растет экспоненциально вместе с ростом n.*

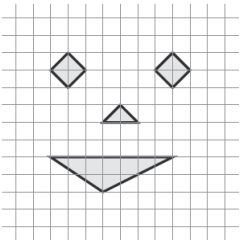
20.3.1. Геометрическая интерпретация

Далее мы приведем несколько геометрических иллюстраций функций $f : \mathbb{R}^2 \to \{\pm 1\}$ и покажем, как выразить их нейронной сетью с функцией активации sign.

Начнем с сети глубиной 2, т. е. с одним скрытым слоем. Каждый нейрон скрытого слоя реализует полупространственный предиктор. А единственный нейрон выходного слоя применяет полупространство к бинарным выходам нейронов скрытого слоя. Как мы уже видели, с помощью полупространств можно реализовать функцию конъюнкции. Поэтому такие сети содержат все гипотезы, являющиеся пересечением $k - 1$ полупространств, где k – число нейронов в скрытом слое, т. е. с их помощью модно выразить все выпуклые политопы с $k - 1$ гранями. На рисунке ниже показано пересечение 5 полупространств.



Мы показали, что нейрон слоя V_2 может реализовать функцию, которая сообщает, принадлежит ли $\mathbf{x}$ некоторому выпуклому политопу. Добавив еще один слой и позволив нейрону выходного слоя реализовывать дизъюнкцию своих входов, мы получим сеть, которая вычисляет объединение политопов. На рисунке ниже иллюстрируется такая функция.



20.4. Выборочная сложность нейронных сетей

Далее мы обсудим выборочную сложность обучения класса $\mathcal{H}_{V,E,\sigma}$. Напомним, что согласно фундаментальной теореме обучения выборочная сложность обучения класса бинарных классификаторов зависит от его VC-размерности. Следовательно, мы можем сконцентрироваться на VC-размерности класса гипотез вида $\mathcal{H}_{V,E,\sigma}$, в котором выходной слой графа содержит всего один нейрон.

Начнем с функции активации sign, т. е. с класса $\mathcal{H}_{V,E,\text{sign}}$. Какова VC-размерность этого класса? Интуитивно кажется, что поскольку мы обучаем $|E|$ параметров, то VC-размерность должна иметь порядок $|E|$. Так оно и есть, как показывает следующая теорема.

Теорема 20.6. *VC-размерность класса* $\mathcal{H}_{V,E,\text{sign}}$ *равна* $O(|E|\log(|E|))$.

Доказательство. Чтобы упростить нотацию, обозначим класс гипотез просто $\mathcal{H}$. Напомним определение функции роста $\tau_{\mathcal{H}}(m)$ из раздела 6.5.1. Эта функция измеряет $\max_{C\subset\mathcal{X}:|C|=m}|\mathcal{H}_C|$, где $\mathcal{H}_C$ – ограничение $\mathcal{H}$ на функции из C в $\{0, 1\}$. Мы можем естественно распространить это определение на множество функций из $\mathcal{X}$ в некоторое конечное множество $\mathcal{Y}$, считая $\mathcal{H}_C$ ограничением $\mathcal{H}$ на функции из C в $\mathcal{Y}$ и оставив определение $\tau_{\mathcal{H}}(m)$ без изменения.

Наша нейронная сеть определена слоистым графом. Пусть $V_0, \ldots, V_T$ – слои графа. Зафиксируем $t\in[T]$. Назначая разные веса ребрам, идущим из V_{t-1} в V_t, мы получаем различные функции из $\mathbb{R}^{|V_{t-1}|}\to\{\pm1\}^{|V_t|}$. Обозначим $\mathcal{H}^{(t)}$ класс всех таких отображений из $\mathbb{R}^{|V_{t-1}|}$ в $\{\pm1\}^{|V_t|}$. Тогда $\mathcal{H}$ можно записать в виде композиции $\mathcal{H}=\mathcal{H}^{(T)}\circ\ldots\circ\mathcal{H}^{(1)}$. В упражнении 20.4 мы покажем, что функция роста композиции классов гипотез ограничена сверху произведением функций роста отдельных классов, т. е.

$$\tau_{\mathcal{H}}(m)\le\prod_{t=1}^{T}\tau_{\mathcal{H}^{(t)}}(m).$$

Кроме того, каждый $\mathcal{H}^{(t)}$ можно записать в виде произведения классов функций $\mathcal{H}^{(t)}=\mathcal{H}^{(t,1)}\times\ldots\times\mathcal{H}^{(t,|V_t|)}$, где $\mathcal{H}^{(t,j)}$ – множество всех функций из слоя $t-1$ в $\{\pm1\}$, которые могут быть реализованы j-м нейроном слоя t. В упражнении 20.3 мы ограничим классы-сомножители, и это даст

$$\tau_{\mathcal{H}^{(t)}}(m)\le\prod_{i=1}^{|V_t|}\tau_{\mathcal{H}^{(t,i)}}(m).$$

Обозначим $d_{t,i}$ количество ребер, входящих в i-й нейрон слоя t. Поскольку нейрон – это однородная полупространственная гипотеза, а VC-размерность класса однородных полупространств равна размерности их входа, имеем, согласно лемме Зауэра,

$$\tau_{\mathcal{H}^{(t,i)}}(m)\le\left(\frac{em}{d_{t,i}}\right)^{d_{t,i}}\le(em)^{d_{t,i}}.$$

В итоге мы получили, что

$$\tau_{\mathcal{H}}(m)\le(em)^{\sum_{t,i}d_{t,i}}=(em)^{|E|}.$$

Теперь предположим, что существует m разбитых точек. Тогда должно быть $\tau_{\mathcal{H}}(m) = 2^m$, откуда получаем

$$2^m \leq (em)^{|E|} \Rightarrow m \leq |E| \log(em)/\log(2).$$

Утверждение теоремы теперь следует из леммы A.2. □

Далее рассмотрим класс $\mathcal{H}_{V,E,\sigma}$, где σ – сигмоидная функция. Как это ни удивительно, VC-размерность $\mathcal{H}_{V,E,\sigma}$ ограничена снизу величиной порядка $\Omega(|E|^2)$ (см. упражнение 20.5). Таким образом, VC-размерность равна квадрату количества настраиваемых параметров. Имеет место также оценка VC-размерности сверху величиной $O(|V|^2|E|^2)$, но доказательство этого факта выходит за рамки книги. Так или иначе, поскольку на практике мы рассматриваем только сети, в которых веса имеют короткое представление в виде чисел с плавающей точкой с $O(1)$ битами, то, воспользовавшись дискретизацией, легко получаем, что VC-размерность таких сетей составляет $O(|E|)$, даже если используется сигмоидная функция активации.

20.5. Время обучения нейронных сетей

В предыдущих разделах мы показали, что класс нейронных сетей с графом полиномиального размера может выразить все функции, допускающие эффективную реализацию, и что выборочная сложность очень удачно зависит от размера сети. В этом разделе мы обратимся к анализу временной сложности обучения нейронных сетей.

Сначала покажем, что реализация правила ERM относительно класса $\mathcal{H}_{V,E,\text{sign}}$ – NP-трудная задача даже для сетей с одним скрытым слоем, содержащим всего 4 нейрона.

Теорема 20.7. *Пусть $k \geq 3$. Для любого n пусть (V, E) – слоистый граф с n входными вершинами, $k + 1$ вершинами в (единственном) скрытом слое, одна из которых – постоянный нейрон, и одной выходной вершиной. Тогда реализовать правило ERM относительно класса $\mathcal{H}_{V,E,\text{sign}}$ NP-трудно.*

Доказательство опирается на сведение к задаче о k-раскраске и оставлено читателю в качестве упражнения 20.6.

Один из способов обойти этот результат о трудности мог бы заключаться в том, что для целей обучения достаточно найти предиктор $h \in \mathcal{H}$ с малой эмпирической ошибкой, а не обязательно в точности удовлетворяющий правилу ERM. Однако оказывается, что даже задача нахождения весов, приводящих к эмпирической ошибке, близкой к минимальной, вычислительно неразрешима (см. Bartlett & Ben-David, 2002).

Можно задаться вопросом, а нельзя ли изменить архитектуру сети и таким образом обойти результат о трудности. То есть не может ли получиться, что ERM относительно исходной структуры сети – вычислительно трудная задача, но ERM относительно другой, большей, сети можно реализовать эффективно (см. примеры такого рода в главе 8)? Еще одна возможность – использовать другие функции активации (например, сигмоиды или иные эффективно вычислимые функции). Есть сильные аргументы в пользу того, что все такие попытки обречены на провал. Действительно, известно, что при некоторых криптографиче-

ских предположениях проблема обучения пересечений полупространств трудна даже в независимой от представления модели обучения (см. Klivans & Sherstov, 2006). Отсюда следует, что при тех же криптографических предположениях любой класс гипотез, содержащий пересечения полупространств, невозможно обучить эффективно.

Для обучения нейронных сетей широко используется эвристика – метод СГС, который мы изучали в главе 14. Там было показано, что СГС – успешный обучаемый, если функция потерь выпукла. В нейронных сетях функция потерь очень далека от выпуклости. Тем не менее мы можем реализовать алгоритм СГС и надеяться найти разумное решение (и это действительно так в некоторых практически важных задачах).

20.6. СГС и обратное распространение

Проблема нахождения в классе $\mathcal{H}_{V,E,\sigma}$ гипотезы с низким риском сводится к проблеме настройки весов ребер. В этом разделе мы покажем, как применить эвристический поиск хороших весов с помощью алгоритма СГС. Мы будем предполагать, что σ – сигмоидная функция, $\sigma(a) = 1/(1 + e^{-a})$, но вывод остается в силе для любой дифференцируемой скалярной функции.

Поскольку множество E конечно, мы можем рассматривать весовую функцию как вектор $\mathbf{w} \in \mathbb{R}^{|E|}$. Предположим, что сеть имеет n входных и k выходных нейронов, и обозначим $h_{\mathbf{w}} : \mathbb{R}^n \to \mathbb{R}^k$ функцию, вычисляемую сетью, когда весовая функция определена вектором $\mathbf{w}$. Обозначим $\Delta(h_{\mathbf{w}}(\mathbf{x}), \mathbf{y})$ потерю в случае предсказания $h_{\mathbf{w}}(\mathbf{x})$, когда действительная цель равна $\mathbf{y} \in \mathcal{Y}$. Для определенности возьмем в качестве Δ квадратичную потерю $\Delta(h_{\mathbf{w}}(\mathbf{x}), \mathbf{y}) = \frac{1}{2}||h\mathbf{w}(\mathbf{x}) - \mathbf{y}||^2$, но рассуждение проходит для любой дифференцируемой функции. Наконец, пусть дано распределение $\mathcal{D}$ на множестве примеров $\mathbb{R}^n \times \mathbb{R}^k$, а $L_{\mathcal{D}}(\mathbf{w})$ – риск сети, т. е.

$$L_{\mathcal{D}}(\mathbf{w}) = \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}[\Delta(h_{\mathbf{w}}(\mathbf{x}), \mathbf{y})].$$

Напомним алгоритм СГС минимизации функции риска $L_{\mathcal{D}}(\mathbf{w})$. Мы повторяем псевдокод из главы 14 с небольшими модификациями, специфическими для нейронных сетей, поскольку целевая функция в этом случае невыпукла. Во-первых, в главе 14 мы инициализировали вектор $\mathbf{w}$ нулями, а здесь случайно выбираем вектор, элементы которого близки к нулю. Это связано с тем, выбор нулевого начального вектора приводит к тому, что у всех скрытых нейронов будут одинаковые веса (если сеть полносвязная). Кроме того, мы надеемся, что если повторять процедуру СГС многократно, каждый раз случайно выбирая новый начальный вектор, то хотя бы в одной из попыток будет найден хороший локальный минимум. Во-вторых, хотя фиксированный размер шага η гарантированно достаточно хорош для выпуклых задач, здесь мы используем переменный размер шага η_t, как было определено в разделе 14.4.2. Из-за невыпуклости функции потерь выбор последовательности η_t более важен и на практике подбор производится методом проб и ошибок. В-третьих, мы выводим вектор, показавший наилучшее качество на контрольном наборе. Дополнительно иногда полезно добавить регуляризацию весов с параметром λ. То есть мы пытаемся минимизировать функцию $L_{\mathcal{D}}(\mathbf{w}) + \lambda/2||\mathbf{w}||^2$. Наконец, градиент нельзя выразить в замкнутом виде.

Вместо этого он реализуется с помощью алгоритма обратного распространения, который мы опишем ниже.

СГС для нейронных сетей

параметры:
число итераций τ
последовательность размеров шагов $\eta_1, \eta_2, \ldots, \eta_\tau$
параметр регуляризации $\lambda > 0$
вход:
слоистый граф (V, E)
дифференцируемая функция активации $\sigma : \mathbb{R} \to \mathbb{R}$
инициализация:
случайным образом выбрать $\mathbf{w}^{(1)} \in \mathbb{R}^{|E|}$
(из такого распределения, что $\mathbf{w}^{(1)}$ достаточно близок к 0)
for $i = 1, 2, \ldots, \tau$
выбрать $(\mathbf{x}, \mathbf{y}) \sim \mathcal{D}$
вычислить градиент $\mathbf{v}_i$ = `backpropagation`$(\mathbf{x}, \mathbf{y}, \mathbf{w}, (V, E), \sigma)$
обновить $\mathbf{w}^{(i+1)} = \mathbf{w}^{(i)} - \eta_i(\mathbf{v}_i + \lambda\mathbf{w}^{(i)})$
выход:
вектор $\bar{\mathbf{w}} - \mathbf{w}^{(i)}$, показавший наилучшее качество на контрольном наборе

Процедура обратного распространения backpropagation

вход:
пример $(\mathbf{x}, \mathbf{y})$, вектор весов $\mathbf{w}$, слоистый граф (V, E),
функция активации $\sigma : \mathbb{R} \to \mathbb{R}$
инициализация:
обозначим слои графа $V_0, \ldots, V_T$, где $V_t = \{v_{t,1}, \ldots, v_{t,kt}\}$
определим $W_{t,i,j}$ вес ребра $(v_{t,j}, v_{t+1,i})$
(мы полагаем $W_{t,i,j} = 0$, если $(v_{t,j}, v_{t+1,i}) \notin E$)
вперед:
set $\mathbf{o}_0 = \mathbf{x}$
for $t = 1, \ldots, T$
for $i = 1, \ldots, k_t$
положить $a_{t,i} = \sum_{j=1}^{k_{t-1}} W_{t-1,i,j} o_{t-1,j}$
положить $o_{t,i} = \sigma(a_{t,i})$
назад:
положить $\delta_T = \mathbf{o}_T - \mathbf{y}$
for $t = T - 1, T - 2, \ldots, 1$
for $i = 1, \ldots, k_t$
$\delta_{t,i} = \sum_{j=1}^{k_{t+1}} W_{t,j,i} \delta_{t+1,j} \sigma'(a_{t+1,j})$
выход:
для каждого ребра $(v_{t-1,j}, v_{t,i}) \in E$
положить частную производную равной $\delta_{t,i}\sigma'(a_{t,i})o_{t-1,j}$

Как процедура обратного распространения вычисляет градиент

Далее мы объясним, как алгоритм обратного распространения вычисляет градиент функции потерь на примере $(\mathbf{x}, \mathbf{y})$ по вектору $\mathbf{w}$. Сначала вспомним несколько определений из векторного математического анализа. Каждый элемент градиента – это частная производная по элементу вектора $\mathbf{w}$, соответствующему одному из ребер графа сети. Напомним определение частной производной. Если дана функция $f : \mathbb{R}^n \to \mathbb{R}$, то частная производная по i-му элементу вектора $\mathbf{w}$ получается, если зафиксировать значения $w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n$, в результате чего получается скалярная функция $g : \mathbb{R} \to \mathbb{R}$, определенная как $g(a) = f(w_1, \ldots, w_{i-1}, w_i + a, w_{i+1}, \ldots, w_n)$, а затем взять производную g в точке 0. Для векторнозначной функции $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ *якобианом* $\mathbf{f}$ в точке $\mathbf{w} \in \mathbb{R}^n$, обозначаемым $J_{\mathbf{w}}(\mathbf{f})$, называется матрица $m \times n$, в которой (i, j)-м элементом является частная производная функции $f_i : \mathbb{R}^n \to \mathbb{R}$ по j-му элементу $\mathbf{w}$. Отметим, что если $m = 1$, якобиан (представленный в виде вектора-строки) оказывается градиентом функции. Ниже приведено два примера вычисления якобиана, которые пригодятся нам в дальнейшем.

- Пусть $\mathbf{f}(\mathbf{w}) = A\mathbf{w}$ для $A \in \mathbb{R}^{m,\,n}$. Тогда $J_{\mathbf{w}}(\mathbf{f}) = A$.
- Для любого n будем обозначать σ функцию из $\mathbb{R}^n$ в $\mathbb{R}^n$, которая применяет сигмоиду к каждому элементу матрицы, т. е. $\alpha = \sigma(\theta)$ означает, что для любого i имеем $\alpha_i = \sigma(\theta_i) = 1/(1 + \exp(-\theta_i))$. Легко проверить, что $J_\theta(\sigma)$ – диагональная матрица, в которой элемент (i, i) равен $\sigma'(\theta_i)$, где σ' – производная (скалярной) сигмоидной функции, равная $\sigma'(\theta_i) = 1/((1 + \exp(\theta_i))(1 + \exp(-\theta_i)))$. Мы будем также обозначать эту матрицу $\mathrm{diag}(\sigma'(\theta))$.

Правило дифференцирования сложной функции можно записать в терминах якобиана следующим образом. Пусть даны две функции $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ и $\mathbf{g} : \mathbb{R}^k \to \mathbb{R}^n$, тогда якобиан сложной функции $(\mathbf{f} \circ \mathbf{g}) : \mathbb{R}^k \to \mathbb{R}^m$ в точке $\mathbf{w}$ равен

$$J_{\mathbf{w}}(\mathbf{f} \circ \mathbf{g}) = J_{\mathbf{g}(\mathbf{w})}(\mathbf{f}) J_{\mathbf{w}}(\mathbf{g}).$$

Например, если $\mathbf{g}(\mathbf{w}) = A\mathbf{w}$, где $A \in \mathbb{R}^{n,k}$, то

$$J_{\mathbf{w}}(\sigma \circ \mathbf{g}) = \mathrm{diag}(\sigma'(A\mathbf{w}))A.$$

Чтобы описать алгоритм обратного распространения, сначала разложим V на слои $V = \biguplus_{t=0}^{T} V_t$. Для каждого t запишем $V_t = \{v_{t,1}, \ldots, v_{t,k_t}\}$, где $k_t = |V_t|$. Кроме того, для каждого t обозначим $W_t \in \mathbb{R}_{k_{t+1},k_t}$ матрицу, которая назначает вес каждому потенциальному ребру между V_t и V_{t+1}. Если такое ребро присутствует в E, то полагаем $W_{t,i,j}$ равным весу, назначенному функцией $\mathbf{w}$ ребру $(v_{t,j}, v_{t+1,i})$. В противном случае добавляем «фантомное» ребро и назначаем ему нулевой вес, $W_{t,i,j} = 0$. Поскольку при вычислении частной производной по весу некоторого ребра все остальные веса фиксируются, эти дополнительные «фантомные» ребра не оказывают никакого влияния на частные производные по весам других ребер. Следовательно, без ограничения общности можно предположить, что присутствуют все ребра, т. е. $E = \cup_t (V_t \times V_{t+1})$.

Далее обсудим, как вычислять частные производные по весам ребер из V_{t-1} в V_t, т. е. по элементам W_{t-1}. Поскольку все остальные веса в сети зафиксированы, выходы всех нейронов слоя V_{t-1} – фиксированные числа, не зависящие от весов в W_{t-1}. Обозначим соответствующий вектор $\mathbf{o}_{t-1}$. Кроме того, обозначим $\ell_t : \mathbb{R}^{k_t} \to \mathbb{R}$ потерю подсети, состоящей из слоев $V_t, \ldots, V_T$, выраженную в виде функции

от выходов нейронов слоя V_t. Входы нейронов слоя V_t можно записать в виде $\mathbf{a}_t = W_{t-1}\mathbf{o}_{t-1}$, а выход нейронов слоя V_t – в виде $\mathbf{o}_t = \boldsymbol{\sigma}(\mathbf{a}_t)$. Таким образом, для любого j имеем $o_{t,j} = \sigma(a_{t,j})$. Получается, что потерю в виде функции от W_{t-1} можно записать так:

$$g_t(W_{t-1}) = \ell_t(\mathbf{o}_t) = \ell_t(\boldsymbol{\sigma}(\mathbf{a}_t)) = \ell_t(\boldsymbol{\sigma}(W_{t-1}\mathbf{o}_{t-1})).$$

Будет удобно переписать эту формулу следующим образом. Пусть $\mathbf{w}_{t-1} \in \mathbb{R}^{k_{t-1}k_t}$ – вектор-столбец, полученный конкатенацией строк W_{t-1} и последующим транспонированием получившегося длинного вектора. Обозначим O_{t-1} матрицу размера $k_t \times (k_{t-1}k_t)$

$$O_{t-1} = \begin{pmatrix} \mathbf{o}_{t-1}^{\top} & 0 & \dots & 0 \\ 0 & \mathbf{o}_{t-1}^{\top} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{o}_{t-1}^{\top} \end{pmatrix}. \tag{20.2}$$

Тогда $W_{t-1}\mathbf{o}_{t-1} = O_{t-1}\mathbf{w}_{t-1}$, поэтому можно также написать

$$g_t(\mathbf{w}_{t-1}) = \ell_t(\boldsymbol{\sigma}(O_{t-1}\mathbf{w}_{t-1})).$$

Применяя правило дифференцирования сложной функции, получаем

$$J_{\mathbf{w}_{t-1}}(g_t) = J_{\boldsymbol{\sigma}(O_{t-1}\mathbf{w}_{t-1})}(\ell_t)\mathrm{diag}(\boldsymbol{\sigma}'(O_{t-1}\mathbf{w}_{t-1}))O_{t-1}.$$

В наших обозначениях имеем $\mathbf{o}_t = \boldsymbol{\sigma}(O_{t-1}\mathbf{w}_{t-1})$ и $\mathbf{a}_t = O_{t-1}\mathbf{w}_{t-1}$, что дает

$$J_{\mathbf{w}_{t-1}}(g_t) = J_{\mathbf{o}_t}(\ell_t)\mathrm{diag}(\boldsymbol{\sigma}'(\mathbf{a}_t))O_{t-1}.$$

Обозначим еще $\boldsymbol{\delta}_t = J_{\mathbf{o}_t}(\ell_t)$. Тогда можно еще раз переписать предыдущую формулу:

$$J_{\mathbf{w}_{t-1}}(g_t) = (\delta_{t,1}\sigma'(\mathbf{a}_{t,1})\mathbf{o}_{t-1}^{\top}, \dots, J_{\boldsymbol{\sigma}(O_{t-1}\mathbf{w}_{t-1})}(\ell_t)\mathrm{diag}(\boldsymbol{\sigma}'(O_{t-1}\mathbf{w}_{t-1}))O_{t-1}. \tag{20.3}$$

Осталось вычислить вектор $\delta_t = J_{\mathbf{o}_t}(\ell_t)$ для всех t. Это градиент функции ℓ_t в точке $\mathbf{o}_t$. Мы вычисляем его рекурсивно. Для начала заметим, что в последнем слое $\ell_T(\mathbf{u}) = \Delta(\mathbf{u}, \mathbf{y})$, где Δ – функция потерь. Поскольку мы предполагаем, что $\Delta(\mathbf{u}, \mathbf{y}) = \frac{1}{2}\|\mathbf{u}-\mathbf{y}\|^2$, то получаем, что $J_{\mathbf{u}}(\ell_T) = (\mathbf{u} - \mathbf{y})$. В частности, $\delta_T = J_{\mathbf{o}_T}(\ell_T) = \mathbf{o}_T - \mathbf{y}$. Далее заметим, что

$$\ell_t(\mathbf{u}) = \ell_{t+1}(\boldsymbol{\sigma}(W_t\mathbf{u})).$$

Поэтому, по правилу дифференцирования сложной функции

$$J_{\mathbf{u}}(\ell_t) = J_{\boldsymbol{\sigma}(W_t\mathbf{u})}(\ell_{t+1})\mathrm{diag}(\boldsymbol{\sigma}'(W_t\mathbf{u}))W_t.$$

В частности,

$$\begin{aligned} \boldsymbol{\delta}_t = J_{\mathbf{o}_t}(\ell_t) &= J_{\boldsymbol{\sigma}(W_t\mathbf{o}_t)}(\ell_{t+1})\mathrm{diag}(\boldsymbol{\sigma}'(W_t\mathbf{o}_t))W_t \\ &= J_{\mathbf{o}_{t+1}}(\ell_{t+1})\mathrm{diag}(\boldsymbol{\sigma}'(\mathbf{a}_{t+1}))W_t \\ &= \boldsymbol{\delta}_{t+1}\,\mathrm{diag}(\boldsymbol{\sigma}'(\mathbf{a}_{t+1}))W_t. \end{aligned}$$

В итоге получается следующее. Сначала мы можем вычислить векторы $\{\mathbf{a}_t, \mathbf{o}_t\}$, двигаясь по сети снизу вверх. Затем мы вычисляем векторы $\{\delta_t\}$, двигаясь по сети сверху вниз. Зная все эти векторы, мы легко можем вычислить частные производные, применяя выражение (20.3). Таким образом, мы показали, что псевдокод обратного распространения действительно вычисляет градиент.

ЛАНЬ®

20.7. Резюме

Нейронные сети на графах размера $s(n)$ можно использовать для описания классов всех предикторов, допускающих реализацию за время $O(\sqrt{s(n)})$. Мы также показали, что их выборочная сложность полиномиально зависит от $s(n)$ (точнее, от количества ребер в сети). Поэтому класс нейросетевых гипотез выглядит идеальным решением. К сожалению, задача обучения сети на обучающих данных является вычислительно трудной. Мы представили метод СГС как эвристический подход к обучению нейронных сетей и описали алгоритм обратного распространения, который эффективно вычисляет градиент функции потерь по весам ребер.

20.8. Библиографические сведения

Нейронные сети активно изучались в 1980-х и в начале 1990-х гг. с переменным эмпирическим успехом. В последние годы сочетание прогресса в области алгоритмов с постоянно возрастающими вычислительными мощностями и размером данных позволило совершить прорыв в эффективности нейронных сетей. В особенности «глубокие сети» (с число уровней больше 2) продемонстрировали впечатляющие успехи в самых разных областях. Назовем лишь несколько примеров: сверточные сети (Le Cun & Bengio, 1995), ограниченные машины Больцмана (Hinton, Osindero & Teh, 2006), автокодировщики (Ranzato et al., 2007; Bengio & Le Cun, 2007; Collobert & Weston, 2008; Lee et al., 2009; Le et al., 2012) и сети сумм произведений (Livni, Shalev-Shwartz & Shamir, 2013, Poon & Domingos, 2011). См. также работу Bengio (2009) и приведенные в ней ссылки.

Выразительная способность нейронных сетей и их связь со схемной сложностью были подробно изучены в работе Parberry (1994). Читателей, интересующихся анализом выборочной сложности нейронных сетей, отсылаем к работе Anthony & Bartlet (1999). Наше изложение доказательства теоремы 20.6 заимствовано из заметок к лекциям Какаде (Kakade) и Тевари (Tewari).

В работе Klivans and Sherstov (2006) показано, что для любого $c > 0$ пересечения n^c полупространств в $\{\pm 1\}^n$ не допускают эффективного PAC-обучения, даже если допустить обучение, не зависящее от представления. Этот результат о трудности опирается на криптографическое предположение о том, что не существует решения задачи о кратчайшем уникальном векторе с полиномиальным временем работы. Как мы сказали, отсюда следует отсутствие эффективного алгоритма обучения нейронных сетей, даже если разрешить более крупные сети или другие функции активации, которые могут быть реализованы эффективно.

Алгоритм обратного распространения был впервые описан в работе Rumelhart, Hinton and Williams (1986).

20.9. Упражнения

20.1. Нейронные сети являются универсальными аппроксиматорами. Пусть $f : [-1, 1]^n \to [-1, 1]$ – ρ-липшицева функция. Зафиксируем некоторое $\epsilon > 0$. Постройте нейронную сеть $N : [-1, 1]^n \to [-1, 1]$ с сигмоидной функцией активации такую, что для любого $\mathbf{x} \in [-1, 1]^n$ имеет место неравенство $|f(\mathbf{x}) - N(\mathbf{x})| \leq \epsilon$.

Указание. По аналогии с доказательством теоремы 19.3 разбейте множество $[-1, 1]^n$ на небольшие прямоугольники. Воспользовавшись липшицевостью функции f, покажите, что она приблизительно постоянна в каждом прямоугольнике. Наконец, покажите, что нейронная сеть может сначала решить, какому прямоугольнику принадлежит входной вектор, а затем предсказать усредненное значение f в этом прямоугольнике.

20.2. Докажите теорему 20.5.

Указание. Для любой функции $f : \{-1, 1\}^n \to \{-1, 1\}$ постройте 1-липшицеву функцию $g : [-1, 1]^n \to [-1, 1]$ такую, что если можно аппроксимировать g, то можно выразить f.

ЛАНЬ®

20.3. Функция роста произведения. Для $i = 1, 2$ обозначим $\mathcal{F}_i$ множество функций из $\mathcal{X}$ в $\mathcal{Y}_i$. Определим класс декартова произведения $\mathcal{H} = \mathcal{F}_1 \times \mathcal{F}_2$. Это означает, что для любых $f_1 \in \mathcal{F}_1$ и $f_2 \in \mathcal{F}_2$ существует $h \in \mathcal{H}$ такая, что $h(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))$. Докажите, что $\tau_{\mathcal{H}}(m) \leq \tau_{\mathcal{F}_1}(m)\, \tau_{\mathcal{F}_2}(m)$.

20.4. Функция роста композиции. Обозначим $\mathcal{F}_1$ множество функций из $\mathcal{X}$ в Z, а $\mathcal{F}_2$ – множество функции из Z в $\mathcal{Y}$. Определим класс композиции $\mathcal{H} = \mathcal{F}_2 \circ \mathcal{F}_1$. Это означает, что для любых $f_1 \in \mathrm{F}_1$ и $f_2 \in \mathrm{F}_2$ существует $h \in \mathcal{H}$ такая, что $h(\mathbf{x}) = f_2(f_1(\mathbf{x}))$. Докажите, что $\tau_{\mathcal{H}}(m) \leq \tau_{\mathcal{F}_2}(m)\, \tau_{\mathcal{F}_1}(m)$.

20.5. VC-размерность сигмоидных сетей. В этом упражнении мы покажем, что существует граф (V, E) такой, что VC-размерность класса нейронных сетей на этом графе с сигмоидной функцией активации равна $\Omega(|E|^2)$. Заметим, что для любого $\epsilon > 0$ сигмоидная функция активации может аппроксимировать ступенчатую функцию активации $\mathbb{1}_{[\sum_i x_i]}$ с точностью ϵ. Чтобы упростить изложение, будем в этом упражнении предполагать, что мы можем точно реализовать функцию активации $\mathbb{1}_{[\sum_i x_i > 0]}$ с помощью сигмоиды.

Зафиксируем некоторое n.

1. Постройте сеть N_1 с $O(n)$ весами, которая реализует функцию из $\mathbb{R}$ в $\{0, 1\}^n$ и удовлетворяет следующему свойству: для любого $\mathbf{x} \in \{0, 1\}^n$, если подать на вход сети вещественное число $0.x_1x_2...\, x_n$, то на выходе получится $\mathbf{x}$.
   Указание. Обозначьте $\alpha = 0.x_1x_2...\, x_n$ и заметьте, что $10^k\alpha - 0{,}5$ не меньше 0,5, если $x_k = 1$, и не больше $-0{,}3$, если $x_k = 0$.
2. Постройте сеть N_2 с $O(n)$ весами, которая реализует функцию из $[n]$ в $\{0, 1\}^n$ такую, что $N_2(i) = \mathbf{e}_i$ для всех i. То есть, получив на входе i, сеть выводит вектор, все элементы которого равны 0 за исключением 1, соответствующей i-му нейрону.
3. Пусть $\alpha_1, ..., \alpha_n$ – n вещественных чисел таких, что каждое α_i имеет вид $0.a_1^{(i)} a_2^{(i)}...a_n^{(i)}$, где $a_j^{(i)} \in \{0, 1\}$. Постройте сеть N_3 с $O(n)$ весами, которая реализует

функцию из $[n]$ в $\mathbb{R}$ и удовлетворяет условию $N_2(i) = \alpha_i$ для любого $i \in [n]$.

4. Объединив N_1 и N_3, получите сеть, которая получает $i \in [n]$ и выводит $\mathbf{a}^{(i)}$.
5. Постройте сеть N_4, которая получает $(i, j) \in [n] \times [n]$ и выводит $a_j^{(i)}$.
   Указание. Заметьте, что функцию AND на $\{0, 1\}^2$ можно вычислить, используя $O(1)$ весов.
6. Сделайте вывод, что существует граф с $O(n)$ весами такой, что VC-размерность соответствующего класса гипотез равна n^2.

20.6. Докажите теорему 20.7.

Указание. Доказательство похоже на доказательство теоремы о трудности обучения пересечений полупространств – см. упражнение 8.3 в главе 8.

ЛАНЬ®

ЛАНЬ®

Часть *III*

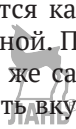
ДОПОЛНИТЕЛЬНЫЕ МОДЕЛИ ОБУЧЕНИЯ

ЛАНЬ®

Глава 21

ОНЛАЙНОВОЕ ОБУЧЕНИЕ

В этой главе мы опишем другую модель обучения – *онлайновое* обучение. Раньше мы изучали модель PAC-обучения, когда обучаемый сначала получает пакет обучающих примеров, на его основе вырабатывает гипотезу и лишь по завершении обучения использует эту гипотезу для предсказания меток новых примеров. В задаче о папайе это означает, что сначала нужно купить много плодов и попробовать их все. Затем мы воспользуемся полученной информацией, чтобы обучить правило предсказания, которое будет определять вкус новых плодов. Напротив, в онлайновом обучении нет четкого разделения между этапом обучения и этапом предсказания. Вместо этого каждый раз, как мы покупаем новую папайю, она сначала рассматривается как *тестовый* пример, поскольку мы должны предсказать, будет ли она вкусной. После этого, откусив кусочек, мы узнаем истинную метку и используем тот же самый плод, как *обучающий* пример, который поможет лучше предсказывать вкус будущих плодов.



В действительности онлайновое обучение имеет вид последовательности раундов. В каждом раунде обучаемый сначала получает образец (покупает папайю и оценивает ее цвет и твердость, т. е. атрибуты образца). Затем обучаемый должен предсказать метку (будет ли папайя вкусной). В конце раунда обучаемый получает правильную метку (пробует папайю и на опыте узнает, вкусная она или нет). И, наконец, обучаемый использует эту информацию для улучшения будущих предсказаний.

В процессе анализа онлайнового обучения мы пойдем тем же путем, что при исследовании PAC-обучения. Начнем с проблем онлайновой бинарной классификации. Мы рассмотрим как реализуемый случай, когда в качестве априорного знания предполагается, что все метки генерируются некоторой гипотезой из какого-то класса гипотез, так и нереализуемый случай, соответствующий агностической модели PAC-обучения. В частности, мы представим важный алгоритм *взвешенного большинства*. Далее мы изучим проблемы онлайнового обучения с выпуклой функцией потерь. И наконец, опишем алгоритм перцетрона как пример использования суррогатной выпуклой функции потерь в модели онлайнового обучения.

21.1. Онлайновая классификация в реализуемом случае

Онлайновое обучение имеет вид последовательности раундов: в раунде t обучаемому предлагают образец $\mathbf{x}_t$, взятый из множества $\mathcal{X}$, и просят определить его метку. Предсказанную метку будем обозначать p_t. Затем обучаемому сообщают правильную метку $y_t \in \{0, 1\}$. Цель обучаемого – допустить как можно меньше ошибок предсказания в этом процессе. Обучаемый пытается вывести информацию из предыдущих раундов, чтобы улучшить будущие предсказания.

Очевидно, что все попытки обучения безнадежны, если между прошлыми и настоящим рандом нет никакой корреляции. Ранее в этой книге мы изучали модель PAC, в которой предполагалось, что прошлые примеры, как и настоящий, независимо выбираются из одного и того же распределения. В онлайновой модели обучения мы не делаем никаких статистических предположений об источнике последовательности примеров. Последовательность может быть детерминированной, стохастической или даже специально противодействующей обучаемому с учетом его поведения (как в случае фильтрации спама). Естественно, противник может сделать так, что число ошибок предсказания нашего алгоритма онлайнового обучения будет сколь угодно большим. Например, противник может в каждом раунде предлагать один и тот же образец, дождаться предсказания обучаемого, а затем сообщить противоположную метку как правильную.

Чтобы высказывать нетривиальные утверждения, мы должны наложить дополнительные ограничения. Одно из таких естественных ограничений – предположение о реализуемости. В реализуемом случае предполагается, что все метки порождаются некоторой гипотезой $h^* : \mathcal{X} \to \mathcal{Y}$. Кроме того, h^* берется из класса гипотез $\mathcal{H}$, известного обучаемому. Это аналог модели PAC-обучения, которую мы изучали в главе 3. При таком ограничении на последовательность обучаемый должен делать как можно меньше ошибок, предполагая, что и h^*, и последовательность образцов могут быть выбраны противником. Если имеется алгоритм обучения A, то будем обозначать $M_A(\mathcal{H})$ максимальное число ошибок, которые A может сделать на последовательности примеров, помеченных некоторой гипотезой $h^* \in \mathcal{H}$. Еще раз подчеркнем, что и h^*, и последовательность образцов могут выбираться противником. Верхняя граница $M_A(\mathcal{H})$ называется *границей ошибок* (mistake-bound), и мы будем заниматься проектированием алгоритмов, в которых $M_A(\mathcal{H})$ минимальна.

Определение 21.1 (граница ошибок, онлайновая обучаемость). Пусть $\mathcal{H}$ – класс гипотез, и A – алгоритм онлайнового обучения. Если дана последовательность $S = (x_1, h^*(y_1)), \ldots, (x_T, h^*(y_T))$, где T – целое число, а $h^* \in \mathcal{H}$, то обозначим $M_A(S)$ число ошибок, допускаемых A на последовательности S. Обозначим $M_A(\mathcal{H})$ верхнюю грань $M_A(S)$ на всех последовательностях описанного вида. Граница вида $M_A(\mathcal{H}) \leq B < \infty$ называется *границей ошибок*. Говорят, что класс гипотез $\mathcal{H}$ допускает онлайновое обучение, если существует алгоритм A, для которого $M_A(\mathcal{H}) \leq B < \infty$.

Наша цель – изучить, какие классы гипотез допускают обучение в онлайновой модели, и, в частности, отыскать хорошие алгоритмы обучения для заданного класса гипотез.

Замечание 21.1. В этом и в следующем разделе вычислительный аспект обучения игнорируется, мы не требуем, чтобы алгоритм был эффективным. В разделах 21.3 и 21.4 мы будем изучать эффективные алгоритмы онлайнового обучения.

Чтобы упростить изложение, начнем со случая конечного класса гипотез, т. е. $|\mathcal{H}| < \infty$.

В модели PAC мы признали ERM хорошим алгоритмом обучения в том смысле, что если класс $\mathcal{H}$ вообще допускает обучение, то он обучаем и по правилу $\text{ERM}_{\mathcal{H}}$. В онлайновом обучении естественно использовать (в любом раунде) в качестве правила обучения любую ERM-гипотезу, т. е. любую гипотезу, согласованную со всеми прошлыми примерами.

Consistent

вход: конечный класс гипотез $\mathcal{H}$
инициализация: $V_1 = \mathcal{H}$
for $t = 1, 2, \ldots$
  получить $\mathbf{x}_t$
  выбрать произвольную гипотезу $h \in V_t$
  предсказать $p_t = h(\mathbf{x}_t)$
  получить истинную метку $y_t = h^*(\mathbf{x}_t)$
  обновить $V_{t+1} = \{h \in V_t : h(\mathbf{x}_t) = y_t\}$

Алгоритм `Consistent` хранит множество V_t всех гипотез, согласованных с $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{t-1}, y_{t-1})$. Это множество часто называют *пространством версий*. Затем он выбирает произвольную гипотезу из V_t и делает предсказание в соответствии с этой гипотезой.

Очевидно, что всякий раз как `Consistent` допускает ошибку предсказания, по крайней мере одна гипотеза удаляется из V_t. Поэтому после M ошибок мы будем иметь $|V_t| \leq |\mathcal{H}| - M$. Поскольку V_t всегда непусто (в силу предположения о реализуемости, оно содержит h^*), имеем $1 \leq |V_t| \leq |\mathcal{H}| - M$. После перегруппировки получаем такой результат.

Следствие 21.2. *Пусть $\mathcal{H}$ – конечный класс гипотез. Для алгоритма `Consistent` справедлива следующая граница ошибок $M_{Consistent}(\mathcal{H}) \leq |\mathcal{H}| - 1$.*

Довольно легко построить класс гипотез и последовательность примеров, для которых `Consistent` действительно делает $|\mathcal{H}| - 1$ ошибок (см. упражнение 21.1.) Поэтому мы представим алгоритм получше, в котором гипотеза $h \in V_t$ выбирается более разумно. Мы увидим, что этот алгоритм гарантированно допускает экспоненциально меньше ошибок.

Halving

вход: конечный класс гипотез $\mathcal{H}$
инициализация: $V_1 = \mathcal{H}$
for $t = 1, 2, \ldots$
  получить $\mathbf{x}_t$
  предсказать $p_t = \text{argmax}_{r \in \{0,1\}} |\{h \in V_t : h(\mathbf{x}_t) = r\}|$
    (при неоднозначности предсказать $p_t = 1$)
  получить истинную метку $y_t = h^*(\mathbf{x}_t)$
  обновить $V_{t+1} = \{h \in V_t : h(\mathbf{x}_t) = y_t\}$

Теорема 21.3. *Пусть $\mathcal{H}$ – конечный класс гипотез. Для алгоритма* `Halving` *справедлива следующая граница ошибок* $M_{Halving}(\mathcal{H}) \le \log_2(|\mathcal{H}|)$.

Доказательство. Просто заметим, что всякий раз как алгоритм ошибается, мы имеем $|V_{t+1}| \le |V_t|/2$ (отсюда и название Halving – деление пополам). Поэтому, если M – общее число ошибок, то имеем

$$1 \le |V_{T+1}| \le |\mathcal{H}| 2^{-M}.$$

Логарифмирование крайних частей неравенства завершает доказательство. □

Разумеется, граница ошибок в алгоритме `Halving` гораздо лучше, чем в `Consistent`. Мы уже видим, что онлайновое обучение отличается от PAC-обучения – если в PAC любая ERM-гипотеза хороша, то в онлайновом обучении выбор произвольной ERM-гипотезы далек от оптимальности.

21.1.1. Онлайновая обучаемость

Далее мы подойдем к проблеме с более общей точки зрения и постараемся охарактеризовать онлайновую обучаемость. В частности, нас будет интересовать такой вопрос: «Каков оптимальный алгоритм онлайнового обучения для данного класса гипотез $\mathcal{H}$?».

Мы опишем размерность классов гипотез, которая характеризует наилучшую достижимую границу ошибок. Эта метрика была предложена Ником Литлстоуном (Nick Littlestone) и потому обозначается Ldim($\mathcal{H}$).

Чтобы обосновать определение Ldim, полезно рассматривать процесс онлайнового обучения как игру двух лиц: обучаемого и окружения. В t-м раунде игры окружение выбирает образец $\mathbf{x}_t$, обучаемый предсказывает метку $p_t \in \{0, 1\}$, после чего окружение сообщает истинную метку $y_t \in \{0, 1\}$. Предположим, что окружение хочет заставить обучаемого ошибаться в первых T раундах игры. Тогда оно должно выдать $y_t = 1 - p_t$, и остается только один вопрос – как выбирать образцы $\mathbf{x}_t$, чтобы гарантировать, что для некоторой гипотезы $h^* \in \mathcal{H}$ имеет место равенство $y_t = h^*(\mathbf{x}_t)$ для всех $t \in [T]$.

Стратегию противоборствующего окружения можно формально описать в виде двоичного дерева. Каждый узел дерева ассоциирован с образцом из $\mathcal{X}$. В начальный момент окружение предлагает обучаемому образец, ассоциированный

с корнем дерева. Затем, если обучаемый предсказывает $p_t = 1$, то окружение объявляет это предсказание неверным (т. е. $y_t = 0$) и переходит к правому потомку текущего узла. Если же обучаемый предсказывает $p_t = 0$, то окружение полагает $y_t = 1$ и переходит к левому потомку. Этот процесс продолжается, и в каждом раунде окружение предлагает образец, ассоциируемый с текущим узлом.

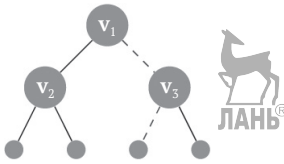
Опишем этот процесс формально. Рассмотрим полное двоичное дерево глубины T (мы определяем глубину, как количество ребер на пути от корня к листу). В таком дереве имеется $2^{T+1} - 1$ узлов, и к каждому узлу мы присоединяем образец. Обозначим эти образцы $\mathbf{v}_1, \ldots, \mathbf{v}_{2^{T+1}-1}$. Мы начинаем с корня дерева и полагаем $\mathbf{x}_1 = \mathbf{v}_1$. В t-м раунде мы полагаем $\mathbf{x}_t = \mathbf{v}_{i_t}$, где i_t – текущий узел. В конце t-го раунда мы переходим к левому потомку i_t, если $y_t = 0$, и к правому потомку, если $y_t = 1$. Таким образом, $i_{t+1} = 2i_t + y_t$. Раскручивая рекурсию, приходим к выводу, что $i_t = 2^{t-1} + \sum_{j=1}^{t-1} y_j 2^{t-1-j}$.

Эта стратегия окружения приносит успех, только если для любого $(y_1, \ldots, y_T)$ существует $h \in \mathcal{H}$ такая, что $y_t = h(\mathbf{x}_t)$ для всех $t \in [T]$. Таким образом, мы приходим к следующему определению.

ЛАНЬ®

Определение 21.4 (дерево, расколотое $\mathcal{H}$). Расколотым деревом (shattered tree) глубины d называется последовательность образцов $\mathbf{v}_1, \ldots, \mathbf{v}_{2^d-1}$ из $\mathcal{X}$ такая, что для любой пометки $(y_1, \ldots, y_d) \in \{0, 1\}^d$ существует такая гипотеза $h \in \mathcal{H}$, что для всех $t \in [d]$ имеет место равенство $h(\mathbf{v}_{i_t}) = y_t$, где $i_t = 2^{t-1} + \sum_{j=1}^{t-1} y_j 2^{t-1-j}$.

На рис. 21.1 приведен пример расколотого дерева глубины 2.



ЛАНЬ®

	h_1	h_2	h_3	h_4
$\mathbf{v}_1$	0	0	1	1
$\mathbf{v}_2$	0	1	*	*
$\mathbf{v}_3$	*	*	0	1

Рис. 21.1. Иллюстрация расколотого дерева глубины 2. Штриховой путь соответствует последовательности примеров $((\mathbf{v}_1, 1), (\mathbf{v}_3, 0))$. Дерево расколото классом $\mathcal{H} = \{h_1, h_2, h_3, h_4\}$, а предсказания каждой гипотезы из $\mathcal{H}$ на образцах $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ приведены в таблице (символ * означает, что $h_j(\mathbf{v}_i)$ может быть равно как 1, так и 0)

Определение 21.5. Размерностью Литлстоуна $\mathrm{Ldim}(\mathcal{H})$ называется максимальное целое число T такое, что существует дерево глубины T, расколотое $\mathcal{H}$.

Из определения Ldim и обсуждения выше немедленно вытекает следующая лемма.

Лемма 21.6. *Никакой алгоритм не может иметь границу ошибок, строго меньшую* $\mathrm{Ldim}(\mathcal{H})$, *т. е. для любого алгоритма* $AM_A(\mathcal{H}) \geq \mathrm{Ldim}(\mathcal{H})$.

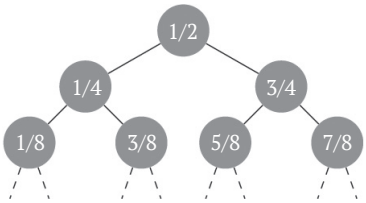
Доказательство. Пусть $T = \mathrm{Ldim}(\mathcal{H})$ и $\mathbf{v}_1, \ldots, \mathbf{v}_{2^T - 1}$ – последовательность, удовлетворяющая требованиям в определении Ldim. Если окружение полагает $\mathbf{x}_t = \mathbf{v}_{i_t}$ и $y_t = 1 - p_t$ для всех $t \in [T]$, то обучаемый сделает T ошибок, тогда как из определения Ldim следует, что существует гипотеза $h \in \mathcal{H}$ такая, что $y_t = h(\mathbf{x}_t)$ для всех t. □

Теперь приведем несколько примеров.

Пример 21.2. Пусть $\mathcal{H}$ – конечный класс гипотез. Очевидно, что глубина любого дерева, расколотого $\mathcal{H}$, не превышает $\log_2(|\mathcal{H}|)$. Поэтому $\text{Ldim}(\mathcal{H}) \le \log_2(|\mathcal{H}|)$. Это неравенство можно вывести и по-другому, объединив лемму 21.6 с теоремой 21.3.

Пример 21.3. Пусть $\mathcal{X} = \{1, \ldots, d\}$ и $\mathcal{H} = \{h_1, \ldots, h_d\}$, где $h_j(x) = 1$ тогда и только тогда, когда $x = j$. Тогда легко показать, что $\text{Ldim}(\mathcal{H}) = 1$, хотя $|\mathcal{H}| = d$ может быть произвольно большим. Следовательно, этот пример показывает, что $\text{Ldim}(\mathcal{H})$ может быть значительно меньше $\log_2(|\mathcal{H}|)$.

Пример 21.4. Пусть $\mathcal{X} = [0, 1]$ и $\mathcal{H} = \{x \mapsto \mathbb{1}_{[x<a]} : a \in [0, 1]\}$, т. е. $\mathcal{H}$ – класс ступенчатых функций на отрезке [0, 1]. Тогда $\text{Ldim}(\mathcal{H}) = \infty$. Чтобы убедиться в этом, рассмотрим дерево.



Это дерево расколото $\mathcal{H}$. А в силу плотности множества вещественных чисел его можно сделать произвольно глубоким.

Лемма 21.6 утверждает, что $\text{Ldim}(\mathcal{H})$ ограничивает снизу границу ошибок любого алгоритма. Интересно, что существует стандартный алгоритм, для которого граница ошибок совпадает с этой нижней границей. Он похож на алгоритм `Halving`. Напомним, что `Halving` делает предсказание, ориентируясь на результаты большинства гипотез, согласованных с предыдущими примерами. Мы обозначили это множество V_t. Иначе говоря, `Halving` разбивает V_t на два множества: $V_t^+ = \{h \in V_t : h(\mathbf{x}_t) = 1\}$ и $V_t^- = \{h \in V_t : h(\mathbf{x}_t) = 0\}$. Затем он предсказывает ту же метку, что и большая из этих двух групп. Такая стратегия обоснована тем, что если `Halving` и делает ошибку, то в итоге оказывается, что $|V_{t+1}| \le 0{,}5|V_t|$.

В оптимальном алгоритме, показанном ниже, используется та же идея, но вместо того, чтобы предсказывать метку, как больший класс, он предсказывает, как класс с большей размерностью Ldim.

Стандартный оптимальный алгоритм (COA)

вход: класс гипотез $\mathcal{H}$
инициализация: $V_1 = \mathcal{H}$
for $t = 1, 2, \ldots$
  получить $\mathbf{x}_t$
  for $r \in \{0, 1\}$ положить $V_t^{(r)} = \{h \in V_t : h(\mathbf{x}_t) = r\}$
  предсказать $p_t = \text{argmax}_{r \in \{0,1\}} \text{Ldim}(V_t^{(r)})$
    (в случае неоднозначности предсказать $p_t = 1$)
  получить истинную метку y_t
  обновить $V_{t+1} = \{h \in V_t : h(\mathbf{x}_t) = y_t\}$

Следующая лемма формально устанавливает оптимальность этого алгоритма.

Лемма 21.7. *Граница ошибок для* $COAM_{SOA}(\mathcal{H}) \le \text{Ldim}(\mathcal{H})$.

Доказательство. Достаточно доказать, что всякий раз как алгоритм ошибается при предсказании, имеет место $\text{Ldim}(V_{t+1}) \le \text{Ldim}(V_t) - 1$. Мы докажем это, предположив противное, т. е. что $\text{Ldim}(V_{t+1}) = \text{Ldim}(V_t)$. Если это верно, то из определения p_t следует, что $\text{Ldim}(V_t^{(r)}) = \text{Ldim}(V_t)$ для $r = 1$ и $r = 0$. Но тогда мы можем построить расколотое дерево глубины $\text{Ldim}(V_t) + 1$ для класса V_t, что ведет к противоречию. □

Объединяя лемму 21.7 с леммой 21.6, получаем

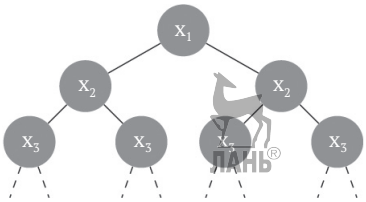
Следствие 21.8. *Пусть* $\mathcal{H}$ *– произвольный класс гипотез. Тогда граница ошибок для стандартного оптимального алгоритма* $M_{SOA}(\mathcal{H}) = \text{Ldim}(\mathcal{H})$, *и ни для какого другого алгоритма не может выполняться неравенство* $M_A(\mathcal{H}) < \text{Ldim}(\mathcal{H})$.

Сравнение с VC-размерностью

В модели PAC-обучения обучаемость характеризуется VC-размерностью $\mathcal{H}$. Напомним, что VC-размерностью класса $\mathcal{H}$ называется максимальное число d такое, что существуют образцы $\mathbf{x}_1, \ldots, \mathbf{x}_d$, разбиваемые $\mathcal{H}$. То есть для любой последовательности меток $(y_1, \ldots, y_d) \in \{0, 1\}^d$ существует гипотеза $h \in \mathcal{H}$, которая дает в точности такую последовательность. Следующая теорема устанавливает связь между VC-размерностью и размерностью Литлстоуна.

Теорема 21.9. *Для любого класса* $\mathcal{H}$ $\text{VCdim}(\mathcal{H}) \le \text{Ldim}(\mathcal{H})$ *и существуют классы, для которых это неравенство строгое. Более того, разность между обеими размерностями может быть сколько угодно велика.*

Доказательство. Сначала докажем, что $\text{VCdim}(\mathcal{H}) \le \text{Ldim}(\mathcal{H})$. Предположим, что $\text{VCdim}(\mathcal{H}) = d$ и пусть $\mathbf{x}_1, \ldots, \mathbf{x}_d$ – разбитое множество. Теперь построим полное двоичное дерево, содержащее образцы $\mathbf{v}_1, \ldots, \mathbf{v}_{2d-1}$, в котором все узлы глубины i равны $\mathbf{x}_i$, как показано на рисунке ниже.



Теперь из определения разбитого множества с очевидностью следует, что мы получили корректное расколотое дерево глубины d, откуда мы делаем вывод, что $\text{VCdim}(\mathcal{H}) \le \text{Ldim}(\mathcal{H})$. Чтобы показать, что разность между обеими размерностями может быть сколь угодно велика, просто заметим, что VC-размерность класса из примера 21.4 равна 1, а его размерность Литлстоуна бесконечна. □

21.2. Онлайновая классификация в нереализуемом случае

В предыдущем разделе мы изучали онлайновую обучаемость в реализуемом случае. Теперь же рассмотрим нереализуемый случай. По аналогии с агностической моделью PAC мы больше не предполагаем, что все метки порождаются некоторой гипотезой $h^* \in \mathcal{H}$, но требуем, чтобы обучаемый конкурировал с наилучшим фиксированным предиктором из $\mathcal{H}$. Это улавливается *сожалением* (regret) алгоритма, которое измеряет, насколько обучаемый в ретроспективе «жалеет», что не слушал предсказаний некоторой гипотезы $h \in \mathcal{H}$. Формально сожаление алгоритма A относительно h при выполнении на последовательности T примеров определяется следующим образом:

$$\mathrm{Regret}_A(h,T) = \sup_{(x_1,y_1),\ldots,(x_T,y_T)} \left[\sum_{t=1}^{T} |p_t - y_t| - \sum_{t=1}^{T} |h(x_t) - y_t| \right], \tag{21.1}$$

а сожаление алгоритма относительно класса гипотез $\mathcal{H}$ по определению равно

$$\mathrm{Regret}_A(\mathcal{H},T) = \sup_{h \in \mathcal{H}} \mathrm{Regret}_A(h,T). \tag{21.2}$$

Мы переформулируем цель обучения: требуется достичь наименьшего возможного сожаления относительно $\mathcal{H}$. Интересен вопрос, можем ли мы спроектировать алгоритм с низким сожалением, понимая под этим, что $\mathrm{Regret}_A(\mathcal{H}, T)$ сублинейно зависит от количества раундов T, откуда следует, что разность между *частотой ошибок* обучаемого и наилучшей гипотезы в $\mathcal{H}$ стремится к нулю, когда T стремится к бесконечности.

Сначала мы покажем, что эта миссия невыполнима – никакой алгоритм не может достичь сублинейной границы сожаления, даже при $|\mathcal{H}| = 2$. Действительно, рассмотрим $\mathcal{H} = \{h_0, h_1\}$, где h_0 – функция, которая всегда возвращает 0, а h_1 – функция, всегда возвращающая 1. Противник может сделать число ошибок любого онлайнового алгоритма равным T, если просто подождет предсказания обучаемого, а затем объявит истинной противоположную метку. С другой стороны, для любой последовательности истинных меток $y_1, \ldots, y_T$ пусть b равно большинству меток среди $y_1, \ldots, y_T$, тогда число ошибок гипотезы h_b не превышает $T/2$. Значит, сожаление любого онлайнового алгоритма будет равно как минимум $T - T/2 = T/2$, т. е. зависимость от T не сублинейная. Этот результат о невозможности приписывается Каверу (Cover, 1965).

Чтобы обойти результат Кавера о невозможности, мы должно наложить дополнительные ограничения на возможности противоборствующего окружения. Для этого мы разрешим обучаемому рандомизировать предсказания. Конечно, само по себе это не поможет обойти результат Кавера, поскольку при его выводе мы не делали никаких предположений о стратегии обучаемого. Чтобы рандомизация имела смысл, мы потребуем, чтобы окружение принимало решение о y_t, ничего не зная о монете, которую обучаемый подбрасывает в t-м раунде. Противник может знать о стратегии предсказания обучаемого, ему даже могут

быть известны результаты случайного подбрасывания монет в предыдущих раундах, но он не знает, какое значение выпало в раунде t. При таком (не слишком существенном) изменении правил игры мы проанализируем *ожидаемое* количество ошибок алгоритма, считая, что математическое ожидание вычисляется относительно собственной рандомизации обучаемого. То есть, если обучаемый выводит $\hat{y}_t$, где $\mathbb{P}[\hat{y}_t = 1] = p_t$, то ожидаемая потеря, которую он несет в t-м раунде, равна

$$\mathbb{P}[\hat{y}_t \neq y_t] = |p_t - y_t|.$$

Иначе говоря, множество предсказаний обучаемого теперь не {0, 1}, а весь отрезок [0, 1], и $p_t \in [0, 1]$ интерпретируется как вероятность предсказать метку 1 в t-м раунде.

При таком предположении становится возможно спроектировать алгоритм с низким сожалением. В частности, мы докажем следующую теорему.

Теорема 21.10. *Для любого класса гипотез $\mathcal{H}$ существует алгоритм онлайновой классификации, предсказания которого берутся из отрезка* [0, 1], *с сожалением, ограниченным сверху величиной*

$$\forall h \in \mathcal{H}, \sum_{t=1}^{T} |p_t - y_t| - \sum_{t=1}^{T} |h(\mathbf{x}_t) - y_t| \leq \sqrt{2 \min\{\log(|\mathcal{H}|), \mathrm{Ldim}(\mathcal{H}) \log(eT)\} T}.$$

Кроме того, ни для какого алгоритма граница ожидаемого сожаления не может быть меньше $\Omega(\sqrt{\mathrm{Ldim}(\mathcal{H})T})$.

Мы приведем конструктивное доказательство верхней границы в этой теореме. Доказательство нижней границы можно найти в работе Ben-David, Pal & Shalev-Shwartz (2009).

Доказательство теоремы 21.10 опирается на алгоритм *взвешенного большинства*, предназначенный для обучения с советом эксперта. Этот алгоритм важен сам по себе, поэтому мы посвятим ему весь следующий подраздел.

21.2.1. Алгоритм взвешенного большинства

Алгоритм взвешенного большинства предназначен для проблемы *предсказания с советом эксперта*. В этой проблеме онлайнового обучения в t-м раунде обучаемый должен выбрать один из советов d экспертов. Мы также разрешаем обучаемому рандомизировать свой выбор, определив распределение на d экспертах. То есть обучаемый может взять вектор $\mathbf{w}^{(t)} \in [0, 1]^d$, для которого $\sum_i w_i^{(t)} = 1$ и выбирать i-го эксперта с вероятностью $w_i^{(t)}$. Выбрав эксперта, обучаемый получает вектор $\mathbf{v}_t \in [0, 1]^d$, где $v_{t,i}$ – стоимость следования совету i-го эксперта. Если предсказания обучаемого рандомизированы, то его потеря определяется как усредненная стоимость, т. е. $\sum_i w_i^{(t)} v_{t,i} = \langle w^{(t)}, \mathbf{v}_t \rangle$. Алгоритм предполагает, что количество раундов T задано. В упражнении 21.4 мы покажем, как избавиться от этого предположения с помощью *трюка удвоения*.

Алгоритм Weighted-Majority

вход: количество экспертов d; количество раундов T
параметр: $\eta = \sqrt{2\log(d)/T}$
инициализация: $\tilde{\mathbf{w}}^{(1)} = (1, \ldots, 1)$
for $t = 1, 2, \ldots$
положить $\mathbf{w}^{(t)} = \tilde{\mathbf{w}}^{(t)}/Z_t$, где $Z_t = \sum_i \tilde{w}_i^{(t)}$
выбрать i-го эксперта случайным образом с вероятностью $\mathbb{P}[i] = w_i^{(t)}$
получить стоимости всех экспертов $\mathbf{v}_t \in [0, 1]^d$
уплатить стоимость $\langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle$
правило обновления $\forall i \tilde{w}_i^{(t+1)} = \tilde{w}_i^{(t)} e^{-\eta v_{t,i}}$

Следующая теорема – ключ к анализу границы сожаления алгоритма взвешенного большинства.

Теорема 21.11. *В предположении, что* $T > 2\log(d)$, *алгоритм взвешенного большинства имеет границу*

$$\sum_{t=1}^{T} \langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle - \min_{i \in [d]} \sum_{t=1}^{T} v_{t,i} \leq \sqrt{2\log(d)T}.$$

Доказательство. Имеем

$$\log \frac{Z_{t+1}}{Z_t} = \log \sum_i \frac{\tilde{w}_i^{(t)}}{Z_t} e^{-\eta v_{t,i}} = \log \sum_i w_i^{(t)} e^{-\eta v_{t,i}}.$$

Воспользовавшись неравенством $e^{-a} \leq 1 - a + a^2/2$, справедливым для всех $a \in (0, 1)$, и тем фактом, что $\sum_i w_i^{(t)} = 1$, получаем

$$\begin{aligned} \log \frac{Z_{t+1}}{Z_t} &\leq \log \sum_i w_i^{(t)} (1 - \eta v_{t,i} + \eta^2 v_{t,i}^2/2) \\ &= \log \left(1 - \underbrace{\sum_i w_i^{(t)} (\eta v_{t,i} - \eta^2 v_{t,i}^2/2)}_{\stackrel{\text{def}}{=} b} \right). \end{aligned}$$

Далее заметим, что $b \in (0, 1)$. Поэтому логарифмируя обе части неравенства $1 - b \leq e^{-b}$, мы получаем, что для всех $b \leq 1$ имеет место неравенство $\log(1 - b) \leq -b$, и, значит:

$$\begin{aligned} \log \frac{Z_{t+1}}{Z_t} &\leq -\sum_i w_i^{(t)} (\eta v_{t,i} - \eta^2 v_{t,i}^2/2) \\ &= -\eta \langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle + \eta^2 \sum_i w_i^{(t)} v_{t,i}^2/2 \\ &\leq -\eta \langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle + \eta^2/2. \end{aligned}$$

Суммируя по t, получаем

$$\log(Z_{T+1}) - \log(Z_1) = \sum_{t=1}^{T} \log \frac{Z_{t+1}}{Z_t} \leq -\eta \sum_{t=1}^{T} \langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle + \frac{T\eta^2}{2}. \tag{21.3}$$

Теперь ограничим Z_{T+1} снизу. Для любого i мы можем записать $\tilde{w}_i^{(T+1)} = e^{-\eta \sum_t v_{t,i}}$, и в результате получим

$$\log Z_{T+1} = \log \left(\sum_i e^{-\eta \sum_t v_{t,i}} \right) \geq \log \left(\max_i e^{-\eta \sum_t v_{t,i}} \right) = -\eta \min_i \sum_t v_{t,i}.$$

Объединяя это неравенство с (21.3) и, вспомнив, что $\log(Z_1) = \log(d)$, получаем:

$$-\eta \min_i \sum_t v_{t,i} - \log(d) \leq -\eta \sum_{t=1}^{T} \langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle + \frac{T\eta^2}{2},$$

и после перегруппировки имеем:

$$\sum_{t=1}^{T} \langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle - \min_i \sum_t v_{t,i} - \leq \frac{\log(d)}{\eta} + \frac{\eta T}{2}.$$

Чтобы завершить доказательство, осталось только подставить сюда значение η. □

Доказательство теоремы 21.10

Вооружившись алгоритмом взвешенного большинства и теоремой 21.11, мы готовы доказать теорему 21.10. Начнем с простого случая, когда $\mathcal{H}$ – конечный класс, так что мы можем записать $\mathcal{H} = \{h_1, \ldots, h_d\}$. В этом случае мы можем рассматривать каждую гипотезу h_i как эксперта, который советует предсказать $h_i(\mathbf{x}_t)$, а стоимость его совета равна $v_{t,i} = |h_i(\mathbf{x}_t) - y_t|$. Поэтому алгоритм предсказывает $p_t = \sum_i w_i^{(t)} h_i(\mathbf{x}_t) \in [0, 1]$, и потеря равна

$$|p_t - y_t| = \left| \sum_{i=1}^{d} w_i^{(t)} h_i(\mathbf{x}_t) - y_t \right| = \left| \sum_{i=1}^{d} w_i^{(t)} (h_i(\mathbf{x}_t) - y_t) \right|.$$

Если теперь $y_t = 1$, то для всех i $h_i(\mathbf{x}_t) - y_t \leq 0$. Поэтому приведенное выше выражение сводится к $\sum_i w_i^{(t)} |h_i(\mathbf{x}_t) - y_t|$. Если же $y_t = 0$, то для всех i $h_i(\mathbf{x}_t) - y_t \geq 0$, и приведенное выше выражение также равно $\sum_i w_i^{(t)} |h_i(\mathbf{x}_t) - y_t|$. В итоге мы показали, что

$$|p_t - y_t| = \sum_{i=1}^{d} w_i^{(t)} |h_i(\mathbf{x}_t) - y_t| = \langle \mathbf{w}^{(t)}, \mathbf{v}_t \rangle.$$

Кроме того, для любого i сумма $\sum_t v_{t,i}$ в точности равна числу ошибок, допущенных гипотезой h_i. Применяя теорему 21.11, получаем

Следствие 21.12. *Пусть $\mathcal{H}$ – конечный класс гипотез. Существует алгоритм онлайновой классификации с предсказаниями из диапазона* [0, 1], *для которого верхняя граница сожаления равна*

$$\sum_{t=1}^{T} |p_t - y_t| - \min_{h \in \mathcal{H}} \sum_{t=1}^{T} |h(\mathbf{x}_t) - y_t| \le \sqrt{2 \log(|\mathcal{H}|) T}.$$

Далее рассмотрим случай общего класса гипотез. Выше мы построили эксперта для каждой отдельной гипотезы. Но если класс $\mathcal{H}$ бесконечен, то такая граница бессодержательна. Основная идея состоит в том, чтобы построить множество экспертов более хитроумным способом. Но надо определить это множество так, чтобы, с одной стороны, оно было не слишком большим, а с другой – включало экспертов, дающих точные предсказания.

Мы построим множество экспертов, так что для любой гипотезы $h \in \mathcal{H}$ и любой последовательности образцов $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ существует по меньшей мере один эксперт, который ведет себя в точности как h на всех этих образцах. Мы определим эксперта для любого $L \le \text{Ldim}(\mathcal{H})$ и любой последовательности $1 \le i_1 < i_2 < \ldots < i_L \le T$. Эксперт будет моделировать игру между COA (см. предыдущий раздел) и окружением на последовательности образцов $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ в предположении, что COA допускает ошибки в раундах $i_1, i_2, \ldots, i_L$ и только в них. Эксперт определяется следующим алгоритмом.

Expert (i_1, i_2, ..., i_L)

вход: класс гипотез $\mathcal{H}$; индексы $i_1, i_2, \ldots, i_L$
инициализация: $V_1 = \mathcal{H}$
for $t = 1, 2, \ldots, T$
  получить $\mathbf{x}_t$
  for $r \in \{0, 1\}$ положить $V_t^{(r)} = \{h \in V_t : h(\mathbf{x}_t) = r\}$
  определить $\tilde{y}_t = \text{argmax}_r \text{Ldim}(V_t^{(r)})$
    (в случае неоднозначности положить $\tilde{y}_t = 0$)
  if $t \in \{i_1, i_2, \ldots, i_L\}$
      предсказать $\hat{y}_t = 1 - \tilde{y}_t$
  else
      предсказать $\hat{y}_t = \tilde{y}_t$
  обновить $V_{t+1} = V_t^{(\hat{y}_t)}$

Отметим, что каждый такой эксперт может давать предсказания в каждом раунде t, наблюдая только образцы $\mathbf{x}_1, \ldots, \mathbf{x}_t$. Теперь наш общий алгоритм онлайнового обучения оказывается приложением алгоритма `Weighted-Majority` с этими экспертами.

Чтобы проанализировать алгоритм, сначала отметим, что число экспертов равно

$$d = \sum_{L=0}^{\text{Ldim}(\mathcal{H})} \binom{T}{L}. \tag{21.4}$$

Можно показать, что когда $T \ge \text{Ldim}(\mathcal{H}) + 2$, правая часть этого равенства ограничена сверху выражением $(eT/\text{Ldim}(\mathcal{H}))^{\text{Ldim}(\mathcal{H})}$ (доказательство см. в лемме A.5).

Согласно теореме 21.11, ожидаемое число ошибок алгоритма `Weighted-Majority` не превосходит числа ошибок лучшего эксперта плюс $\sqrt{2 \log(d) T}$. Далее мы

покажем, что число ошибок лучшего эксперта не превосходит числа ошибок лучшей гипотезы в классе $\mathcal{H}$. Следующая ключевая лемма показывает, что на любой последовательности образцов для любой гипотезы $h \in \mathcal{H}$ существует эксперт с точно таким же поведением.

Лемма 21.13. *Пусть $\mathcal{H}$ – произвольный класс гипотез, для которого* $\text{Ldim}(\mathcal{H}) < \infty$. *Пусть $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ – произвольная последовательность образцов. Для любой $h \in \mathcal{H}$ существует $L \leq \text{Ldim}(\mathcal{H})$ и индексы $1 \leq i_1 < i_2 < \ldots < i_L \leq T$ такие, что при выполнении алгоритма* $\text{Expert}(i_1, i_2, \ldots, i_L)$ *на последовательности $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ эксперт предсказывает $h(\mathbf{x}_t)$ в каждом онлайновом раунде с номером $t = 1, 2, \ldots, T$.*

Доказательство. Зафиксируем $h \in \mathcal{H}$ и последовательность $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$. Мы должны построить L и индексы $i_1, i_2, \ldots, i_L$. Рассмотрим выполнение СОА на входной последовательности примеров $(\mathbf{x}_1, h(\mathbf{x}_1)), (\mathbf{x}_2, h(\mathbf{x}_2)), \ldots, (\mathbf{x}_T, h(\mathbf{x}_T))$. СОА делает не более $\text{Ldim}(\mathcal{H})$ ошибок на такой входной последовательности. Положим L равным числу ошибок СОА и определим $\{i_1, i_2, \ldots, i_L\}$ как множество номеров раундов, в которых СОА допускает ошибки.

Теперь рассмотрим выполнение алгоритма $\text{Expert}(i_1, i_2, \ldots, i_L)$ на последовательности $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$. По построению множество V_t, конструируемое алгоритмом $\text{Expert}(i_1, i_2, \ldots, i_L)$, совпадает с множеством V_t, которое СОА конструирует при выполнении на последовательности $(\mathbf{x}_1, h(\mathbf{x}_1)), (\mathbf{x}_2, h(\mathbf{x}_2)), \ldots, (\mathbf{x}_T, h(\mathbf{x}_T))$. Предсказание СОА отличается от предсказаний h тогда и только тогда, когда номер раунда принадлежит множеству $\{i_1, i_2, \ldots, i_L\}$. Поскольку $\text{Expert}(i_1, i_2, \ldots, i_L)$ предсказывает точно так же, как СОА, если t не принадлежит $\{i_1, i_2, \ldots, i_L\}$, и противоположно СОА, если t принадлежит $\{i_1, i_2, \ldots, i_L\}$, то мы заключаем, что предсказания эксперта всегда совпадают с предсказаниями h. □

Эта лемма, в частности, справедлива для гипотезы из $\mathcal{H}$, которая допускает наименьшее количество ошибок на заданной последовательности примеров, поэтому мы приходим к такому следствию.

Следствие 21.14. *Пусть $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T)$ – некоторая последовательность примеров и пусть $\mathcal{H}$ – класс гипотез, для которого* $\text{Ldim}(\mathcal{H}) < \infty$. *Существует $L \leq \text{Ldim}(\mathcal{H})$ и индексы $1 \leq i_1 < i_2 < \ldots < i_L \leq T$ такие, что алгоритм* $\text{Expert}(i_1, i_2, \ldots, i_L)$ *допускает не больше ошибок, чем наилучшая гипотеза $h \in \mathcal{H}$, т. е.*

$$\min_{h \in \mathcal{H}} \sum_{t=1}^{T} |h(\mathbf{x}_t) - y_t|$$

ошибок на данной последовательности примеров.

Итак, принимая во внимание теорему 21.11, мы доказали верхнюю границу в теореме 21.10.

21.3. Онлайновая выпуклая оптимизация

В главе 12 мы изучали выпуклые проблемы обучения и доказали ряд результатов об обучаемости для агностической модели PAC-обучения. В этом разделе мы покажем, что аналогичные результаты имеют место для выпуклых проблем в онлайновом обучении. В частности, мы рассмотрим следующую проблему.

Онлайновая выпуклая оптимизация

определения:
класс гипотез $\mathcal{H}$; множество Z; функция потерь $\ell: \mathcal{H} \times Z \to \mathbb{R}$
предположения:
множество $\mathcal{H}$ выпуклое
$\forall z \in Z$, $(\cdot, z)$ – выпуклая функция
for $t = 1, 2, \ldots, T$
обучаемый предсказывает вектор $\mathbf{w}^{(t)} \in \mathcal{H}$
окружение в ответ сообщает $z_t \in Z$
потеря обучаемого составляет $\ell(\mathbf{w}^{(t)}, z_t)$

Как и в проблеме онлайновой классификации, мы анализируем *сожаление* алгоритма. Напомним, что сожаление онлайнового алгоритма относительно конкурирующей гипотезы, в роли которой в данном случае выступает некоторый вектор $\mathbf{w}^* \in \mathcal{H}$, определяется следующим образом:

$$\text{Regret}_A(\mathbf{w}^*, T) = \sum_{t=1}^{T} \ell(\mathbf{w}^{(t)}, z_t) - \sum_{t=1}^{T} \ell(\mathbf{w}^*, z_t). \tag{21.5}$$

Как и раньше, сожаление алгоритма относительно множества конкурирующих векторов $\mathcal{H}$ определяется как

$$\text{Regret}_A(\mathcal{H}, T) = \sup_{\mathbf{w}^* \in \mathcal{H}} \text{Regret}_A(\mathbf{w}^*, T).$$

В главе 14 мы показали, что алгоритм стохастического градиентного спуска решает выпуклые проблемы обучения в агностической модели PAC. Теперь мы покажем, что очень похожий алгоритм онлайнового градиентного спуска решает онлайновые выпуклые проблемы обучения.

Онлайновый градиентный спуск

параметр: $\eta > 0$
инициализация: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \ldots, T$
предсказать $\mathbf{w}^{(t)}$
получить z_t и положить $f_t(\cdot) = \ell(\cdot, z_t)$
выбрать $\mathbf{v}_t \in \partial f_t(\mathbf{w}^{(t)})$
обновить:
1. $\mathbf{w}^{(t+1/2)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
2. $\mathbf{w}^{(t+1)} = \text{argmin}_{\mathbf{w} \in \mathcal{H}} \|\mathbf{w} - \mathbf{w}^{(t+1/2)}\|$

Теорема 21.15. *Для алгоритма онлайнового градиентного спуска имеет место следующая верхняя граница сожаления для любого* $\mathbf{w}^* \in \mathcal{H}$:

$$\text{Regret}_A(\mathbf{w}^*, T) \leq \frac{\|\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\mathbf{v}_t\|^2.$$

Если дополнительно предположить, что функция f_t ρ-липшицева для любого t, то при $\eta = 1/\sqrt{T}$ справедлива оценка

$$\text{Regret}_A(\mathbf{w}^*, T) \le \frac{1}{2}(\|\mathbf{w}^*\|^2 + \rho^2)\sqrt{T}.$$

Если дополнительно предположить, что класс $\mathcal{H}$ является B-ограниченным, то при $\eta = B/(\rho\sqrt{T})$ справедлива оценка

$$\text{Regret}_A(\mathcal{H}, T) \le B\rho\sqrt{T}.$$

Доказательство. Анализ проводится так же, как для стохастического градиентного спуска, с использованием проекций. В силу леммы о проецировании, определения $\mathbf{w}^{(t+1/2)}$ и определения субградиента, имеем для любого t

$$\begin{aligned}
&\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 \\
&\quad = \|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t+1/2)} - \mathbf{w}^*\|^2 + \|\mathbf{w}^{(t+1/2)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 \\
&\quad \le \|\mathbf{w}^{(t+1/2)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 \\
&\quad = \|\mathbf{w}^{(t)} - \eta\mathbf{v}_t - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 \\
&\quad = -2\eta\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t\rangle + \eta^2\|\mathbf{v}_t\|^2 \\
&\quad \le -2\eta(f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*)) + \eta^2\|\mathbf{v}_t\|^2.
\end{aligned}$$

Суммируя по t и замечая, что в сумме в левой части все члены, кроме крайних, сокращаются, получаем:

$$\|\mathbf{w}^{(T+1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(1)} - \mathbf{w}^*\|^2 \le -2\eta\sum_{t=1}^{T}(f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*)) + \eta^2\sum_{t=1}^{T}\|\mathbf{v}_t\|^2.$$

Перегруппируя члены и пользуясь тем фактом, что $\mathbf{w}^{(1)} = \mathbf{0}$, получаем:

$$\begin{aligned}
\sum_{t=1}^{T}(f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*)) &\le \frac{\|\mathbf{w}^{(1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(T+1)} - \mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2 \\
&\le \frac{\|\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2.
\end{aligned}$$

Тем самым доказана первая граница. Вторая вытекает из предположения о ρ-липшицевости f_t, откуда следует, что $\|\mathbf{v}_t\| \le \rho$. □

21.4. Алгоритм онлайнового перцептрона

Перцептрон – классический алгоритм онлайнового обучения для бинарной классификации с классом однородных полупространств в качестве гипотез, т. е. $\mathcal{H} = \{\mathbf{x} \mapsto \text{sign}(\langle \mathbf{w}, \mathbf{x}\rangle) : \mathbf{w} \in \mathbb{R}^d\}$. В разделе 9.1.2 мы описали пакетный вариант перцептрона, предназначенный для решения проблемы ERM относительно $\mathcal{H}$. Теперь представим онлайновый вариант этого алгоритма.

Пусть $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{-1, 1\}$. В раунде t обучаемый получает вектор $\mathbf{x}_t \in \mathbb{R}^d$. Обучаемый хранит вектор весов $\mathbf{w}^{(t)} \in \mathbb{R}^d$ и предсказывает $p_t = \text{sign}(\langle \mathbf{w}^{(t)}, \mathbf{x}_t\rangle)$. Затем он получает метку $y_t \in \mathcal{Y}$ и несет потерю 1, если $p_t \ne y_t$, и 0 в противном случае.

Цель обучаемого – допустить как можно меньше ошибок предсказания. В разделе 21.1 мы охарактеризовали оптимальный алгоритм и показали, что наилучшая достижимая граница ошибок зависит от размерности Литлстоуна класса. Ниже мы покажем, что если $d \geq 2$, то $\mathrm{Ldim}(\mathcal{H}) = \infty$, откуда следует, что нет никакой надежды допустить малое число ошибок предсказания. Действительно, рассмотрим дерево, в котором $\mathbf{v}_1 = (1/2, 1, 0, \ldots, 0)$, $\mathbf{v}_2 = (1/4, 1, 0, \ldots, 0)$, $\mathbf{v}_3 = (3/4, 1, 0, \ldots, 0)$ и т. д. В силу плотности множества вещественных чисел это дерево расколото подмножеством $\mathcal{H}$, которое содержит все гипотезы, параметризованные вектором $\mathbf{w}$ вида $\mathbf{w} = (-1, a, 0, \ldots, 0)$ для $a \in [0, 1]$. Отсюда мы делаем вывод, что действительно $\mathrm{Ldim}(\mathcal{H}) = \infty$.

Чтобы обойти этот результат о невозможности, алгоритм перцептрона полагается на технику *суррогатной выпуклой потери* (см. раздел 12.3). Она также тесно связана с понятием *зазора*, которое мы изучали в главе 15.

Вектор весов $\mathbf{w}$ допускает ошибку на примере $(\mathbf{x}, y)$, если знак $\langle \mathbf{w}, \mathbf{x} \rangle$ не совпадает с y. Поэтому мы можем записать бинарную функцию потерь в виде

$$\ell(\mathbf{w}, (\mathbf{x}, y)) = \mathbb{1}_{[y\langle \mathbf{w}, \mathbf{x} \rangle \leq 0]}.$$

В тех раундах, в которых алгоритм допускает ошибку предсказания, мы будем использовать кусочно-линейную потерю в качестве суррогатной выпуклой функции потерь:

$$f_t(\mathbf{w}) = \max\{0, 1 - y_t \langle \mathbf{w}, \mathbf{x}_t \rangle\}.$$

Кусочно-линейная потеря удовлетворяет двум условиям:

- f_t – выпуклая функция;
- для любого $\mathbf{w}$ $f_t(\mathbf{w}) \geq \ell(\mathbf{w}, (\mathbf{x}_t, y_t))$. В частности это справедливо для $\mathbf{w}^{(t)}$.

В тех раундах, в которых алгоритм предсказывает правильно, мы определим $ft(\mathbf{w}) = 0$. Очевидно, что и в этом случае f_t выпукла. Кроме того, $f_t(\mathbf{w}^{(t)}) = \ell(\mathbf{w}^{(t)}, (\mathbf{x}_t, y_t)) = 0$.

Замечание 21.5. В разделе 12.3 мы использовали одну и ту же суррогатную функцию потерь для всех примеров. В онлайновой модели суррогатная функция может зависеть от раунда и даже от $\mathbf{w}^{(t)}$. Разрешение использовать зависящие от раунда суррогаты проистекает из анализа худшего случая, который мы применяем в онлайновом обучении.

Теперь выполним алгоритм онлайнового градиентного спуска на последовательности функций $f_1, \ldots, f_T$, когда классом гипотез является множество всех векторов $\mathbb{R}^d$ (и, следовательно, шаг проецирования не имеет смысла). Напомним, что алгоритм инициализирует $\mathbf{w}^{(1)} = \mathbf{0}$, а правило обновления имеет вид

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$$

для некоторого $\mathbf{v}_t \in \partial f_t(\mathbf{w}^{(t)})$. В нашем случае, если $y_t \langle \mathbf{w}^{(t)}, \mathbf{x}_t \rangle > 0$, то f_t – нулевая функция, и мы можем взять $\mathbf{v}_t = \mathbf{0}$. В противном случае легко проверить, что $\mathbf{v}_t = -y_t \mathbf{x}_t$ принадлежит $\partial f_t(\mathbf{w}^{(t)})$. Таким образом, мы получаем правило обновления

$$\mathbf{w}^{(t+1)} = \begin{cases} \mathbf{w}^{(t)}, & \text{если } y_t \langle \mathbf{w}^{(t)}, \mathbf{x}_t \rangle > 0 \\ \mathbf{w}^{(t)} + h y_t \mathbf{x}_t, & \text{в противном случае} \end{cases}.$$

Обозначим $\mathcal{M}$ множество раундов, в которых $\text{sign}(\langle \mathbf{w}^{(t)}, \mathbf{x}_t \rangle) \neq y_t$. Отметим, что предсказание перцептрона в t-м раунде можно переписать в виде

$$p_t = \text{sign}(\langle \mathbf{w}^{(t)}, \mathbf{x}_t \rangle) = \text{sign}\left(h \sum_{i \in \mathcal{M}: i<t} y_i \langle \mathbf{x}_i, \mathbf{x}_t \rangle \right).$$

Отсюда следует, что предсказания перцептрона и множество $\mathcal{M}$ не зависят от значения η при условии, что $\eta > 0$. Таким образом, мы получили алгоритм перцептрона:

Перцептрон

инициализация: $\mathbf{w}_1 = \mathbf{0}$
for $t = 1, 2, \ldots, T$
  получить $\mathbf{x}_t$
  предсказать $p_t = \text{sign}(\langle \mathbf{w}^{(t)}, \mathbf{x}_t \rangle)$
  if $y_t \langle \mathbf{w}^{(t)}, \mathbf{x}_t \rangle \leq 0$
    $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + y_t \mathbf{x}_t$
  else
    $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)}$

При анализе перцептрона мы будем опираться на анализ онлайнового градиентного спуска из предыдущего раздела. В нашем случае субградиент функции f_t, используемой в перцептроне, равен $\mathbf{v}_t = -\mathbb{1}_{[y_t \langle \mathbf{w}^{(t)}, \mathbf{x}_t \rangle \leq 0]} y_t \mathbf{x}_t$. Действительно, правило обновления в перцептроне имеет вид $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \mathbf{v}_t$ и, как уже было отмечено, оно эквивалентно $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$ для любого $\eta > 0$. Поэтому согласно теореме 21.15

$$\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} f_t(\mathbf{w}^*) \leq \frac{1}{2\eta} \|\mathbf{w}^*\|_2^2 + \frac{h}{2} \sum_{t=1}^{T} \|\mathbf{v}_t\|_2^2.$$

Поскольку $f_t(\mathbf{w}^{(t)})$ – суррогат для бинарной потери, мы знаем, что $\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) \geq |\mathcal{M}|$. Обозначим $R = \max_t \|\mathbf{x}_t\|$, тогда

$$|\mathcal{M}| - \sum_{t=1}^{T} f_t(\mathbf{w}^*) \leq \frac{1}{2\eta} \|\mathbf{w}^*\|_2^2 + \frac{\eta}{2} \sum_{t=1}^{T} |\mathcal{M}| R^2.$$

Полагая $\eta = \dfrac{\|\mathbf{w}^*\|}{R\sqrt{|\mathcal{M}|}}$ и перегруппировывая члены, получаем

$$|\mathcal{M}| - R\|\mathbf{w}^*\| \sqrt{|\mathcal{M}|} - \sum_{t=1}^{T} f_t(\mathbf{w}^*) \leq 0. \tag{21.6}$$

Из этого неравенства следует

Теорема 21.16. *Предположим, что алгоритм перцептрона выполняется на последовательности примеров* $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$, *и пусть* $R = \max_t \|\mathbf{x}_t\|$. *Обозначим* $\mathcal{M}$ *множество раундов, в которых перцептрон допускает ошибку, и положим* $f_t(\mathbf{w}) = \mathbb{1}_{[t \in \mathcal{M}]}[1 - y_t \langle \mathbf{w}, \mathbf{x}_t \rangle]_+$. *Тогда для любого* $\mathbf{w}^*$

$$|\mathcal{M}| \leq \sum_{t=1} f_t(\mathbf{w}^*) + R\|\mathbf{w}^*\| \sqrt{\sum_t f_t(\mathbf{w}^*)} + R^2\|\mathbf{w}^*\|^2.$$

В частности, если существует $\mathbf{w}^*$ *такой, что* $y_t\langle \mathbf{w}, \mathbf{x}_t\rangle \geq 1$ *для всех* t*, то*

$$|\mathcal{M}| \leq R^2\|\mathbf{w}^*\|^2.$$

Доказательство. Теорема вытекает из неравенства (21.6) и следующего утверждения: если $x, b, c \in \mathbb{R}_+$, то из неравенства $x - b\sqrt{x} - c \leq 0$ следует, что $x \leq c + b^2 + b\sqrt{c}$. Это утверждение легко доказать, проанализировав корни выпуклой параболы $Q(y) = y_2 - by - c$. □

Последнее допущение теоремы 21.16 называется *разделимостью с большим зазором* (см. главу 15). Оно означает, что существует вектор $\mathbf{w}^*$, который не только удовлетворяет условию, что точка $\mathbf{x}_t$ находится по правильную сторону полупространства, но и гарантирует, что $\mathbf{x}_t$ расположена не слишком близко к решающей границе. Точнее, расстояние от $\mathbf{x}_t$ до решающей границы не меньше $\gamma = 1/\|\mathbf{w}^*\|$, и тогда верхняя граница принимает вид $(R/\gamma)^2$.

Если предположение о разделимости не выполнено, то верхняя граница включает член $[1 - y_t\langle \mathbf{w}^*, \mathbf{x}_t\rangle]_+$, который измеряет, насколько сильно нарушено требование разделимости с зазором.

И напоследок заметим, что возможны случаи, когда существуют гипотезы $\mathbf{w}^*$, не допускающие ни одной ошибки на последовательности, но перцептрон все равно делает много ошибок. Это прямое следствие того факта, что $\text{Ldim}(\mathcal{H}) = \infty$. Чтобы обойти этот результат о невозможности, мы должны сделать дополнительные предположения о последовательности примеров – верхняя граница в теореме 21.16 имеет смысл, только если суммарная суррогатная потеря $\sum_t f_t(\mathbf{w}^*)$ не слишком велика.

ЛАНЬ®

21.5. Резюме

В этой главе мы изучили модель онлайнового обучения. Многие результаты, полученные для модели PAC-обучения, имеют аналоги в онлайновой модели. Во-первых, мы показали, что онлайновую обучаемость характеризует комбинаторная размерность Литлстоуна. Для этого мы ввели алгоритм SOA (для реализуемого случая) и алгоритм взвешенного большинства (для нереализуемого случая). Мы также изучили выпуклую онлайновую оптимизацию и показали, что онлайновый градиентный спуск является успешным обучаемым, если функция потерь выпуклая и липшицева. Наконец, мы представили онлайновый алгоритм перцептрона, являющийся комбинацией онлайнового градиентного спуска и понятия суррогатной выпуклой функции потерь.

21.6. Библиографические сведения

Стандартный оптимальный алгоритм был описан в пионерской работе Литлстоуна (1988). Обобщение на нереализуемый случай, а также другие варианты,

например, размерность Литлстоуна, основанная на зазоре, приведены в работе (Ben-David et al., 2009). Характеристики онлайновой обучаемости в проблемах, выходящих за рамки классификации, получены в работах (Abernethy, Bartlett, Rakhlin & Tewari, 2008; Rakhlin, Sridharan & Tewari, 2010; Daniely et al., 2011). Алгоритм взвешенного большинства описан в работах Littlestone & Warmuth, 1994, и Vovk, 1990.

Термин «онлайновое выпуклое программирование» был введен в работе Zinkevich (2003), но сама постановка задачи описана несколькими годами раньше в работе (1999). Идея перцептрона восходит к работе Розенблатта (Rosenblatt, 1958). Анализ для реализуемого случая (с предположениями о зазоре) приведен в работах (Agmon, 1954; Minsky & Papert, 1969). В работе Freund and Schapire (Freund & Schapire, 1999) представлен анализ для нереализуемого случая с квадратично-линейной потерей, основанный на сведении к реализуемому случаю. Прямой анализ для нереализуемого случая с кусочно-линейной потерей дан в работе Джентайла (Gentile, 2003).

За дополнительной информацией отсылаем читателя к работам Cesa-Bianchi and Lugosi (2006) and Shalev-Shwartz (2011).

21.7. Упражнения

21.1. Найдите класс гипотез $\mathcal{H}$ и последовательность примеров, на которой алгоритм `Consistent` делает $|\mathcal{H}| - 1$ ошибок.

21.2. Найдите класс гипотез $\mathcal{H}$ и последовательность примеров, на которой граница ошибок в алгоритме `Halving` строгая.

21.3. Пусть $d \geq 2$, $\mathcal{X} = \{1, \ldots, d\}$ и $\mathcal{H} = \{h_j : j \in [d]\}$, где $h_j(x) = 1_{[x=j]}$. Вычислите $M_{\text{Halving}}(\mathcal{H})$ (т. е. выведите нижнюю и верхнюю границы $M_{\text{Halving}}(\mathcal{H})$ и докажите, что они равны).

21.4. Трюк удвоения:

В теореме 21.15 параметр η зависит от периода времени T. В этом упражнении мы покажем, как избавиться от этой зависимости с помощью простого трюка.

Рассмотрим алгоритм, для которого верхняя граница сожаления имеет вид $\alpha\sqrt{T}$, но параметры требуют знания T. Описанный ниже трюк удвоения позволяет преобразовать этот алгоритм в другой, не нуждающийся в знании о периоде времени. Идея в том, чтобы разделить время на отрезки возрастающего размера и выполнять исходный алгоритм на каждом отрезке.

Трюк удвоения

вход: алгоритм A, параметры которого зависят от периода времени
for $m = 0, 1, 2, \ldots$
  выполнить A в 2^m раундах $t = 2^m, \ldots, 2^{m+1} - 1$

Покажите, что если сожаление A в каждом отрезке из 2^m раундов не превосходит $\alpha\sqrt{2^m}$, то полное сожаление не превосходит

$$\frac{\sqrt{2}}{\sqrt{2}-1}\alpha\sqrt{T}.$$

21.5. Преобразование онлайнового алгоритма в пакетный. В этом упражнении мы продемонстрируем, как можно использовать успешный онлайновый алгоритм обучения, чтобы вывести успешного PAC-обучаемого.

Рассмотрим проблему PAC-обучения для бинарной классификации, параметризованную множеством образцов $\mathcal{X}$ и классом гипотез $\mathcal{H}$. Предположим, что существует онлайновый алгоритм обучения A с границей ошибок $M_A(\mathcal{H}) < \infty$. Рассмотрим выполнение этого алгоритма на последовательности T примеров, независимо выбранных из одного и того же распределения $\mathcal{D}$ на множестве образцов $\mathcal{X}$ и помеченных некоторой гипотезой $h^* \in \mathcal{H}$. Предположим, что для каждого раунда t предсказание этого алгоритма основано на гипотезе $h_t : \mathcal{X} \to \{0, 1\}$. Покажите, что

$$\mathbb{E}[L_{\mathcal{D}}(h_r) \leq \frac{M_A(\mathcal{H})}{T},$$

где математическое ожидание берется по случайной выборке образцов и по случайному выбору r из равномерного распределения на $[T]$.

Указание. Рассуждайте так же, как при доказательстве теоремы 14.8.

ЛАНЬ®

Глава 22

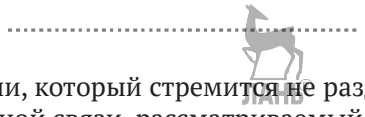
КЛАСТЕРИЗАЦИЯ

Кластеризация – один из самых распространенных методов исследовательского анализа данных. Во всех научных дисциплинах, от общественных наук до биологии, ученые стремятся сначала получить интуитивное представление о данных, выявив среди них значимые группы. Например, в вычислительной биологии кластеризуют гены на основе сходства их экспрессии в различных экспериментах; в розничной торговле строят кластеры покупателей на основе их профилей с целью таргетирования маркетинговых усилий; астрономы кластеризуют звезды на основе их пространственной близости.

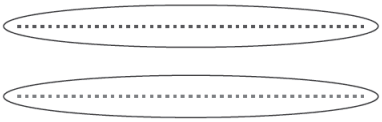
Естественно, прежде всего необходимо уточнить, что такое кластеризация. Интуитивно понятно, что задача кластеризации – разбить объекты на группы, так чтобы похожие оказались в одной группе, а непохожие – в разных. Ясно, что это описание очень неточно и, возможно, неоднозначно. Но, как ни странно, совершенно неясно, как должно выглядеть более строгое определение.

У этого затруднения есть несколько причин. Основная проблема состоит в том, что две цели, упомянутые в высказанном ранее предложении, во многих случаях противоречат друг другу. С точки зрения математики, схожесть (или близость) – нетранзитивное отношение, а нахождение в одном кластере – отношение эквивалентности, которое, в частности, обладает свойством транзитивности. Точнее, может существовать длинная последовательность объектов x_1, ..., x_m такая, что каждый элемент x_i очень похож на своих соседей x_{i-1} и x_{i+1}, но x_1 и x_m совершенно непохожи. Если мы хотим гарантировать, что любые два похожих элемента попадают в один кластер, то должны помещать в один и тот же кластер все элементы последовательности. Но в таком случае в одном кластере окажутся непохожие элементы (x_1 и x_m), и, стало быть, будет нарушено второе требование.

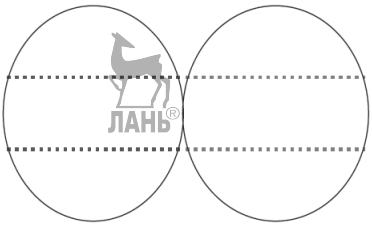
Чтобы наглядно проиллюстрировать этот момент, предположим, что требуется отнести к двум кластерам точки, изображенные на следующем рисунке.



Алгоритм кластеризации, который стремится не разделять близкие точки (например, алгоритм одиночной связи, рассматриваемый в разделе 22.1), распределит такие входные данные по двум горизонтально расположенным кластерам:

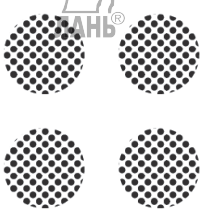


С другой стороны, метод, который стремится в первую очередь не включать удаленные точки в один кластер (например, алгоритм 2-средних, описанный в разделе 22.1), кластеризует те же данные, разделив их по вертикали на левую и правую половину.

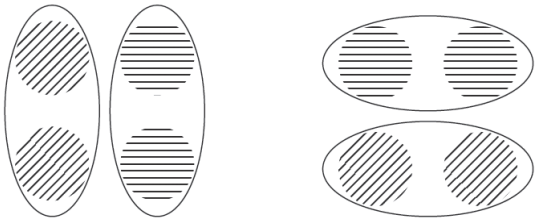


Еще одна проблема – отсутствие «объективной истины», характерное как для кластеризации, так и для прочих методов *обучения без учителя*. До сих пор в этой книге мы занимались в основном *обучением с учителем* (например, проблемой обучения классификатора на наборе помеченных обучающих данных). Цель обучения с учителем понятна – мы хотим обучить классификатор, который будет как можно вернее предсказывать метки будущих примеров. Кроме того, такой обучаемый может оценить успех или риск своих гипотез, вычисляя эмпирическую потерю на помеченных примерах. Но кластеризация – проблема обучения без учителя, т. е. никаких предсказываемых меток не существует. Вместо этого мы хотим организовать данные некоторым осмысленным способом. Поэтому никакой очевидной процедуры оценки успеха кластеризации нет. Даже если мы знаем всё об истинном распределении данных, непонятно, что называть «правильной» кластеризацией и как оценивать предложенную кластеризацию.

Рассмотрим, к примеру, следующее множество точек в $\mathbb{R}^2$:



и предположим, что требуется распределить их по двум кластерам. Налицо решения, имеющие одинаковое право на существование:



И это не искусственный феномен, а ситуация, встречающаяся в реальных приложениях. Заданное множество объектов можно кластеризовать различными осмысленными способами. Это может быть связано с наличием различных определений расстояния (сходства) между объектами. Например, записи речи можно кластеризовать по акценту говорящего или по содержанию, рецензии на фильмы – по тематике фильма или по эмоциональной окраске, картины – по сюжету или по стилю. И так далее.

Короче говоря, возможны различные способы кластеризации предложенного набора данных. И потому есть широкий спектр алгоритмов кластеризации, которые распределяют одни и те же данные по кластерам совершенно различно.

Модель кластеризации

Задачи кластеризации могут варьироваться как по типу входных данных, так и по типу результата, которого от них ожидают. Для определенности будем рассматривать следующую типичную постановку задачи.

- **Вход** – множество элементов $\mathcal{X}$ и определенная на нем функция расстояния, т. е. функция $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, которая симметрична, обладает свойством $d(x, x) = 0$ для всех $x \in \mathcal{X}$ и часто удовлетворяет еще неравенству треугольника. Или это может быть функция сходства $s : \mathcal{X} \times \mathcal{X} \to [0, 1]$, которая симметрична и обладает свойством $s(x, x) = 1$ для всех $x \in \mathcal{X}$. Кроме того, некоторые алгоритмы кластеризации требуют, чтобы был задан входной параметр k (определяющий требуемое количество кластеров).
- **Выход** – разбиение области определения $\mathcal{X}$ на подмножества, т. е. $C = (C_1, \ldots, C_k)$, где $\bigcup_{i=1}^{k} C_i = \mathcal{X}$ и для всех $i \neq j$ $C_i \cap C_j = \emptyset$. Иногда кластеризация бывает «мягкой», когда разбиение $\mathcal{X}$ на кластеры вероятностное, т. е. результатом является функция, которая сопоставляет каждой точке $x \in \mathcal{X}$ вектор $(p_1(x), \ldots, p_k(x))$, где $p_i(x) = \mathrm{P}[x \in C_i]$ – вероятность того, что x принадлежит кластеру C_i. Еще один возможный результат – дендрограмма (от греч. dendron – дерево, gramma – рисование), т. е. иерархически организованное дерево подмножеств, в листьях которого находятся одноэлементные множества, а в корне – вся область образцов. Позже мы обсудим эту постановку более подробно.

Ниже приведен обзор наиболее популярных методов кластеризации. В последнем разделе этой главы мы вернемся к общему обсуждению предмета кластеризации.

22.1. Алгоритмы кластеризации на основе связи

Кластеризация на основе связи – пожалуй, самая простая и прямолинейная парадигма кластеризации. Такие алгоритмы выполняют последовательность раундов. Вначале производится тривиальная кластеризация, когда каждый кластер состоит ровно из одной точки. Затем в цикле производится объединение «ближайших» кластеров, получившихся в предыдущем раунде. Таким образом, количество кластеров в каждом раунде уменьшается. Если продолжать этот процесс до логического завершения, то в итоге останется один большой кластер, содержащий все точки. Поэтому для точного определения такого алгоритма необходимо два параметра. Во-первых, мы должны определить, как измерять расстояние между кластерами, а во-вторых, когда прекращать объединение. Напомним, что входом алгоритма кластеризации является функция расстояния d. Существует много способов обобщить d для измерения расстояния между подмножествами (кластерами). Перечислим самые популярные.

1. Кластеризация методом одиночной связи, когда расстояние между кластерами определяется как минимальное расстояние между их членами, т. е.

$$D(A, B) \stackrel{\text{def}}{=} \min\{d(x, y) : x \in A, y \in B\}.$$

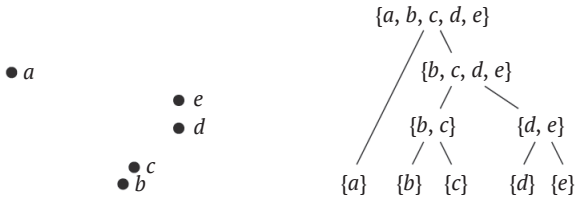
2. Кластеризация методом средней связи, когда расстояние между кластерами определяется как среднее расстояние между всеми парами точек, взятых по одной из каждого кластера:

$$D(A,B) \stackrel{\text{def}}{=} \frac{1}{|A||B|} \sum_{x \in A, y \in B} d(x,y).$$

3. Кластеризация методом максимальной связи, когда расстояние между кластерами определяется как максимальное расстояние между их членами:

$$D(A, B) \stackrel{\text{def}}{=} \max\{d(x, y) : x \in A, y \in B\}.$$

Алгоритмы кластеризации на основе связи являются *агломеративными* в том смысле, что в начале работы данные полностью фрагментированы, и алгоритм объединяет их во все более крупные кластеры. В отсутствие правила остановки результат такого алгоритма можно описать в виде *дендрограммы* – дерева подмножеств области образцов, в листьях которого находятся одиночные точки, а в корне – вся область образцов. Например, если на входе было множество $\mathcal{X} = \{a, b, c, d, e\} \subset \mathbb{R}^2$ с евклидовым расстоянием (рисунок слева), то в результате работы будет построена дендрограмма, изображенная на рисунке справа.



Алгоритм одиночной связи тесно связан с алгоритмом Краскала нахождения минимального остовного дерева взвешенного графа. Действительно, рассмотрим полный граф, вершинами которого являются элементы $\mathcal{X}$, а ребру (x, y) назначен вес, равный расстоянию $d(x, y)$. Каждая операция объединения двух кластеров, выполненная алгоритмом одиночной связи, соответствует выбору ребра вышеупомянутого графа. Можно также показать, что множество ребер, выбираемых этим алгоритмом, образует минимальное остовное дерево.

Если мы хотим превратить дендрограмму в разбиение пространства (кластеризацию), то должны задать *критерий остановки*. Чаще всего употребляются следующие критерии:

- фиксированное число кластеров – задается параметр k и объединение прекращается, когда число кластеров станет равно k;
- верхняя граница расстояния – задается число $r \in \mathbb{R}_+$. Объединение прекращается, когда все расстояния между кластерами окажутся больше r. Можно также положить r равным $\alpha \max\{d(x, y) : x, y \in \mathcal{X}\}$ для некоторого $\alpha < 1$. Такой критерий остановки называется «верхней границей масштабированного расстояния».

22.2. Метод k-средних и другие методы кластеризации на основе минимизации стоимости

Еще один распространенный подход к кластеризации состоит в том, чтобы вначале определить функцию стоимости на параметризованном множестве возможных вариантов кластеризации; тогда цель алгоритма – найти разбиение (кластеризацию) с минимальной стоимостью. При такой парадигме задача кластеризации становится задачей оптимизации. Целевая функция отображает пару $(\mathcal{X}, d)$ и предложенное решение $C = (C_1, \ldots, C_k)$ в вещественное число. Если задана такая целевая функция, которую мы обозначим G, то *цель* алгоритма кластеризации – найти для данной пары $(\mathcal{X}, d)$ кластеризацию C, доставляющую минимум $G((X, d), C)$. Для достижения этой цели необходимо применить некий алгоритм поиска.

Оказывается, что большинство получающихся задач оптимизации являются NP-трудными, а некоторые даже аппроксимировать NP-трудно. Стало быть, когда говорят, например, о кластеризации методом k-средних, чаще всего имеют в виду некий конкретный алгоритм аппроксимации, а не определенную функцию стоимости и не точное решение соответствующей задачи минимизации.

Многие распространенные целевые функции требуют задания количества кластеров k в качестве параметра. На практике наиболее подходящее для данной задачи значение k обычно задает пользователь алгоритма.

Ниже мы опишем некоторые наиболее распространенные целевые функции.

- **Целевая функция *k*-средних** – одна из самых популярных. В этом случае данные разбиваются на непересекающиеся множества $C_1, \ldots, C_k$, где каждое C_i представлено центроидом μ_i. Предполагается, что множество входов $\mathcal{X}$ погружено в некоторое метрическое пространство $(\mathcal{X}', d)$ (т. е. $\mathcal{X} \subseteq \mathcal{X}'$) и что

центроиды являются элементами $\mathcal{X}'$. Целевая функция *k*-средних измеряет квадрат расстояния от каждой точки $\mathcal{X}$ до центроида кластера. Центроид C_i определяется как

$$\mu_i(C_i) = \arg\min_{\mu \in \mathcal{X}'} \sum_{x \in C_i} d(x, \mu)^2.$$

Тогда целевая функция *k*-средних имеет вид

$$G_{\text{k-means}}((\mathcal{X}, d), (C_1, \ldots, C_k)) = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i(C_i))^2.$$

Это выражение можно переписать также в виде

$$G_{\text{k-means}}((\mathcal{X}, d), (C_1, \ldots, C_k)) = \min_{\mu_1, \ldots, \mu_k \in \mathcal{X}'} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i)^2. \quad (22.1)$$

Такая функция полезна, например, в задачах цифровой связи, когда $\mathcal{X}$ можно рассматривать как множество подлежащих передаче сигналов. При этом $\mathcal{X}$ может быть очень большим множеством вещественных векторов, а цифровая аппаратура позволяет передавать только конечное число бит для каждого сигнала. Один из способов добиться хорошего качества передачи при таких ограничениях – представить каждый элемент $\mathcal{X}$ «близким» элементом некоторого конечного множества $\mu_1, \ldots, \mu_k$ и заменить передачу любого $x \in \mathcal{X}$ передачей индекса ближайшего μ_i. Целевую функцию *k*-средних можно интерпретировать как меру искажения передачи, вносимого такой схемой представления данных.

- **Целевая функция *k*-медоидов** похожа на функцию *k*-средних, но дополнительно требуется, что центроиды кластеров были элементами самого множества входов. Целевая функция определяется как

$$G_{\text{K-medoid}}((\mathcal{X}, d), (C_1, \ldots, C_k)) = \min_{\mu_1, \ldots, \mu_k \in \mathcal{X}} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i)^2.$$

- **Целевая функция *k*-медиан** очень похожа на *k*-медоидов, но «искажение», обусловленное заменой данных центроидом кластера, измеряется как расстояние, а не как квадрат расстояния.

$$G_{\text{K-median}}((\mathcal{X}, d), (C_1, \ldots, C_k)) = \min_{\mu_1, \ldots, \mu_k \in \mathcal{X}} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i).$$

Такая целевая функция имеет смысл в задаче о *размещении объектов*. Рассмотрим задачу о размещении *k* пожарных команд в городе. Можно рассматривать здания как элементы данных, а наша цель – разместить команды, так чтобы минимизировать среднее расстояние между зданием и ближайшей пожарной командой.

Все приведенные выше примеры можно рассматривать как *центральные* целевые функции. Решение такой проблемы кластеризации определяется множе-

ством центров кластеров, и алгоритм сопоставляет каждому образцу ближайший к нему центр. Вообще, центральная целевая функция задается выбором некоторой монотонной функции $f : \mathbb{R}_+ \to \mathbb{R}_+$, после чего определяется

$$G_f((\mathcal{X}, d), (C_1, \ldots, C_k)) = \min_{\mu_1, \ldots, \mu_k \in \mathcal{X}'} \sum_{i=1}^{k} \sum_{x \in C_i} f(d(x, \mu_i)),$$

где $\mathcal{X}'$ – либо $\mathcal{X}$, либо надмножество $\mathcal{X}$.

Но встречаются и нецентральные целевые функции, например: *сумма внутрикластерных расстояний* (sum of in-cluster distances – SOD)

$$G_{\text{SOD}}((\mathcal{X}, d), (C_1, \ldots, C_k)) = \sum_{i=1}^{k} \sum_{x,y \in C_i} d(x, y),$$

и целевая функция MinCut, которую мы обсудим в разделе 22.3.

22.2.1. Алгоритм k-средних

Целевая функция k-средних очень популярна в практических применениях кластеризации. Но обычно нахождение для нее оптимального решения – вычислительно неразрешимая задача (она является NP-трудной и даже аппроксимировать ее NP-трудно). В качестве альтернативы часто используют описанный ниже итеративный алгоритм – настолько часто, что сам термин «кластеризация методом k-средних» стал означать результат этого алгоритма, а не кластеризацию, которая минимизирует целевую функцию k-средних. Мы опишем алгоритм, подразумевая евклидову метрику $d(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||$.

Алгоритм k-средних

вход: $\mathcal{X} \subset \mathbb{R}^n$; количество кластеров k
инициализация: случайным образом выбрать начальные центроиды $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$
повторять, пока не сойдется
  $\forall i \in [k]$ положить $C_i = \{\mathbf{x} \in \mathcal{X} : i = \text{argmin}_j ||\mathbf{x} - \boldsymbol{\mu}_j||\}$
  (неоднозначность разрешается произвольно)
  $\forall i \in [k]$ обновить $\boldsymbol{\mu}_i = (1/|C_i|) \sum_{\mathbf{x} \in C_i} \mathbf{x}$

Лемма 22.1. *Никакая итерация алгоритма k-средних не увеличивает целевой функции k-средних (определенной в (22.1)).*

Доказательство. Чтобы упростить обозначения, будем использовать сокращенную запись целевой функции k-средних $G(C_1, \ldots, C_k)$:

$$G(C_1, \ldots, C_k) = \min_{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k \in \mathbb{R}^n} \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2. \tag{22.2}$$

Удобно определить $\boldsymbol{\mu}(C_i) = (1/|C_i|) \sum_{\mathbf{x} \in C_i} \mathbf{x}$ и заметить, что $\boldsymbol{\mu}(C_i) = \text{argmin}_{\boldsymbol{\mu} \in \mathbb{R}^n} \sum_{\mathbf{x} \in C_i} ||\mathbf{x} - \boldsymbol{\mu}||^2$. Тогда целевую функцию k-средних можно переписать в виде

$$G(C_1, \ldots, C_k) = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \boldsymbol{\mu}(C_i)\|^2. \tag{22.3}$$

Рассмотрим обновление на t-й итерации алгоритма k-средних. Пусть $C_1^{(t-1)}$, ..., $C_k^{(t-1)}$ – предыдущее разбиение, $\boldsymbol{\mu}_i^{(t-1)} = \boldsymbol{\mu}(C_i^{(t-1)})$ и $C_1^{(t)}$, ..., $C_k^{(t)}$ – новое разбиение, вычисленное на t-й итерации. Воспользовавшись определением целевой функции в виде (22.2), мы сразу обнаруживаем, что

$$G(C_1^{(t)}, \ldots, C_k^{(t)}) \leq \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i^{(t)}} \|\mathbf{x} - \boldsymbol{\mu}_i^{(t-1)}\|^2. \tag{22.4}$$

Кроме того, из определения нового разбиения ($C_1^{(t)}$, ..., $C_k^{(t)}$) следует, что оно доставляет минимум выражению $\sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i^{(t-1)}\|^2$ по всем возможным разбиениям (C_1, ..., C_k). Поэтому

$$\sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i^{(t)}} \|\mathbf{x} - \boldsymbol{\mu}_i^{(t-1)}\|^2 \leq \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i^{(t-1)}} \|\mathbf{x} - \boldsymbol{\mu}_i^{(t-1)}\|^2. \tag{22.5}$$

Воспользовавшись выражением (22.3), мы видим, что правая часть неравенства (22.5) равна $G(C_1^{(t-1)}, \ldots, C_k^{(t-1)})$. Объединяя это с неравенствами (22.4) и (22.5), получаем, что $G(C_1^{(t)}, \ldots, C_k^{(t)}) \leq G(C_1^{(t-1)}, \ldots, C_k^{(t-1)})$, что и требовалось доказать. □

Эта лемма говорит, что целевая функция k-средних монотонно неубывающая, но не дает никаких гарантий относительно количества итераций, необходимых для сходимости алгоритма. Кроме того, не существует нетривиальной нижней границы расхождения между значением целевой функции k-средних на выходе алгоритма и минимально возможным значением этой целевой функции. На самом деле алгоритм k-средних может сойтись к точке, которая не является даже локальным минимумом (см. упражнение 22.2). Чтобы улучшить результаты алгоритма, часто рекомендуют повторить процедуру несколько раз с разными случайными начальными значениями центроидов (например, можно в качестве центроидов выбирать случайные входные точки).

22.3. Спектральная кластеризация

Часто отношения между точками множества $\mathcal{X} = \{x_1, \ldots, x_m\}$ удобно представлять *графом подобия*; каждая вершина представляет одну точку x_i, а любые две вершины соединены ребром, вес которого равен их сходству, $W_{i,j} = s(x_i, x_j)$, где $W \in \mathbb{R}^{m,m}$. Например, можно положить $W_{i,j} = \exp(-d(x_i, x_j)^2/\sigma^2)$, где $d(\cdot,\cdot)$ – функция расстояния, а σ – параметр. Теперь проблему кластеризации можно сформулировать следующим образом: найти такое разбиение графа, что веса ребер, соединяющих вершины из разных групп, малы, а веса ребер, соединяющих вершины из одной группы, велики.

В описанных выше целевых функциях кластеризации упор делался на одной стороне нашего интуитивного определения кластеризации – сделать так, чтобы точки, принадлежащие одному кластеру, были похожи. Теперь мы акцентируем

внимание на другом требовании – точки, отнесенные к разным кластерам, должны быть непохожи.

22.3.1. Разрезание графа

Если дан граф, представленный матрицей подобия W, то самый простой и непосредственный способ построить разбиение графа состоит в том, чтобы решить задачу о минимальном разрезе, т. е. выбрать разбиение $C_1, \ldots, C_k$, доставляющее минимум целевой функции

$$\text{cut}(C_1, \ldots, C_k) = \sum_{i=1}^{k} \sum_{r \in C_i,\, s \notin C_i} W_{r,s}.$$

При $k = 2$ задачу о минимальном разрезе можно решить эффективно. Но на практике получающееся при этом разбиение зачастую неудовлетворительно. Проблема в том, что во многих случаях минимальный разрез получается просто отделением одной вершины от всех остальных. Конечно, это не то, чего мы хотим достичь в результате кластеризации; мы стремимся к тому, чтобы в кластерах было достаточно много точек – в пределах разумного.

Было предложено несколько решений этой проблемы. Простейшее из них – нормировать разрез и определить нормированную функцию минимального разреза (mincut) следующим образом:

$$\text{RatioCut}(C_1, \ldots, C_k) = \sum_{i=1}^{k} \frac{1}{|C_i|} \sum_{r \in C_i,\, s \notin C_i} W_{r,s}.$$

Эта целевая функция принимает относительно малые значения, если кластеры не слишком малы. К сожалению, после такой балансировки задача становится вычислительно трудной. Одним из способов ослабить задачу минимизации RatioCut является спектральная кластеризация.

22.3.2. Лапласиан графа и ослабленные разрезы графа

В случае спектральной кластеризации основным математическим объектом является матрица Лапласа графа. В литературе встречается несколько определений лапласиана, мы будем придерживаться одного из них.

Определение 22.2 (ненормированный лапласиан графа). *Ненормированным лапласианом графа* называется матрица $L = D - W$ размера $m \times m$, где D – диагональная матрица с элементами $D_{i,i} = \sum_{j=1}^{m} W_{i,j}$. Матрица D называется *степенной матрицей*.

Следующая лемма устанавливает связь между RatioCut и лапласианом.

Лемма 22.3. *Пусть $C_1, \ldots, C_k$ – кластеризация и $H \in \mathbb{R}^{m,k}$ – такая матрица, что*

$$H_{i,j} = \frac{1}{\sqrt{|C_j|}} \mathbb{1}_{[i \in C_j]}.$$

Тогда столбцы H взаимно ортогональны и

$$\text{RatioCut}(C_1, \ldots, C_k) = \text{trace}(H^\top LH).$$

Доказательство. Пусть $\mathbf{h}_1, \ldots, \mathbf{h}_k$ – столбцы H. Взаимная ортогональность этих векторов сразу следует из определения. Далее с помощью стандартных алгебраических преобразований можно показать, что $\text{trace}(H^\top LH) = \sum_{i=1}^{k} \mathbf{h}_i^\top L \mathbf{h}_i$ и что для любого вектора $\mathbf{v}$ имеет место

$$\mathbf{v}^\top L\mathbf{v} = \frac{1}{2}\left(\sum_r D_{r,r} v_r^2 - 2\sum_{r,s} v_r v_s W_{r,s} + \sum_s D_{s,s} v_s^2\right) = \frac{1}{2}\sum_{r,s} W_{r,s}(v_r - v_s)^2.$$

Подставляя сюда $\mathbf{v} = \mathbf{h}_i$ и замечая, что $(h_{i,r} - h_{i,s})^2$ отлично от нуля, только если $r \in C_i$, $s \notin C_i$, или наоборот, получаем, что

$$\mathbf{h}_i^\top L\mathbf{h}_i = \frac{1}{|C_i|}\sum_{r \in C_i, s \notin C_i} W_{r,s}. \qquad \square$$

Следовательно, чтобы минимизировать RatioCut, мы должны найти матрицу H с ортогональными столбцами и такую, что любой элемент $H_{i,j}$ равен либо 0, либо $1/\sqrt{|C_j|}$. К сожалению, это задача целочисленного программирования, у которой нет эффективного решения. Поэтому мы ослабим последнее требование и будем просто искать ортогональную матрицу $H \in \mathbb{R}^{m,k}$, которая минимизирует $\text{trace}(H^\top LH)$. В следующей главе, посвященной методу главных компонент (и, в частности, в доказательстве теоремы 23.2), мы увидим, что для решения этой задачи нужно взять матрицу U, столбцами которой являются собственные векторы, соответствующие k минимальным собственным значениям L. Получающийся алгоритм называется ненормированной спектральной кластеризацией.

22.3.3. Ненормированная спектральная кластеризация

Ненормированная спектральная кластеризация

вход: $W \in \mathbb{R}^{m,m}$; количество кластеров k
инициализация: вычислить ненормированный лапласиан графа L
обозначим $U \in \mathbb{R}^{m,m}$ матрицу, столбцы которой являются собственными векторами L, соответствующими k наименьшим собственным значениям
обозначим $\mathbf{v}_1, \ldots, \mathbf{v}_m$ строки матрицы U
кластеризовать точки $\mathbf{v}_1, \ldots, \mathbf{v}_m$ методом k-средних
выход: кластеры $C_1, \ldots, C_K$, полученные методом k-средних

Алгоритм спектральной кластеризации сначала ищет матрицу H, состоящую из k собственных векторов, соответствующих наименьшим собственным значениям матрицы Лапласа графа. Затем он берет точки, определяемые строками H. Полезность такой смены представления объясняется свойствами лапласиана графа. Во многих ситуациях новое представление позволяет простому алгоритму k-средних найти кластеры. На интуитивном уровне, если H определена, как

в лемме 22.3, то каждая точка в новом представлении – это индикаторный вектор, в котором не равен нулю только элемент, соответствующий кластеру, которому он принадлежит.

22.4. Метод информационного горлышка*

Метод информационного горлышка (information bottleneck method) – метод кластеризации, предложенный Тишби (Tishby), Перейрой (Pereira) и Бялеком (Bialek). Он опирается на понятия *теории информации*. Для иллюстрации рассмотрим проблему кластеризации текстовых документов, каждый из которых представлен в виде мешка слов, т. е. каждый документ представлен вектором $\mathbf{x} = \{0, 1\}^n$, где n – размер словаря, а $x_i = 1$ тогда и только тогда, когда слово с индексом i встречается в документе. Если дан набор m документов, то его представление в виде мешка слов можно интерпретировать как совместную вероятность случайной величины x, определяющей документ (и, следовательно, принимающей значения из множества $[m]$), и случайной величины y, определяющей слово в словаре (и, следовательно, принимающей значения из множества $[n]$).

При такой интерпретации под информационным горлышком понимается определение кластеризации как еще одной случайной величины C, которая принимает значения из множества $[k]$ (где k также устанавливается методом). Определив x, y, C как случайные величины, мы можем воспользоваться аппаратом теории информации, чтобы выразить целевую функцию кластеризации. Конкретно, целевая функция метода информационного горлышка определена как

$$\min_{p(C|x)} I(x;C) - \beta I(C;y),$$

где $I(\cdot;\cdot)$ – взаимная информация двух случайных величин[1], β – параметр, а минимизация производится по всем возможным вероятностным распределениям точек между кластерами. Интуитивно мы хотели бы достичь двух противоположных целей. С одной стороны, мы хотим, что взаимная информация документа и кластера была как можно меньше. Это значит, что мы стремимся к сильному сжатию исходных данных. С другой стороны, хотелось бы иметь высокую взаимную информацию переменной кластеризации и идентификаторов слов, поскольку это означает стремление сохранить «релевантную» информацию о документе (отраженную в словах, которые в нем встречаются). Это обобщает классическое понятие минимальной достаточной статистики[2], используемое в параметрических статистиках произвольных распределений.

---

[1] То есть при заданном совместном распределении p взаимная информация $I(x; C) = \sum_a \sum_b p(a, b) \log(p(a, b)/(p(a)p(b)))$, где сумма берется по всем возможным значениям x и всем возможным значениям C.

[2] Достаточной статистикой называется функция данных, обладающая свойством достаточности относительно статистический модели и ассоциированного с ней неизвестного параметра в том смысле, что «никакая другая статистика, которую можно вычислить по той же самой выборке, не дает никакой дополнительной информации о значении параметра». Например, если предположить, что случайная величина имеет нормальное распределение с единичной дисперсией и неизвестным математическим ожиданием, то среднее является достаточной статистикой.

В общем случае задача оптимизации, ассоциированная с информационным горлышком, трудна. Некоторые из предложенных методов ее решения, похожи на EM-алгоритм, который мы будем обсуждать в главе 24.

22.5. Общий взгляд на кластеризацию

До сих пор мы в основном перечисляли различные полезные инструменты кластеризации. Но некоторые фундаментальные вопросы так и не получили ответов. И самый главный из них – что же такое кластеризация? Чем алгоритм кластеризации отличается от произвольной функции, которая принимает пространство входов и возвращает его разбиение? Существуют ли какие-то базовые свойства кластеризации, которые не зависят от конкретного алгоритма или задачи?

Один из подходов к таким вопросам – аксиоматический. Было предпринято несколько попыток дать аксиоматическое определение кластеризации. Продемонстрируем одну из них, описанную в работе Kleinberg (2003).

Рассмотрим функцию кластеризации F, которая принимает произвольное конечное множество $\mathcal{X}$ с функций несходства d, определенной на парах элементов, и возвращает разбиение $\mathcal{X}$.

Рассмотрим следующие три свойства такой функции.

- **Масштабная инвариантность (SI).** Для любой области определения $\mathcal{X}$, любой функции несходства d и любого $\alpha > 0$ имеет место следующее свойство: $F(\mathcal{X}, d) = F(\mathcal{X}, \alpha d)$ (где $(\alpha d)(x, y) \stackrel{\text{def}}{=} \alpha d(x, y)$).
- **Богатство (Ri).** Для любой конечной области $\mathcal{X}$ и любого разбиения $C = (C_1, \ldots, C_k)$ $\mathcal{X}$ на непустые подмножества существует функция несходства d на $\mathcal{X}$ такая, что $F(\mathcal{X}, d) = C$.
- **Согласованность (Co).** Если d и d' – функции несходства на $\mathcal{X}$ такие, что для любых $x, y \in \mathcal{X}$ имеет место следующее утверждение: если x, y принадлежат одному и тому же кластеру в $F(\mathcal{X}, d)$, то $d'(x, y) \leq d(x, y)$, а если x, y принадлежат разным кластерам в $F(\mathcal{X}, d)$, то $d'(x, y) \geq d(x, y)$ – то $F(\mathcal{X}, d) = F(\mathcal{X}, d')$.

Немного поразмыслив, мы поймем, что масштабная инвариантность – совершенно естественное требование, поскольку было бы странно, если бы результат функции кластеризации зависел от единицы измерения расстояния между точками. Требование богатства по существу означает, что результат функции кластеризации полностью контролируется функцией d, что тоже согласуется с интуицией. Требование согласованности – единственное из всех, которое можно отнести к основному (неформальному) определению кластеризации: мы хотим, чтобы похожие точки попадали в один кластер, а непохожие – в разные, поэтому если точки, уже находящиеся в одном кластере, становятся более похожими, а точки, уже находящиеся в разных кластерах, – менее похожими, то функция кластеризации должна еще сильнее «поддерживать» принятые ранее решения о кластеризации.

Однако в работе Kleinberg (2003) доказан следующий результат о невозможности.

Теорема 22.4. *Не существует функции F, удовлетворяющей всем трем свойствам: масштабной инвариантности, богатства и согласованности.*

Доказательство. Предположим противное – что некоторая функция F удовлетворяет всем трем свойствам. Возьмем некоторую область определения $\mathcal{X}$, содержащую не менее трех точек. Согласно свойству богатства, должна существовать функция d_1 такая, что $F(\mathcal{X}, d_1) = \{\{x\} : x \in \mathcal{X}\}$, и функция d_2 такая, что $F(\mathcal{X}, d_2) \neq F(\mathcal{X}, d_1)$.

Пусть $\alpha \in \mathbb{R}_+$ таково, что для любых $x, y \in \mathcal{X}$, $\alpha d_2(x, y) \geq d_1(x, y)$. Положим $d_3 = \alpha d_2$. Рассмотрим $F(\mathcal{X}, d_3)$. Согласно свойству масштабной инвариантности F, должно быть $F(\mathcal{X}, d_3) = F(\mathcal{X}, d_2)$. С другой стороны, поскольку все различные $x, y \in \mathcal{X}$ находятся в разных кластерах относительно $F(\mathcal{X}, d_1)$ и $d_3(x, y) \geq d_1(x, y)$, из согласованности F следует, что $F(\mathcal{X}, d_3) = F(\mathcal{X}, d_1)$. Мы пришли к противоречию, поскольку выбирали d_1, d_2 так, что $F(\mathcal{X}, d_2) \neq F(\mathcal{X}, d_1)$. □

Важно отметить, что среди этих трех свойств нет какой-то одной «плохой аксиомы». Для любых двух аксиом существуют естественные удовлетворяющие им функции кластеризации (примеры можно построить, просто изменяя критерий остановки в функции кластеризации из метода одиночной связи). С другой стороны, Клейнберг показал, что любой алгоритм кластеризации, который минимизирует центральную целевую функцию, неизбежно не удовлетворяет свойству согласованности (правда, кластеризация путем минимизации суммы k внутрикластерных расстояний этому свойству удовлетворяет).

Результат Клейнберга о невозможности можно обойти, варьируя свойства. Например, если мы хотим обсуждать функции кластеризации с заранее заданным количеством кластеров, то естественно было бы заменить богатство k-богатством (т. е. потребовать, чтобы любое разбиение области определения на k подмножеств было достижимо с помощью функции кластеризации). Свойства k-богатства, масштабной инвариантности и согласованности одновременно выполняются для кластеризации методом k-средних, и потому такая система аксиом непротиворечива. Можно вместо этого ослабить свойство согласованности. Например, будем говорить, что две кластеризации $C = (C_1, \ldots, C_k)$ и $C' = (C'_1, \ldots, C'_l)$ *совместимы*, если для любых кластеров $C_i \in C$ и $C'_j \in C'$ либо $C_i \subseteq C'_j$, либо $C'_j \subseteq C_i$, либо $C_i \cap C'_j = \emptyset$ (стоит отметить, что для любой дендрограммы любые две кластеризации, полученные редукцией дендрограммы, совместимы). Свойство «уточненной согласованности» заключается в том, что при выполнении условий свойства согласованности новая кластеризация $F(\mathcal{X}, d')$ совместима со старой $F(\mathcal{X}, d)$. Многие распространенные функции кластеризации удовлетворяют этому свойству наряду с масштабной инвариантностью и богатством. Более того, можно найти много других свойств функций кластеризации, которые кажутся интуитивно понятными и желательными и удовлетворяются некоторыми популярными функциями.

Есть много способов интерпретировать эти результаты. Мы предлагаем рассматривать их как свидетельство отсутствия «идеальной» функции кластеризации. У любой функции кластеризации неизбежно будут какие-то нежелательные свойства. Поэтому при выборе функции для конкретной задачи следует принимать во внимание особенности именно этой задачи. Не существует общего

решения для кластеризации, как не существует алгоритма классификации, позволяющего обучить любую допускающую обучение задачу (это содержание теоремы об отсутствии бесплатных завтраков). Кластеризация, как и классификация, должна учитывать априорные знания о решаемой задаче.

22.6. Резюме

Кластеризация – проблема обучения без учителя, в которой требуется разбить множество точек на «осмысленные» подмножества. Мы представили несколько подходов к кластеризации, в т. ч. алгоритмы, основанные на связи, семейство методов k-средних, спектральную кластеризацию и информационное горлышко. Мы обсудили трудности, с которыми сталкиваются попытки формализовать интуитивное представление о кластеризации.

22.7. Библиографические сведения

Алгоритм k-средних иногда называют алгоритмом Ллойда, в честь Стюарта Ллойда, предложившего его в 1957 г. Читателей, интересующихся более подробным обзором спектральной кластеризации, отсылаем к великолепному пособию фон Люксбург (von Luxburg, 2007). Метод информационного горлышка был предложен в работе Tishby, Pereira and Bialek (1999). Дополнительное обсуждение аксиоматического подхода см. в работе Ackerman and Ben-David (2008).

22.8. Упражнения

22.1. Субоптимальность метода k-средних. Докажите, что при любом параметре $t > 1$ существует пример, в котором алгоритм k-средних мог бы найти решение, для которого целевая функция k-средних принимает значение не меньше $t \cdot \text{OPT}$, где OPT – минимум этой целевой функции.

22.2. Метод k-средних необязательно сходится к локальному минимуму. Покажите, что алгоритм k-средних может сойтись к точке, не являющейся локальным минимумом. *Указание.* Пусть $k = 2$ и областью определения является множество $\{1, 2, 3, 4\} \subset \mathbb{R}$. Рассмотрите случай, когда на этапе инициализации были выбраны центры $\{2, 4\}$, а для устранения неоднозначности в определении C_i переменной i присваивается наименьшее значение в множестве $\text{argmin}_j \|\mathbf{x} - \boldsymbol{\mu}_j\|$.

22.3. Пусть дано метрическое пространство $(\mathcal{X}, d)$, где $|\mathcal{X}| < \infty$, и $k \in \mathrm{N}$. Мы хотим найти разбиение $\mathcal{X}$ на подмножества $C_1, \ldots, C_k$, доставляющее минимум выражению

$$G_{\text{k-diam}}((\mathcal{X}, d), (C_1, \ldots, C_k)) = \max_{j \in [d]} \text{diam}(C_j),$$

где $\text{diam}(C_j) = \max_{x, x' \in C_j} d(x, x')$ (мы считаем, что $\text{diam}(C_j) = 0$, если $|C_j| < 2$).

Задача минимизации *k-diam* является NP-трудной, как и задача минимизации целевой функции *k*-средних. Но по счастью, существует очень простой аппроксимирующий алгоритм: вначале выберем некоторое $x \in \mathcal{X}$ и положим $\mu_1 = x$. Затем алгоритм итеративно устанавливает

$$\forall i \in \{2, \ldots, k\},\ \mu_j = \operatorname*{argmax}_{x \in X} \min_{i \in [j-1]} d(x, \mu_j).$$

Наконец, мы полагаем

$$\forall i \in \{k\},\ C_i = \{x \in X : i = \operatorname*{argmin}_{j \in [k]} d(x, \mu_j)\}.$$

Докажите, что описанный алгоритм является 2-аппроксимирующим. Это означает, что если обозначить его выход $\hat{C}_1, \ldots, \hat{C}_k$, а оптимальное решение обозначить $C_1^*, \ldots, C_k^*$, то

$$G_{\text{k–diam}}((\mathcal{X}, d), (\hat{C}_1, \ldots, \hat{C}_k)) \leq 2 \cdot G_{\text{k–diam}}((\mathcal{X}, d), (C_1^*, \ldots, C_k^*)).$$

Указание. Рассмотрим точку μ_{k+1}(т. е. следующий центр, который мы выбрали бы, если бы захотели получить $k + 1$ кластеров). Положим $r = \min_{j \in [k]} d(\mu_j, \mu_{k+1})$. Докажите справедливость следующих неравенств:

$$G_{\text{k–diam}}((\mathcal{X}, d), (\hat{C}_1, \ldots, \hat{C}_k)) \leq 2r;$$
$$G_{\text{k–diam}}((\mathcal{X}, d), (C_1^*, \ldots, C_k^*)) \geq r.$$

22.4. Напомним, что функция кластеризации F называется центральной, если для некоторой монотонной функции $f : \mathbb{R}_+ \to \mathbb{R}_+$ при любых входных данных $(\mathcal{X}, d)$, $F(\mathcal{X}, d)$ является кластеризацией, доставляющей минимум целевой функции

$$G_f((\mathcal{X}, d), (C_1, \ldots, C_k)) = \min_{\mu_1, \ldots, \mu_k \in \mathcal{X}'} \sum_{i=1}^{k} \sum_{x \in C_i} f(d(x, \mu_i)),$$

где $\mathcal{X}'$ либо совпадает с $\mathcal{X}$, либо является надмножеством $\mathcal{X}$.

Докажите, что для любого $k > 1$ функция кластеризации *k-diam*, определенная в предыдущем упражнении, не является центральной.

Указание. Для входных данных кластеризации $(\mathcal{X}, d)$ с $|\mathcal{X}| > 2$ рассмотрите, что случится с кластеризацией *k-diam* и с любой центральной кластеризацией, если добавить много близких точек в некоторые (но не все) члены $\mathcal{X}$.

22.5. Напомним, что мы обсуждали три «свойства» кластеризации: масштабную инвариантность, богатство и согласованность. Рассмотрим кластеризацию методом одиночной связи.

1. Найдите, какие из этих трех свойств удовлетворяются в методе одиночной связи с остановкой после формирования заранее заданного числа кластеров (любого отличного от нуля).
2. Найдите, какие из этих трех свойств удовлетворяются в методе одиночной связи с остановкой после достижения верхней границы расстояния (любой отличной от нуля).
3. Покажите, что для любых двух из этих свойств существует критерий остановки для кластеризации методом одиночной связи, при котором эти два свойства удовлетворяются.

22.6. Пусть дано некоторое число k, назовем k-богатством следующее требование:

Для любого конечного множества X и любого разбиения $C = (C_1, \ldots, C_k)$ этого множества на непустые подмножества существует функция несходства d на X такая, что $F(X, d) = C$.

Докажите, что для любого k существует функция кластеризации, удовлетворяющая всем трем свойствам: масштабной инвариантности, k-богатства и согласованности.

ЛАНЬ®

Глава 23

ПОНИЖЕНИЕ РАЗМЕРНОСТИ

Понижение размерности – это процедура отображения данных из пространства высокой размерности в пространство гораздо более низкой размерности. Оно тесно связано с идеей сжатия (с потерями) в теории информации. Для понижения размерности есть несколько причин. Во-первых, чем выше размерность, тем сложнее вычисления. Кроме того, в некоторых ситуациях высокая размерность ухудшает способность алгоритма обучения к обобщению (например, в алгоритме ближайших соседей выборочная сложность экспоненциально возрастает с увеличением размерности – см. главу 19). Наконец, понижение размерности упрощает интерпретацию данных, выявление в них осмысленной структуры и полезно для целей иллюстрации.

В этой главе мы опишем популярные методы понижения размерности. Все они основаны на применении к данным некоторого линейного преобразования. То есть, если исходные данные находятся в пространстве $\mathbb{R}^d$, а мы хотим погрузить их в пространство $\mathbb{R}^n (n < d)$, то необходимо найти матрицу $W \in \mathbb{R}^{n,d}$, индуцирующую отображение $\mathbf{x} \mapsto W\mathbf{x}$. Выбирать W естественно таким способом, который допускает восстановление исходного $\mathbf{x}$ с разумной точностью. Нетрудно показать, что в общем случае точное восстановление $\mathbf{x}$ по $W\mathbf{x}$ невозможно (см. упражнение 23.1).

Первый метод называется методом главных компонент (Principal Component Analysis – PCA). В этом случае сжатие и восстановление производятся с помощью линейных преобразований, а алгоритм ищет такие преобразования, для которых сумма квадратов расхождений между исходным и восстановленным вектором минимальна.

Затем мы опишем понижение размерности с помощью случайной матрицы W. Мы докажем важную лемму Джонсона–Линденштрауса, которая анализирует искажение, вносимое рандомизированной техникой понижения размерности.

Наконец, мы покажем, как понизить размерности *разреженных* векторов, опять-таки с помощью случайной матрицы. Эта процедура называется сжатым измерением сигнала (compressed sensing). Процесс восстановления в этом случае нелинейный, но все же может быть эффективно реализован методами линейного программирования.

И завершим главу обсуждением «априорных предположений», лежащих в основе PCA и сжатого измерения сигнала, поскольку это поможет понять достоинства и недостатки обоих методов.

23.1. Метод главных компонент (PCA)

Пусть $\mathbf{x}_1, \ldots, \mathbf{x}_m$ – m векторов в $\mathbb{R}^d$. Мы хотим понизить размерность этих векторов с помощью линейного преобразования. Матрица $W \in \mathbb{R}^{n,d}$, где $n < d$, индуцирует отображение $\mathbf{x} \mapsto W\mathbf{x}$, где $W\mathbf{x} \in \mathbb{R}^n$ – представление $\mathbf{x}$ меньшей размерности. Другую матрицу $U \in \mathbb{R}^{d,n}$ можно использовать для приближенного восстановления исходного вектора $\mathbf{x}$ по его сжатому представлению. То есть для сжатого вектора $\mathbf{y} = W\mathbf{x}$, где $\mathbf{y}$ находится в пространстве низкой размерности $\mathbb{R}^n$, можно построить вектор $\tilde{\mathbf{x}} = U\mathbf{y}$ такой, что $\tilde{\mathbf{x}}$ – восстановленный вариант $\mathbf{x}$, находящийся в исходном пространстве высокой размерности $\mathbb{R}^d$.

В методе PCA мы ищем матрицы сжатия W и восстановления U, так чтобы квадрат расстояния между исходным и восстановленным вектором был минимален, т. е. решаем следующую задачу оптимизации

$$\operatorname*{argmin}_{W \in \mathbb{R}^{n,d}, U \in \mathbb{R}^{d,n}} \sum_{i=1}^{m} \|\mathbf{x}_i - UW\mathbf{x}_i\|_2^2. \tag{23.1}$$

Но прежде покажем, что оптимальное решение имеет определенный вид.

Лемма 23.1. *Пусть* (U, W) *– решение задачи (23.1). Тогда столбцы* U *ортогональны (т. е.* $U^\top U$ *– единичная матрица в* $\mathbb{R}^n$*) и* $W = U$.

Доказательство. Зафиксируем произвольные U, W и рассмотрим отображение $\mathbf{x} \mapsto UW\mathbf{x}$. Область значений этого отображения $R = \{UW\mathbf{x} : \mathbf{x} \in \mathbb{R}^d\}$ – n-мерное линейное подпространство $\mathbb{R}^d$. Обозначим $V \in \mathbb{R}^{d,n}$ матрицу, столбцы которой образуют ортонормированный базис этого подпространства, т. е. натянутая на них линейная оболочка совпадает с R и $V^\top V = I$. Таким образом, любой вектор из R можно записать в виде $V\mathbf{y}$, где $\mathbf{y} \in \mathbb{R}^n$. Для любых $\mathbf{x} \in \mathbb{R}^d$ и $\mathbf{y} \in \mathbb{R}^n$ имеем

$$\|\mathbf{x} - V\mathbf{y}\|_2^2 = \|\mathbf{x}\|^2 + \mathbf{y}^\top V^\top V\mathbf{y} - 2\mathbf{y}^\top V^\top \mathbf{x} = \|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 - 2\mathbf{y}^\top (V^\top \mathbf{x}),$$

где мы воспользовались тем фактом, что $V^\top V$ – единичная матрица в $\mathbb{R}^n$. Для минимизации этого выражения относительно $\mathbf{y}$ мы приравниваем градиент по $\mathbf{y}$ нулю и получаем, что $\mathbf{y} = V\mathbf{x}$. Следовательно, для любого $\mathbf{x}$ имеет место равенство

$$VV^\top \mathbf{x} = \operatorname*{argmin}_{\tilde{\mathbf{x}} \in \mathbb{R}} \|\mathbf{x} - \tilde{\mathbf{x}}\|_2^2.$$

В частности, оно справедливо для векторов $\mathbf{x}_1, \ldots, \mathbf{x}_m$, и потому мы можем заменить U, W на V, V, не увеличив при этом целевую функцию

$$\sum_{i=1}^{m} \|\mathbf{x}_i - UW\mathbf{x}_i\|_2^2 \geq \sum_{i=1}^{m} \|\mathbf{x}_i - VV^\top \mathbf{x}_i\|_2^2.$$

Поскольку это справедливо для любых U и W, утверждение леммы доказано. □

В силу леммы мы можем переписать задачу оптимизации (23.1) в виде:

$$\operatorname*{argmin}_{U \in \mathbb{R}^{d,n} : U^\top U = I} \sum_{i=1}^{m} \|\mathbf{x}_i - UU^\top \mathbf{x}_i\|_2^2. \tag{23.2}$$

Еще упростим задачу, выполнив ряд элементарных преобразований. Для любого $\mathbf{x} \in \mathbb{R}^d$ и любой матрицы $U \in \mathbb{R}^{d,n}$ такой, что $U^\top U = I$, имеем

$$\begin{aligned}||\mathbf{x} - UU^{\mathrm{T}}\mathbf{x}||^2 &= ||\mathbf{x}||^2 - 2\mathbf{x}^{\mathrm{T}}UU^{\mathrm{T}}\mathbf{x} + \mathbf{x}^{\mathrm{T}}UU^{\mathrm{T}}UU^{\mathrm{T}}\mathbf{x} \\ &= ||\mathbf{x}||^2 - \mathbf{x}^{\mathrm{T}}UU^{\mathrm{T}}\mathbf{x} \\ &= ||\mathbf{x}||^2 - \mathrm{trace}(U^{\mathrm{T}}\mathbf{x}\mathbf{x}^{\mathrm{T}}U), \end{aligned} \tag{23.3}$$

где следом (trace) матрицы называется сумма ее диагональных элементов. Поскольку след – линейный оператор, это позволяет переписать (23.2) следующим образом:

$$\underset{U\in\mathbb{R}^{d,n}:U^{\mathrm{T}}U=I}{\mathrm{argmax}}\ \mathrm{trace}\left(U^{\mathrm{T}}\sum_{i=1}^{m}\mathbf{x}_i\mathbf{x}_i^{\mathrm{T}}\,\mathrm{U}\right). \tag{23.4}$$

Положим $A = \sum_{i=1}^{m}\mathbf{x}_i\mathbf{x}_i^{\mathrm{T}}$. Матрица A симметрична, так что для нее существует спектральное разложение $A = VDV^{\mathrm{T}}$, где D – диагональная матрица, а $V^{\mathrm{T}}V = VV^{\mathrm{T}} = I$. Диагональными элементами D являются собственные значения A, а столбцы V – соответствующие им собственные векторы. Без ограничения общности можно считать, что $D_{1,1} \geq D_{2,2} \geq \ldots \geq D_{d,d}$. Так как матрица A положительно полуопределенная, то верно также, что $D_{d,d} \geq 0$. Мы утверждаем, что решением задачи (23.4) является матрица U, столбцами которой являются n собственных векторов A, соответствующих n наибольшим собственным значениям.

Теорема 23.2. *Пусть* $\mathbf{x}_1, \ldots, \mathbf{x}_m$ *– произвольные векторы в* $\mathbb{R}^d$, $A = \sum_{i=1}^{m}\mathbf{x}_i\mathbf{x}_i^{\mathrm{T}}$ *и* $\mathbf{u}_1, \ldots, \mathbf{u}_n$ *– n собственных векторов A, соответствующих n наибольшим собственным значениям. Тогда решением задачи оптимизации PCA (23.1) будет матрица со столбцами* $\mathbf{u}_1, \ldots, \mathbf{u}_n$ *и* $W = U^{\mathrm{T}}$.

Доказательство. Пусть VDV^{T} – спектральное разложение A. Зафиксируем некоторую матрицу $U \in \mathbb{R}^{d,n}$ с ортогональными столбцами и положим $B = V^{\mathrm{T}}U$. Тогда $VB = VV^{\mathrm{T}}U = U$. Отсюда следует, что

$$U^{\mathrm{T}}AU = B^{\mathrm{T}}V^{\mathrm{T}}VDV^{\mathrm{T}}VB = B^{\mathrm{T}}DB,$$

и, следовательно,

$$\mathrm{trace}(U^{\mathrm{T}}AU) = \sum_{j=1}^{d}D_{j,j}\sum_{i=1}^{n}B_{j,i}^2.$$

Заметим, что $B^{\mathrm{T}}B = U^{\mathrm{T}}VV^{\mathrm{T}}U = U^{\mathrm{T}}U = I$. Поэтому столбцы B также ортогональны, откуда следует, что $\sum_{j=1}^{d}\sum_{i=1}^{n}B_{j,i}^2 = n$. Кроме того, обозначим $\tilde{B} \in \mathbb{R}^{d,d}$ матрицу, первые n столбцов которой совпадают со столбцами B и $\tilde{B}^{\mathrm{T}}\tilde{B} = I$. Тогда для любого j имеем $\sum_{i=1}^{d}\tilde{B}_{j,i}^2 = 1$, откуда следует, что $\sum_{i=1}^{n}B_{j,i}^2 \leq 1$. Отсюда следует, что

$$\mathrm{trace}(U^{\mathrm{T}}AU) = \max_{\boldsymbol{\beta}\in[0,1]^d:\,\|\boldsymbol{\beta}\|_1\leq n}\sum_{j=1}^{d}D_{j,j}b_j.$$

Нетрудно проверить (см. 23.2), что правая часть равна $\sum_{j=1}^{n}D_{j,j}$. Таким образом, мы показали, что для любой матрицы $U \in \mathbb{R}^{d,n}$ с ортогональными столбцами $\mathrm{trace}(U^{\mathrm{T}}AU) \leq \sum_{j=1}^{n}D_{j,j}$. С другой стороны, если взять в качестве U матрицу, столбцами которой являются первые n собственных векторов A, то получим, что $\mathrm{trace}(U^{\mathrm{T}}AU) = \sum_{j=1}^{n}D_{j,j}$, и утверждение теоремы доказано. □

Замечание 23.1. Из доказательства теоремы 23.2 видно также, что значение целевой функции в задаче (23.4) равно $\sum_{i=1}^{n}D_{i,i}$. Объединяя с равенством (23.3) и за-

мечая, что $\sum_{i=1}^{m}||\mathbf{x}_i||^2 = \text{trace}(A) = \sum_{i=1}^{d} D_{i,i}$, получаем, что оптимальное значение целевой функции (23.1) равно $\sum_{i=n+1}^{d} D_{i,i}$.

ЛАНЬ®

Замечание 23.2. Общепринятой практикой является «центрирование» примеров перед применением PCA. То есть мы сначала вычисляем $\boldsymbol{\mu} = (1/m)\sum_{i=1}^{m}\mathbf{x}_i$, а затем применяем PCA к векторам $(\mathbf{x}_1 - \boldsymbol{\mu}), \ldots, (\mathbf{x}_m - \boldsymbol{\mu})$. Это также связано с интерпретацией PCA как максимизации дисперсии (см. упражнение 23.4).

23.1.1. Более эффективное решение для случая $d \gg m$

В некоторых ситуациях исходная размерность данных много больше количества примеров m. Вычислительная сложность нахождения описанного выше решения методом PCA составляет $O(d^3)$ (нахождение собственных значений A) плюс $O(md^2)$ (построение матрицы A). А сейчас мы покажем простой прием, который позволяет находить решение PCA более эффективно, если $d \gg m$.

Напомним, что матрица A определена как $\sum_{i=1}^{m}\mathbf{x}_i\mathbf{x}_i^\top$. Удобно переписать это в виде $A = X^\top X$, где $X \in \mathbb{R}^{m,d}$ – матрица, i-я строка которой равна $\mathbf{x}_i^\top$. Рассмотрим матрицу $B = XX^\top$. Иными словами, (i, j)-й элемент матрицы $B \in \mathbb{R}^{m,m}$ равен $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$. Пусть $\mathbf{u}$ – собственный вектор B, т. е. $B\,\mathbf{u} = \lambda\mathbf{u}$ для некоторого $\lambda \in \mathbb{R}$. Умножив это равенство на $X^\top$ и воспользовавшись определением B, получим $X^\top XX^\top\mathbf{u} = \lambda X^\top\mathbf{u}$. Но из определения A следует, что $A(X^\top\mathbf{u}) = \lambda(X\mathbf{u})$. Таким образом, $(X^\top\mathbf{u})/||X^\top\mathbf{u}||$ – собственный вектор A с собственным значением λ.

Стало быть, мы можем найти решение PCA, вычислив собственные значения B вместо A. Сложность составляет $O(m^3)$ (вычисление собственных значений B) плюс m^2d (построение матрицы B).

Замечание 23.3. Из предыдущего обсуждения также следует, что для нахождения решения PCA нужно только уметь вычислять скалярные произведения векторов. Это позволяет выполнять алгоритм PCA неявно, когда d очень велико (или даже бесконечно), с помощью ядер. Тем самым мы приходим к *ядерному* алгоритму PCA.

23.1.2. Реализация и демонстрация

Ниже приведен псевдокод алгоритма PCA.

Алгоритм PCA

вход
  матрица m примеров $X \in \mathbb{R}^{m,d}$
  количество компонент n
if $(m > d)$
  $A = X^\top X$
  положить $\mathbf{u}_1, \ldots, \mathbf{u}_n$ равными собственным векторам A, соответствующим наибольшим собственным значениям
else
  $B = XX^\top$
  положить $\mathbf{v}_1, \ldots, \mathbf{v}_n$ равными собственным векторам B, соответствующим наибольшим собственным значениям
  for $i = 1, \ldots, n$ положить $\mathbf{u}_i = (1/||X^\top\mathbf{v}_i||)X^\top\mathbf{v}_i$
выход: $\mathbf{u}_1, \ldots, \mathbf{u}_n$

Для иллюстрации работы PCA сгенерируем векторы в $\mathbb{R}^2$, располагающиеся приблизительно на одной прямой, т. е. в одномерном подпространстве $\mathbb{R}^2$. Например, пусть каждый пример имеет вид $(x, x + y)$, где x случайно выбирается из равномерного распределения на отрезке $[-1, 1]$, y – из нормального распределения со средним 0 и стандартным отклонением 0,1. Применим к этим данным алгоритм PCA. Тогда собственный вектор, соответствующий наибольшему собственному значению, будет близок к вектору $(1/\sqrt{2}, 1/\sqrt{2})$. При проецировании точки $(x, x + y)$ на главную компоненту мы получим скаляр $(2x + y)/\sqrt{2}$. После реконструкции получится вектор $((x + y/2), (x + y/2))$. На рис. 23.1 изображены исходные и реконструированные данные.

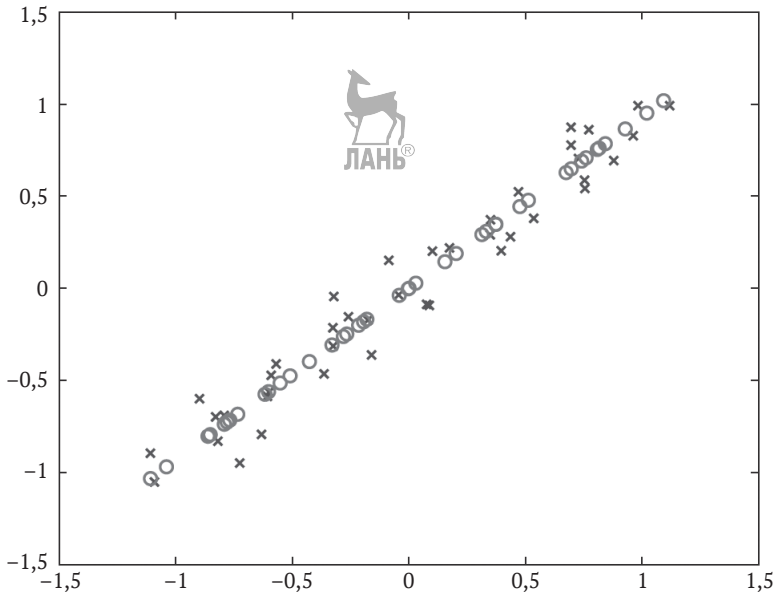


Рис. 23.1. Множество векторов в $\mathbb{R}^2$ (крестики) и их реконструкций после понижения размерности до $\mathbb{R}^1$ методом PCA (кружочки)

Далее мы продемонстрируем эффективность PCA в применении к набору данных о лицах. Мы взяли изображения лиц из набора данных Йельского университета (Georghiades, Belhumeur & Kriegman, 2001). Каждое изображение состоит из $50 \times 50 = 2500$ пикселей, т. е. исходная размерность очень высока.

В левой верхней части рис. 23.2 показаны изображения нескольких лиц. Применяя PCA, мы понизили размерность до $\mathbb{R}^{10}$, а затем реконструировали исходное изображение. Результат реконструкции показан справа вверху. Наконец, в нижней части рисунка показаны двумерные представления тех же изображений. Как видим, даже по двумерной картинке все еще можно как-то различить разных людей.

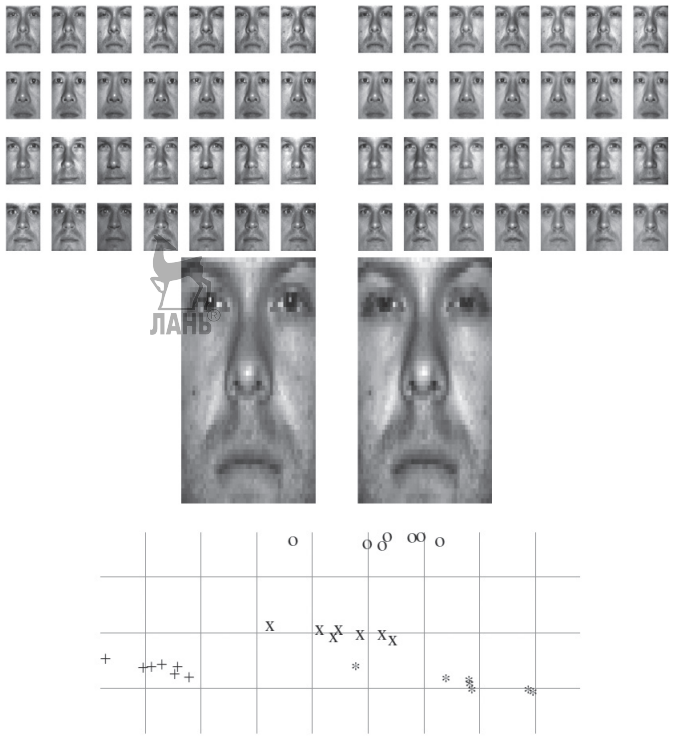


Рис. 23.2. Изображения лиц, взятые из набора данных Йельского университета. Слева вверху: исходные изображения в пространстве $\mathbb{R}^{50\times 50}$. Справа вверху: изображения после понижения размерности до $\mathbb{R}^{10}$ и последующей реконструкции. В середине: одно увеличенное изображение до и после применения PCA. Внизу: изображения после понижения размерности до $\mathbb{R}^2$. Разными значками обозначены лица разных людей

23.2. Случайные проекции

В этом разделе мы покажем, что понижение размерности путем использования случайного линейного преобразования приводит к простой схеме сжатия с поразительно низким искажением. Преобразование $\mathbf{x} \mapsto W\mathbf{x}$, где W – случайная матрица, часто называют случайной проекцией. В частности, мы сформулируем вариант знаменитой леммы Джонсона и Линденштрауса, показывающей, что случайное проецирование искажает евклидовы расстояния не слишком сильно.

Пусть $\mathbf{x}_1$, $\mathbf{x}_2$ – два вектора в $\mathbb{R}^d$. Матрица W не слишком сильно искажает расстояние между $\mathbf{x}_1$ и $\mathbf{x}_2$, если отношение

$$\frac{\|W\mathbf{x}_1 - W\mathbf{x}_2\|}{\|\mathbf{x}_1 - \mathbf{x}_2\|}$$

близко к 1. Иными словами, расстояния между $\mathbf{x}_1$ и $\mathbf{x}_2$ до и после преобразования почти одинаковы. Чтобы показать, что $\|W\mathbf{x}_1 - W\mathbf{x}_2\|$ не слишком отличается от

$||\mathbf{x}_1 - \mathbf{x}_2||$, достаточно убедиться, что W не сильно искажает норму вектора–разности $\mathbf{x} = \mathbf{x}_1 - \mathbf{x}_2$. Поэтому далее мы будем рассматривать отношение $||W\mathbf{x}||/||\mathbf{x}||$.

Начнем с анализа искажения, вызванного применением случайной проекции к одному вектору.

Лемма 23.3. *Зафиксируем некоторый вектор $\mathbf{x} \in \mathbb{R}^d$. Пусть $W \in \mathbb{R}^{n,d}$ – случайная матрица такая, что все элементы $W_{i,j}$ – независимые нормально распределенные случайные величины. Тогда для любого $\epsilon \in (0, 3)$ имеем*

$$\mathbb{P}\left[\left|\frac{\|(1/\sqrt{n})W\mathbf{x}\|^2}{\|\mathbf{x}\|^2} - 1\right| > \epsilon\right] \leq 2e^{-\epsilon^2 n/6}.$$

Доказательство. Без ограничения общности можно предположить, что $||\mathbf{x}||^2 = 1$. Поэтому неравенство можно переписать в эквивалентном виде:

$$\mathbb{P}(1 - \epsilon)n \leq ||W\mathbf{x}||^2 \leq (1 + \epsilon)n \geq 1 - 2e^{-\epsilon^2 n/6}.$$

Обозначим $\mathbf{w}_i$ i-ю строку W. Случайная величина $\langle \mathbf{w}_i, \mathbf{x} \rangle$ является взвешенной суммой d нормально распределенных случайных величин и потому имеет нормальное распределение с нулевым средним и дисперсией $\Sigma_j x_j^2 = ||\mathbf{x}||^2 = 1$. Поэтому случайная величина $||W\mathbf{x}||^2 = \sum_{i=1}^n (\langle \mathbf{w}_i, \mathbf{x} \rangle)^2$ имеет распределение χ_n^2. Теперь утверждение следует из свойства концентрации меры случайных величин с распределением χ^2, составляющего содержание леммы B.12 в разделе B.7. □

Лемма Джонсона–Линденштрауса выводится отсюда путем простого рассуждения, основанного на лемме о границе объединения.

Лемма 23.4 (лемма Джонсона-Линденштрауса). *Пусть Q – конечное множество векторов в $\mathbb{R}^d$. Пусть $\delta \in (0, 1)$ и целое число n таковы, что*

$$\epsilon = \sqrt{\frac{6\log(2|Q|/\delta)}{n}} \leq 3.$$

Тогда с вероятностью не менее $1 - \delta$ для случайной матрицы $W \in \mathbb{R}^{n,d}$, все элементы которой имеют нормальное распределение с нулевым средним и дисперсией $1/n$, имеет место оценка

$$\sup_{\mathbf{x} \in Q} \left|\frac{\|W\mathbf{x}\|^2}{\|\mathbf{x}\|^2} - 1\right| < \epsilon.$$

Доказательство. Лемма 23.3 в сочетании с леммой о границе объединения показывают, что для любого $\epsilon \in (0, 3)$

$$\mathbb{P}\left[\sup_{\mathbf{x} \in Q} \left|\frac{\|W\mathbf{x}\|^2}{\|\mathbf{x}\|^2} - 1\right| > \epsilon\right] \leq 2|Q|e^{-\epsilon^2 n/6}.$$

Обозначим δ правую часть неравенства, тогда

$$\epsilon = \sqrt{\frac{6\log(2|Q|/\delta)}{n}}.$$

□

Интересно, что граница в лемме 23.4 не зависит от исходной размерности $\mathbf{x}$. На самом деле она остается в силе, даже если $\mathbf{x}$ принадлежит бесконечномерному гильбертову пространству.

23.3. Сжатое измерение сигнала

Сжатое измерение сигнала (compressed sensing) – метод понижения размерности, в котором используется априорное предположение о том, что исходный вектор является разреженным в некотором базисе. Чтобы понять, зачем это нужно, рассмотрим вектор $\mathbf{x} \in \mathbb{R}^d$, содержащий не более s ненулевых элементов, т. е.

$$||\mathbf{x}||_0 \stackrel{\text{def}}{=} |\{i : x_i = 0\}| \leq s.$$

Очевидно, что $\mathbf{x}$ можно сжать, представив его с помощью s пар (индекс, значение). При таком сжатии информация не теряется – мы можем точно реконструировать $\mathbf{x}$ по s парам. Сделаем еще один шаг и предположим, что $\mathbf{x} = U\boldsymbol{\alpha}$, где $\boldsymbol{\alpha}$ – разреженный вектор, $||\boldsymbol{\alpha}||_0 \leq s$, а U – фиксированная ортогональная матрица. Это означает, что $\mathbf{x}$ имеет разреженное представление в другом базисе. Оказывается, что многие естественно возникающие векторы разрежены в некотором представлении (по крайней мере, приближенно). Например, формат JPEG-2000 для сжатия изображений основан на том факте, что естественные изображения приближенно разрежены в вейвлетном базисе.

Так можем ли мы приближенно представить $\mathbf{x}$ с помощью s чисел? Простой способ решить эту задачу – умножить $\mathbf{x}$ на U^{T}, получив тем самым разреженный вектор α, а затем представить α с помощью s пар (индекс, значение). Однако для этого нужно сначала «измерить» $\mathbf{x}$, сохранить его и только потому множить на U^{T}. Возникает естественный вопрос: зачем тратить усилия на сбор всех данных, если большая их часть будет отброшена? Нельзя ли сразу измерить ту часть, которая останется в конечном итоге?

Сжатое измерение сигнала как раз и позволяет одновременно собирать и сжимать данные. Ключевой результат состоит в том, что случайное линейное преобразование может сжать $\mathbf{x}$ без потери информации. Количество необходимых измерений имеет порядок $s\log(d)$. Смысл в том, что мы собираем лишь важную информацию о сигнале. Ниже мы увидим, что расплачиваться за это приходится более медленной реконструкцией. В некоторых ситуациях имеет смысл сэкономить на времени сжатия даже такой ценой. Например, камера наблюдения должна получать и сжимать большой объем зрительной информации, хотя как правило декодировать ее вообще никогда не приходится. Да и во многих других практических приложениях сжатие посредством линейного преобразования выгодно, поскольку может быть эффективно реализовано аппаратно. Например, коллектив под руководством Баранюка (Baraniuk) и Келли (Kelly) предложил архитектуру камеры, в которой используется матрица цифровых микрозеркал для выполнения оптических вычислений линейного преобразования изображения. В таком случае для получения каждого сжатого измерения достаточно выполнить всего одно физическое измерение. Еще одно важное применение сжатого измерения сигнала – медицинские изображения, поскольку чем меньше измерений, тем меньше получаемая пациентом доза облучения.

Неформально говоря, основная предпосылка сжатого измерения сигнала – следующие три «удивительных» результата:

1) любой разреженный сигнал можно точно реконструировать, если он был сжат преобразованием $\mathbf{x} \mapsto W\mathbf{x}$, где W – матрица, обладающая свойством ограниченной изопериметрии (Restricted Isoperimetric Property – RIP). Такая матрица гарантированно дает малое искажение нормы любого вектора, имеющего разреженное представление;
2) реконструкцию можно вычислить за полиномиальное время с помощью решения линейной программы;
3) случайная матрица размера $n \times d$ с высокой вероятностью удовлетворяет условию RIP, если n по порядку величины больше или равно $s\log(d)$.

Приведем формальное определение.

Определение 23.5 (свойство ограниченной изопериметрии). Говорят, что матрица $W \in \mathbb{R}^{n,d}$ обладает свойством (ϵ, s)-RIP, если для любого $\mathbf{x} \neq 0$ такого, что $\|\mathbf{x}\|_0 \leq s$, выполняется неравенство

$$\left| \frac{\|W\mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} - 1 \right| \leq \epsilon.$$

Первая теорема устанавливает тот факт, что RIP-матрицы дают метод сжатия разреженных векторов без потери информации. Она также предлагает (неэффективную) схему реконструкции.

Теорема 23.6. *Пусть $\epsilon < 1$ и W – матрица, обладающая свойством $(\epsilon, 2s)$-RIP. Пусть $\mathbf{x}$ – такой вектор, что $\|\mathbf{x}\|_0 \leq s$, $\mathbf{y} = W\mathbf{x}$ – результат сжатия $\boldsymbol{x}$ и*

$$\tilde{\mathbf{x}} \in \operatorname*{argmin}_{\mathbf{v}: W\mathbf{v} = \mathbf{y}} \|\mathbf{v}\|_0 -$$

реконструированный вектор. Тогда $\tilde{\mathbf{x}} = \mathbf{x}$.

Доказательство. Предположим противное – что $\tilde{\mathbf{x}} \neq \mathbf{x}$. Поскольку $\mathbf{x}$ удовлетворяет ограничениям задачи оптимизации, определяющей $\tilde{\mathbf{x}}$, то очевидно, что $\|\tilde{\mathbf{x}}\|_0 \leq \|\mathbf{x}\|_0 \leq s$. Поэтому $\|\mathbf{x} - \tilde{\mathbf{x}}\|_0 \leq 2s$, и мы можем применить неравенство RIP к вектору $\mathbf{x} - \tilde{\mathbf{x}}$. Но, поскольку $W(\mathbf{x} - \tilde{\mathbf{x}}) = \mathbf{0}$, то получается, что $|0 - 1| \leq \epsilon$. Мы пришли к противоречию. □

Схема реконструкции, предлагаемая теоремой 23.6, кажется неэффективной, поскольку мы должны минимизировать комбинаторную целевую функцию (разреженность $\mathbf{v}$). Тем удивительнее тот факт, что мы можем заменить комбинаторную целевую функцию $\|\mathbf{v}\|_0$ выпуклой функцией $\|\mathbf{v}\|_1$, и таким образом прийти к задаче линейного программирования, которую можно решить эффективно. Формализуем это в следующей теореме.

Теорема 23.7. *Предположим, что выполняются условия теоремы 23.6 и что $\epsilon < 1/(1 + \sqrt{2})$. Тогда*

$$\mathbf{x} = \operatorname*{argmin}_{\mathbf{v}: W\mathbf{v} = \mathbf{y}} \|\mathbf{v}\|_0 = \operatorname*{argmin}_{\mathbf{v}: W\mathbf{v} = \mathbf{y}} \|\mathbf{v}\|_1.$$

На самом деле мы докажем более сильный результат, который остается справедливым, даже если $\mathbf{x}$ – не разреженный вектор.

Теорема 23.8. *Пусть* $\epsilon < 1/(1+\sqrt{2})$ *и* W *– матрица, обладающая свойством* $(\epsilon, 2s)$*-RIP. Пусть* $\mathbf{x}$ *– произвольный вектор. Обозначим*

$$\mathbf{x}_s \in \underset{\mathbf{v}:\|\mathbf{v}\|_0 \le s}{\operatorname{argmin}} \|\mathbf{x}-\mathbf{v}\|_1,$$

то есть $\mathbf{x}_s$ *– вектор, совпадающий с* $\mathbf{x}$ *на* s *самых больших элементах* $\mathbf{x}$*, а все остальные его элементы равны 0. Обозначим* $\mathbf{y} = W\mathbf{x}$ *результат сжатия* $\mathbf{x}$*, и пусть*

$$\mathbf{x}^* \in \underset{\mathbf{v}:W\mathbf{v}=\mathbf{y}}{\operatorname{argmin}} \|\mathbf{v}\|_1 -$$

реконструированный вектор. Тогда

$$\|\mathbf{x}^* - \mathbf{x}\|_2 \le 2\frac{1+\rho}{1-\rho} s^{-1/2} \|\mathbf{x}-\mathbf{x}_s\|_1,$$

где $\rho = \sqrt{2}\epsilon / (1-\epsilon)$.

Заметим, что в частном случае, когда $\mathbf{x} = \mathbf{x}_s$, мы получаем точную реконструкцию $\mathbf{x}^* = \mathbf{x}$, т. е. теорема 23.7 – частный случай теоремы 23.8. Доказательство теоремы 23.8 приведено в разделе 23.3.1.

Наконец, третий результат устанавливает, что случайные матрицы, для которых $n \ge \Omega(s\log(d))$, с большой вероятностью обладают свойством RIP. Фактически теорема показывает, что произведение случайной матрицы на ортогональную тоже дает RIP-матрицу. Это важно для сжатия сигналов вида $\mathbf{x} = U\boldsymbol{\alpha}$, где $\mathbf{x}$ – не разреженный вектор, зато $\boldsymbol{\alpha}$ – разреженный. В таком случае, если W – случайная матрица и мы сжимаем с помощью преобразования $\mathbf{y} = W\mathbf{x}$, то результат будет таким же, как при сжатии $\boldsymbol{\alpha}$ с помощью преобразования $\mathbf{y} = (WU)\boldsymbol{\alpha}$, а т. к. WU – тоже RIP-матрица, то мы можем реконструировать $\boldsymbol{\alpha}$ (а значит, и $\mathbf{x}$) по $\mathbf{y}$.

Теорема 23.9. *Пусть* U *– произвольная фиксированная ортогональная матрица размера* $d \times d$ *и пусть* ϵ, δ *– скаляры в интервале* $(0, 1)$, s *– целое число из множества* $[d]$, *а* n *– целое число, удовлетворяющее неравенству*

$$n \ge 100\frac{s\log(40d/(\delta\epsilon))}{\epsilon^2}.$$

Пусть $W \in \mathbb{R}^{n,d}$ *– матрица, все элементы которой имеют нормальное распределение с нулевым средним и дисперсией* $1/n$*. Тогда с вероятностью не меньше* $1-\delta$ *матрица* WU *будет обладать свойством* (ϵ, s)*-RIP.*

23.3.1. Доказательства*

Доказательство теоремы 23.8

Мы будем следовать работе Candès (2008).

Пусть $\mathbf{h} = \mathbf{x}^* - \mathbf{x}$. Если дан вектор $\mathbf{v}$ и множество индексов I, то обозначим $\mathbf{v}_I$ вектор, в котором i-й элемент равен v_i, если $i \in I$, и 0 в противном случае.

Для начала разобьем множество индексов $[d] = \{1, \ldots, d\}$ на непересекающиеся множества размера s, т. е. $[d] = T_0 \cup T_1 \cup T_2 \ldots T_{d/s-1}$, где для любого i $|T_i| = s$, и для простоты будем предполагать, что d нацело делится на s. Определим это разбиение следующим образом. В T_0 мы помещаем s индексов, соответствующих s наибольшим по абсолютному значению элементам $\mathbf{x}$ (неоднозначности разрешаются произвольным образом). Пусть $T_0^c = [d] \setminus T_0$. Далее в T_1 включим индексы, соответствующие s наибольшим по абсолютному значению элементам $\mathbf{h}_{T_0^c}$. Обозначим $T_{0,1} = T_0 \cup T_1$ и $T_{0,1}^c = [d] \setminus T_{0,1}$. В T_2 включим индексы, соответствующие s наибольшим по абсолютному значению элементам $\mathbf{h}_{T_{0,1}^c}$. Множества T_3, T_4, ... строятся аналогично.

Для доказательства теоремы нам понадобится следующая лемма, которая показывает, что из свойства RIP вытекает приближенная ортогональность.

Лемма 23.10. *Пусть матрица W обладает свойством $(\epsilon, 2s)$-RIP. Тогда для любых двух непересекающихся множеств I, J размера не больше s каждое и для любого вектора $\mathbf{u}$ имеет место неравенство $\langle W\mathbf{u}_I, W\mathbf{u}_J \rangle \le \epsilon \|\mathbf{u}_I\|_2 \|\mathbf{u}_J\|_2$.*

Доказательство. Без ограничения общности можно предположить, что $\|\mathbf{u}_I\|_2 = \|\mathbf{u}_J\|_2 = 1$.

$$\langle W\mathbf{u}_I, W\mathbf{u}_J \rangle \frac{\|W\mathbf{u}_I + W\mathbf{u}_J\|_2^2 - \|W\mathbf{u}_I - W\mathbf{u}_J\|_2^2}{4}.$$

Но, поскольку $|J \cup I| \le 2s$, то из свойства RIP получаем, что $\|W\mathbf{u}_I + W\mathbf{u}_J\|_2^2 \le (1+\epsilon)(\|\mathbf{u}_I\|_2^2 + \|\mathbf{u}_J\|_2^2) = 2(1+\epsilon)$ и что $-\|W\mathbf{u}_I - W\mathbf{u}_J\|_2^2 \le -(1-\epsilon)(\|\mathbf{u}_I\|_2^2 + \|\mathbf{u}_J\|_2^2) = -2(1-\epsilon)$. Что и требовалось доказать. □

Теперь мы готовы доказать теорему. Очевидно, что

$$\|\mathbf{h}\|_2 = \|\mathbf{h}_{T_{0,1}} + \mathbf{h}_{T_{0,1}^c}\|_2 \le \|\mathbf{h}_{T_{0,1}}\|_2 + \|\mathbf{h}_{T_{0,1}^c}\|_2. \tag{23.5}$$

Для доказательства теоремы мы докажем следующие два утверждения:

Утверждение 1: $\|\mathbf{h}_{T_{0,1}^c}\|_2 \le \|\mathbf{h}_{T_0}\|_2 + 2s^{-1/2}\|\mathbf{x} - \mathbf{x}_s\|_1$.

Утверждение 2: $\|\mathbf{h}_{T_{0,1}}\|_2 \le \frac{2\rho}{1-\rho} s^{-1/2}\|\mathbf{x} - \mathbf{x}_s\|_1$.

Объединяя эти два утверждения с неравенством (23.5), получаем:

$$\begin{aligned}\|\mathbf{h}\|_2 &\le \|\mathbf{h}_{T_{0,1}}\|_2 + \|\mathbf{h}_{T_{0,1}^c}\|_2 \le 2\|\mathbf{h}_{T_{0,1}}\|_2 + 2s^{-1/2}\|\mathbf{x} - \mathbf{x}_s\|_1 \\ &\le 2\left(\frac{2\rho}{1-\rho} + 1\right) s^{-1/2}\|\mathbf{x} - \mathbf{x}_s\|_1 \\ &= 2\frac{1+\rho}{1-\rho} s^{-1/2}\|\mathbf{x} - \mathbf{x}_s\|_1,\end{aligned}$$

что и доказывает теорему. □

Доказательство утверждения 1

Чтобы доказать это утверждение, свойство RIP нам вообще не понадобится, мы будем использовать лишь тот факт, что $\mathbf{x}^*$ минимизирует норму ℓ_1. Возьмем не-

которое $j > 1$. Для любого $i \in T_j$ и $i' \in T_{j-1}$ имеем $|h_i| \le |h_{i'}|$. Следовательно, $||\mathbf{h}_{T_j}||_\infty \le ||\mathbf{h}_{T_{j-1}}||_1/s$. Таким образом,

$$||\mathbf{h}_{T_j}||_2 \le s^{1/2}||\mathbf{h}_{T_j}||_\infty \le s^{-1/2}||\mathbf{h}_{T_{j-1}}||_1.$$

Суммируя по $j = 2, 3, \ldots$ и пользуясь неравенством треугольника, получаем

$$\|\mathbf{h}_{T_{0,1}^c}\|_2 \le \sum_{j\ge 2}\|\mathbf{h}_{T_j}\|_2 \le s^{-1/2}\|\mathbf{h}_{T_0^c}\|_1. \tag{23.6}$$

Далее мы покажем, что норма $||\mathbf{h}_{T_0^c}||_1$ не может быть большой. Действительно, из определения $\mathbf{x}^*$ следует, что $||\mathbf{x}||_1 \ge ||\mathbf{x}^*||_1 = ||\mathbf{x} + \mathbf{h}||_1$. Тогда в силу неравенства треугольника получаем

$$\|\mathbf{x}\|_1 \ge \|\mathbf{x}+\mathbf{h}\|_1 = \sum_{i\in T_0}|x_i + h_i| + \sum_{i\in T_0^c}|x_i + h_i| \ge \|\mathbf{x}_{T_0}\|_1 - \|\mathbf{h}_{T_0}\|_1 + \|\mathbf{h}_{T_0^c}\|_1 - \|\mathbf{x}_{T_0^c}\|_1, \tag{23.7}$$

а т. к. $||\mathbf{x}_{T_0^c}||_1 = ||\mathbf{x} - \mathbf{x}_s||_1 = ||\mathbf{x}||_1 - ||\mathbf{x}_{T_0}||_1$, то

$$||\mathbf{h}_{T_0^c}||_1 \le ||\mathbf{h}_{T_0}||_1 + 2||\mathbf{x}_{T_0^c}||_1. \tag{23.8}$$

Объединяя это с (23.6), получаем

$$||\mathbf{h}_{T_{0,1}^c}||_2 \le s^{-1/2}(||\mathbf{h}_{T_0}||_1 + 2||\mathbf{x}_{T_0^c}||_1) \le ||\mathbf{h}_{T_0}||_2 + 2s^{-1/2}||\mathbf{x}_{T_0^c}||_1,$$

чем и завершается доказательство утверждения 1. □

Доказательство утверждения 2

Для доказательства второго утверждения мы воспользуемся свойством RIP, из которого следует, что

$$(1 - \epsilon)||\mathbf{h}_{T_{0,1}^c}||_2^2 \le ||W\mathbf{h}_{T_{0,1}}||_2^2. \tag{23.9}$$

Поскольку $W\mathbf{h}_{T_{0,1}} = W\mathbf{h} - \sum_{j\ge 2}W\mathbf{h}_{T_j} = -\sum_{j\ge 2}W\mathbf{h}_{T_j}$, имеем

$$\|W\mathbf{h}_{T_{0,1}}\|_2^2 = -\sum_{j\ge 2}\langle W\mathbf{h}_{T_{0,1}}, W\mathbf{h}_{T_j}\rangle = -\sum_{j\ge 2}\langle W\mathbf{h}_{T_0} + W\mathbf{h}_{T_1}, W\mathbf{h}_{T_j}\rangle.$$

Применяя свойство RIP к скалярным произведениям, получаем, что для всех $i \in \{1, 2\}$ и $j \ge 2$ имеет место неравенство

$$|\langle W\mathbf{h}_{T_i}, W\mathbf{h}_{T_j}\rangle| \le \epsilon||\mathbf{h}_{T_i}||_2\,||\mathbf{h}_{T_j}||_2.$$

Из того, что $||\mathbf{h}_{T_0}||_2 + ||\mathbf{h}_{T_1}||_1 \le \sqrt{2}||\mathbf{h}_{T_{0,1}}||_2$, следует

$$\|W\mathbf{h}_{T_{0,1}}\|_2^2 \le \sqrt{2}\epsilon\|\mathbf{h}_{T_{0,1}}\|_2\sum_{j\ge 2}\|\mathbf{h}_{T_j}\|_2.$$

Объединяя это с (23.6) и (23.9), получаем

$$(1 - \epsilon)||\mathbf{h}_{T_{0,1}}||_2^2 \le \sqrt{2}\epsilon||\mathbf{h}_{T_{0,1}}||_2 s^{-1/2}||\mathbf{h}_{T_0^c}||_1.$$

После перегруппировки членов имеем

$$\|\mathbf{h}_{T_{0,1}}\|_2 \le \frac{\sqrt{2}\epsilon}{1-\epsilon} s^{-1/2} \|\mathbf{h}_{T_0^c}\|_1.$$

Наконец, воспользовавшись неравенством (23.8), получаем

$$||\mathbf{h}_{T_{0,1}}||_2 \le \rho s^{-1/2}(||\mathbf{h}_{T_0}||_1 + 2||\mathbf{x}_{T_0^c}||_1) \le \rho||\mathbf{h}_{T_0}||_2 + 2\rho s^{-1/2}||\mathbf{x}_{T_0^c}||_1,$$

но, поскольку $||\mathbf{h}_{T_0}||_2 \le ||\mathbf{h}_{T_{0,1}}||_2$, то

$$\|\mathbf{h}_{T_{0,1}}\|_2 \le \frac{2\rho}{1-\rho} s^{-1/2} \|\mathbf{x}_{T_0^c}\|_1.$$

И на этом доказательство второго утверждения завершается. □

Доказательство теоремы 23.9

В доказательстве теоремы мы следуем подходу, предложенному в работе (Baraniuk, Davenport, DeVore & Wakin, 2008). Идея в том, чтобы объединить лемму Джонсона–Линденштрауса с простым рассуждением о покрытии.

Начнем со свойства покрытия единичного шара.

Лемма 23.11. *Пусть* $\epsilon \in (0, 1)$. *Существует конечное множество* $Q \subset \mathbb{R}^d$ *размера* $|Q| \le (3/\epsilon)^d$ *такое, что*

$$\sup_{\mathbf{x}:||\mathbf{x}||\le 1} \min_{\mathbf{v}\in Q} \|\mathbf{x}-\mathbf{v}\| \le \epsilon.$$

Доказательство. Пусть k – целое число и

$$Q' = \{\mathbf{x} \in \mathbb{R}^d: \forall j \in [d]\ \exists i \in \{-k, -k+1, \ldots, k\} \text{ такое, что } x_j = i/k\}.$$

Очевидно, что $|Q'| = (2k+1)^d$. Возьмем $Q = Q' \cap B_2(1)$, где $B_2(1)$ единичный шар по норме ℓ_2 в $\mathbb{R}^d$. Поскольку точки Q' равномерно распределены по единичному ℓ_∞-шару, размер Q равен размеру Q', умноженному на отношение объемов единичных шаров по норме ℓ_2 и ℓ_∞. Объем ℓ_∞-шара равен 2^d, а объем шара $B_2(1)$ равен

$$\frac{\pi^{d/2}}{\Gamma(1+d/2)}.$$

Для простоты предположим, что d четно, и потому

$$\Gamma(1+d/2) = (d/2)! \ge \left(\frac{d/2}{e}\right)^{d/2},$$

где в последнем неравенстве мы использовали формулу Стирлинга. В итоге мы получили, что

$$|Q| \le (2k+1)^d (\pi/e)^{d/2} (d/2)^{-d/2} 2^{-d}. \qquad (23.10)$$

Теперь определим k. Для любого $\mathbf{x} \in B_2(1)$ пусть $\mathbf{v} \in Q$ – вектор, i-й элемент которого равен $\text{sign}(x_i)\ \lfloor |x_i|\ k \rfloor / k$. Тогда для любого его элемента имеем $|x_i - v_i| \le 1/k$ и значит

$$\|\mathbf{x} - \mathbf{v}\| \le \frac{\sqrt{d}}{k}.$$

Чтобы правая часть была не больше ϵ, мы возьмем $k = \lceil \sqrt{d}/\epsilon \rceil$. Подставляя это значение в (23.10), приходим к выводу, что

$$|Q| \le (3\sqrt{\delta}/(2\epsilon))^d (\pi/\epsilon)^{d/2} (\delta/2)^{-d/2} - \left(\frac{3}{\epsilon}\sqrt{\frac{\pi}{2\epsilon}}\right)^d \le \left(\frac{3}{\epsilon}\right)^d.$$

Что и требовалось доказать. □

Пусть $\mathbf{x}$ – вектор, который можно записать в виде $\mathbf{x} = U\boldsymbol{\alpha}$, где U – ортогональная матрица и $\|\boldsymbol{\alpha}\|_0 \le s$. Объединяя доказанное выше свойство покрытия с леммой Джонсона–Линденштрауса (лемма 23.4), мы сможем показать, что случайная матрица W не искажает такой вектор $\mathbf{x}$.

Лемма 23.12. *Пусть U – ортогональная матрица размера $d \times d$ и $I \subset [d]$ – множество индексов размера $|I| = s$. Пусть S – линейная оболочка векторов $\{U_i : i \in I\}$, где U_i – i-й столбец U. Пусть $\delta \in (0, 1)$, $\epsilon \in (0, 1)$ и $n \in \mathbb{N}$ выбраны так, что*

$$n \ge 24 \frac{\log(2/d) + s\log(12/\epsilon)}{\epsilon^2}.$$

Тогда с вероятностью не менее $1 - \delta$ для случайной матрицы $W \in \mathbb{R}^{n,d}$, все элементы которой независимо выбраны из распределения $N(0, 1/n)$, имеем

$$\sup_{\mathbf{x} \in S} \left| \frac{\|W\mathbf{x}\|}{\|\mathbf{x}\|} - 1 \right| < \epsilon.$$

Доказательство. Достаточно доказать лемму для всех $\mathbf{x} \in S$ с $\|\mathbf{x}\| = 1$. Мы можем записать $\mathbf{x} = U_I\boldsymbol{\alpha}$, где $\boldsymbol{\alpha} \in \mathbb{R}^s$, $\|\boldsymbol{\alpha}\|_2 = 1$ и U_I – матрица со столбцами $\{U_i : i \in I\}$. Из леммы 23.11 мы знаем, что существует множество Q размера $|Q| \le (12/\epsilon)^s$ такое, что

$$\sup_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|=1} \min_{\mathbf{v} \in Q} \|\boldsymbol{\alpha} - \mathbf{v}\| \le (\epsilon/4).$$

Но, поскольку U ортогональна, мы также имеем

$$\sup_{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|=1} \min_{\mathbf{v} \in Q} \|U_I\boldsymbol{\alpha} - U_I\mathbf{v}\| \le (\epsilon/4).$$

Применяя лемму 23.4 к множеству $\{U_I\mathbf{v} : \mathbf{v} \in Q\}$, получаем, что для n, удовлетворяющих условию леммы, следующее неравенство имеет место с вероятностью не менее $1 - \delta$:

$$\sup_{\mathbf{v} \in Q} \left| \frac{\|WU_I\mathbf{v}\|^2}{\|U_I\mathbf{v}\|^2} - 1 \right| \le \epsilon/2.$$

Отсюда также следует, что

$$\sup_{\mathbf{v}\in Q}\left|\frac{\|WU_I\mathbf{v}\|}{\|U_I\mathbf{v}\|}-1\right|\le\epsilon/2.$$

Пусть a – наименьшее число такое, что

$$\forall\mathbf{x}\in S,\ \frac{\|W\mathbf{x}\|}{\|\mathbf{x}\|}\le 1+a.$$

Очевидно, что $a < \infty$. Наша цель – показать, что $a \le \epsilon$. Это следствие того факта, что для любого вектора $\mathbf{x}\in S$ единичной нормы существует $\mathbf{v}\in Q$ такой, что $||\mathbf{x} - U_I\mathbf{v}|| \le /4$ и, следовательно,

$$||W\mathbf{x}|| \le ||WU_I\mathbf{v}|| + ||W(\mathbf{x} - U_I\mathbf{v})|| \le 1 + \epsilon/2 + (1 + a)\epsilon/4.$$

Таким образом

$$\forall\mathbf{x}\in S,\ \frac{\|W\mathbf{x}\|}{\|\mathbf{x}\|}\le 1+(\epsilon/2+(1+a)\epsilon/4).$$

Но из определения a следует, что

$$a\le\epsilon/2+(1+a)\epsilon/4\Rightarrow a\le\frac{\epsilon/2+\epsilon/4}{1-\epsilon/4}\le\epsilon.$$

Тем самым доказано, что для любого $\mathbf{x}\in S$ имеет место $||W\mathbf{x}||/||\mathbf{x}|| - 1 \le \epsilon$. Отсюда следует и вторая часть неравенства, потому что

$$||W\mathbf{x}|| \ge ||WU_I\mathbf{v}|| - ||W(\mathbf{x} - U_I\mathbf{v})|| \ge 1 - \epsilon/2 - (1 + \epsilon)\epsilon/4 \ge 1 - \epsilon. \qquad \square$$

Доказанная только что лемма означает, что для любого вектора $\mathbf{x}\in S$ единичной нормы имеет место неравенство

$$(1 - \epsilon) \le ||W\mathbf{x}|| \le (1 + \epsilon).$$

Откуда следует, что

$$(1 - 2\epsilon) \le ||W\mathbf{x}||^2 \le (1 + 3\epsilon).$$

Теперь теорема 23.9 вытекает из применения леммы о границе объединения ко всем элементам I.

23.4. PCA или сжатое измерение сигнала?

Допустим, мы хотели бы применить какой-нибудь метод понижения размерности к заданному набору примеров. Что выбрать: PCA или сжатое измерение сигнала? В этом разделе мы обсудим этот вопрос, проанализировав предположения, лежащие в основе обоих методов.

Для начала неплохо бы понимать, при каких условиях тот и другой метод гарантируют идеальное восстановление. PCA дает такую гарантию, когда множе-

ство примеров содержится в n-мерном подпространстве $\mathbb{R}^d$, а сжатое измерение сигнала – когда множество примеров разрежено (в некотором базисе). Опираясь на эти наблюдения, мы можем описать ситуации, в которых PCA будет лучше сжатого измерения сигнала и наоборот.

Первым рассмотрим случай, когда примерами являются векторы стандартного базиса $\mathbb{R}^d$ – $\mathbf{e}_1, \ldots, \mathbf{e}_d$, где каждый вектор $\mathbf{e}_i$ содержит 1 в i-й координате и 0 во всех остальных. Здесь примеры 1-разрежены. Поэтому сжатое измерение сигнала даст идеальное восстановление, если $n \geq \Omega(\log(d))$. С другой стороны, PCA покажет плохое качество, т. к. данные отнюдь не находятся в n-мерном подпространстве ни при каком $n < d$. В самом деле, легко проверить, что в этом случае средняя ошибка реконструкции PCA (т. е. значение целевой функции (23.1), поделенное на m) равна $(d - n)/d$, что больше 1/2 при $n \leq d/2$.

Далее продемонстрируем случай, когда PCA лучше сжатого измерения сигнала. Рассмотрим m примеров, расположенных в n-мерном подпространстве. Очевидно, что в таком случае PCA дает точное восстановление. Что же до сжатого измерения сигнала, то заметим, что примеры n-разрежены в любом ортонормированном базисе, на первые n вектором которого натянуто подпространство. Поэтому сжатое измерение также будет работать, если размерность понижается до $\Omega(n\log(d))$. Но если размерность в точности равна n, то сжатое измерение может дать неудовлетворительные результаты. Кроме того, PCA более устойчив к некоторым типам шумов. Обсуждение этого вопроса см. в работе Chang, Weiss & Freeman, 2009.

23.5. Резюме

Мы познакомились с двумя методами понижения размерности с помощью линейных преобразований: PCA и случайное проецирование. Мы показали, что PCA оптимален в смысле среднеквадратичной ошибки реконструкции, если ограничиться только линейными реконструкциями. Но если допустить нелинейную реконструкцию, то PCA уже необязательно будет оптимальной процедурой. В частности, для разреженных данных случайное проецирование может давать гораздо лучшие результаты, чем PCA. В этом и состоит суть метода сжатого измерения сигнала.

ЛАНЬ®

23.6. Библиографические сведения

Метод PCA эквивалентен наилучшей аппроксимации подпространства с помощью сингулярного разложения (singular value decomposition – SVD). Метод SVD описан в приложении C и восходит к работам Эудженио Бельтрами (1873) и Камилла Жордана (1874). Он много раз переоткрывался. В литературу по статистике его ввел Пирсон (Pearson, 1901). Помимо PCA и SVD, есть и другие названия, основанные на той же идее и используемые в различных научных сообществах, например: теорема Эккарта–Янга (в честь Карла Эккарта (Carl Eckart) и Гэйла Янга (Gale Young)), которые анализировали этот метод в 1936 г.), теорема Шмидта–Мирского, факторный анализ и преобразование Хотеллинга.

Сжатое измерение сигнала описано в работах Donoho (2006) и Candes & Tao (2005). См. также Candes (2006).

23.7. Упражнения

23.1. В этом упражнении мы покажем, что в общем случае точная реконструкция линейной схемы сжатия невозможна.

1. Пусть $A \in \mathbb{R}^{n,d}$ – произвольная матрица сжатия и $n \leq d - 1$. Покажите, что существуют векторы $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, $\mathbf{u} \neq \mathbf{v}$ такие, что $A\mathbf{u} = A\mathbf{v}$.
2. Сделайте отсюда вывод, что точная реконструкция линейной схемы сжатия невозможна.

23.2. Пусть вектор $\alpha \in \mathbb{R}^d$ таков, что $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_d \geq 0$. Покажите, что

$$\max_{\beta \in [0,1]^d : \|\beta\|_1 \leq n} \sum_{j=1}^{d} \alpha_j \beta_j = \sum_{j=1}^{n} \alpha_j .$$

Указание. Рассмотрим все векторы $\beta \in [0, 1]^d$ такие, что $\|\beta\|_1 \leq n$. Пусть i – наименьший индекс, для которого $\beta_i < 1$. Если $i = n + 1$, то все доказано. В противном случае покажите, что можно увеличить β_i, быть может, уменьшив при этом β_j для некоторого $j > i$, и получить при этом лучшее решение. Отсюда следует, что оптимальным будет решение, в котором $\beta_i = 1$ для $i \leq n$ и $\beta_i = 0$ для $i > n$.

23.3. Ядерный PCA. В этом упражнении мы покажем, как можно использовать PCA для построения нелинейной схемы понижения размерности на основе ядерного трюка (см. главу 16).

Пусть $\mathcal{X}$ – пространство образцов, и $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ – множество точек в $\mathcal{X}$. Рассмотрим отображение в пространство признаков $\psi : \mathcal{X} \to V$, где V – некоторое гильбертово пространство (возможно, бесконечномерное). Пусть $K : \mathcal{X} \times \mathcal{X}$ – ядерная функция, т. е. $k(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$. Ядерный PCA заключается в отображении элементов S в V функцией ψ и последующем применении PCA для отображения множества $\{\psi(\mathbf{x}_1), \ldots, \psi(\mathbf{x}_m)\}$ в $\mathbb{R}^n$. Результатом этой процедуры будет множество элементов пониженной размерности.

Покажите, как выполнить эту процедуру за время, полиномиально зависящее от m и n, в предположении, что для вычисления $K(\cdot, \cdot)$ требуется постоянное время. Если в вашей реализации используется умножение двух матриц A и B, убедитесь, что их произведение можно вычислить. Аналогично, если требуется спектральное разложение некоторой матрицы C, проверьте, что его вычисление возможно.

23.4. Интерпретация PCA как максимизации дисперсии. Пусть $\mathbf{x}_1, \ldots, \mathbf{x}_m$ – m векторов в $\mathbb{R}^d$ и пусть $\mathbf{x}$ – случайный вектор, выбранный из $\mathbf{x}_1, \ldots, \mathbf{x}_m$ согласно равномерному распределению. Предположим, что $\mathbb{E}[\mathbf{x}] = \mathbf{0}$.

1. Рассмотрим задачу о нахождении единичного вектора $\mathbf{w} \in \mathbb{R}^d$, для которого случайная величина $\langle \mathbf{w}, \mathbf{x} \rangle$ имеет максимальную дисперсию. То есть мы хотим решить задачу

$$\arg\max_{\mathbf{w}:\|\mathbf{w}\|=1} \mathrm{Var}[\langle \mathbf{w}, \mathbf{x} \rangle] = \arg\max_{\mathbf{w}:\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle)^2 .$$

Покажите, что для решения задачи $\mathbf{w}$ должен быть первой главной компонентой $\mathbf{x}_1, \ldots, \mathbf{x}_m$.

2. Пусть $\mathbf{w}_1$ – первая главная компонента, как в предыдущем вопросе. Предположим, что мы хотим найти второй единичный вектор $\mathbf{w}_2 \in \mathbb{R}^d$, который максимизирует дисперсию $\langle \mathbf{w}_2, \mathbf{x} \rangle$, но при этом не коррелирован с $\langle \mathbf{w}_1, \mathbf{x} \rangle$. То есть требуется решить задачу оптимизации

$$\operatorname*{argmax}_{\mathbf{w}:||\mathbf{w}||=1,\ \mathbb{E}[(\langle \mathbf{w}_1, \mathbf{x}\rangle)(\langle \mathbf{w}, \mathbf{x}\rangle)]=0} \operatorname{Var}[\langle \mathbf{w}, \mathbf{x} \rangle].$$

Покажите, что для решения задачи $\mathbf{w}$ должен быть второй главной компонентой $\mathbf{x}_1, \ldots, \mathbf{x}_m$.
Указание. Заметьте, что

$$\mathbb{E}[(\langle \mathbf{w}_1, \mathbf{x} \rangle)(\langle \mathbf{w}, \mathbf{x} \rangle)] = \mathbf{w}_1^\top \mathbb{E}[\mathbf{x}\mathbf{x}^\top]\mathbf{w} = m\mathbf{w}_1^\top A \mathbf{w},$$

где $A = \sum_i \mathbf{x}_i \mathbf{x}_i^\top$. Поскольку $\mathbf{w}$ – собственный вектор A, ограничение $\mathbb{E}[(\langle \mathbf{w}_1, \mathbf{x} \rangle)(\langle \mathbf{w}, \mathbf{x} \rangle)] = 0$ эквивалентно ограничению

$$\langle \mathbf{w}_1, \mathbf{w} \rangle = 0.$$

23.5. Связь между SVD и PCA. Воспользуйтесь теоремой о SVD (следствие C.6) для вывода другого доказательства теоремы 23.2.

23.6. Случайная проекция сохраняет скалярные произведения. Лемма Джонсона–Линденштрауса говорит, что случайная проекция сохраняет расстояния между векторами из конечного множества. В этом упражнении вам нужно будет доказать, что если множество векторов содержится в единичном шаре, то сохраняются не только расстояния между любыми двумя векторами, но и скалярное произведение векторов.

Пусть Q – конечное множество векторов в $\mathbb{R}^d$ и предположим, что для любого $\mathbf{x} \in Q$ имеет место $||\mathbf{x}|| \leq 1$.

1. Пусть $\delta \in (0, 1)$ и целое число n таковы, что

$$\epsilon = \sqrt{\frac{6 \log(|Q|^2/\delta)}{n}} \leq 3.$$

Докажите, что с вероятностью не менее $1 - \delta$ для случайно выбранной матрицы $W \in \mathbb{R}^{n,d}$, все элементы которой независимы и имеют нормальное распределение $\mathcal{N}(0, 1/n)$, для любых $\mathbf{u}, \mathbf{v} \in Q$ имеет место неравенство

$$|\langle W\mathbf{u}, W\mathbf{v} \rangle - \langle \mathbf{u}, \mathbf{v} \rangle| \leq \epsilon.$$

Указание. Воспользуйтесь леммой Джонсона–Линденштрауса, чтобы ограничит, m сверху $(||W(\mathbf{u}+\mathbf{v})||)/(||\mathbf{u}+\mathbf{v}||)$ и $(||W(\mathbf{u}-\mathbf{v})||)/(||\mathbf{u}-\mathbf{v}||)$.

2. **(*)** Пусть $\mathbf{x}_1, \ldots, \mathbf{x}_m$ – множество векторов в $\mathbb{R}^d$ с нормой не больше 1 и предположим, что эти векторы линейно разделимы с зазором γ. Предположим еще, что $d \gg 1/\gamma^2$. Покажите, что существует константа $c > 0$ такая, что если случайным образом спроецировать эти векторы на $\mathbb{R}^n$ для $n = c/\gamma^2$, то с вероятностью не менее 99% спроецированные векторы будут линейно разделимы с зазором $\gamma/2$.

Глава 24

ПОРОЖДАЮЩИЕ МОДЕЛИ

ЛАНЬ®

Мы начали эту книгу с рассмотрения системы обучения, *не зависящей от распределения*, т. е. не делали никаких предположений об истинном распределении данных. Кроме того, мы применяли *дискриминантный* подход, цель которого – не обучиться истинному распределению, а обучить достаточно верный предиктор. В этой главе мы опишем *порождающий* подход, когда предполагается, что истинное распределение данных имеет определенную параметрическую форму, и наша цель – оценить параметры модели. Эта задача называется *параметрическим оцениванием плотности распределения*.

Преимуществом дискриминантного подхода является прямая оптимизация представляющей интерес величины (верности предсказания) вместо обучения истинному распределению. Эта мысль была сформулирована Владимиром Вапником в его принципе решения задач в условиях ограниченного объема информации:

ЛАНЬ®

Решая поставленную задачу, старайтесь избегать решения более общей задачи в качестве промежуточного шага.

Разумеется, если нам удастся верно обучиться истинному распределению, мы будем считаться «экспертами» в том смысле, что сможем предсказывать, используя оптимальный байесовский классификатор. Проблема в том, что обычно обучиться истинному распределению обычно труднее, чем обучить верный предиктор. Но в некоторых ситуациях имеет смысл принять порождающий подход к обучению. Например, иногда проще (с вычислительной точки зрения) оценить параметры модели, чем обучить дискриминантный предиктор. Кроме того, в некоторых случаях никакой конкретной задачи не поставлено, но мы просто хотели бы построить модель данных либо для того, чтобы делать предсказания впоследствии, не переобучая предиктор, либо чтобы иметь возможность интерпретировать данные.

Мы начнем с популярного статистического метода оценивания параметров модели – принципа максимального правдоподобия. Мы также опишем EM-алгоритм, который вычисляет максимальное правдоподобие при наличии скрытых (латентных) переменных. И завершим главу кратким описанием байесовских рассуждений.

24.1. Оценка максимального правдоподобия

Начнем с простого примера. Фармацевтическая компания разработала новое лекарство для лечения смертельной болезни. Мы хотим оценить вероятность выживания при приеме этого лекарства. Для этого компания выбрала обучающий набор из m человек, согласившихся принимать лекарство. Обозначим $S = (x_1, \ldots, x_m)$ обучающий набор, в котором $x_i = 1$, если i-й человек выжил, и $x_i = 0$ в противном случае. Мы можем смоделировать истинное распределение с помощью одного параметра $\theta \in [0, 1]$, описывающего вероятность выживания.

Теперь хотелось бы оценить параметр θ на основе обучающего набора S. Естественно возникает идея использовать в качестве оценки среднее количество единиц в S, т. е.

$$\hat{\theta} = \frac{1}{m}\sum_{i=1}^{m} x_i. \tag{24.1}$$

Очевидно, что $\mathbb{E}_S[\hat{\theta}] = \theta$, т. е. $\hat{\theta}$ – *несмещенная оценка* θ. Кроме того, поскольку $\hat{\theta}$ – среднее m независимых и одинаково распределенных случайных величин, то, согласно неравенству Хёфдинга, с вероятностью не менее $1 - \delta$ для случайной выборки S справедливо неравенство

$$|\hat{\theta} - \theta| \le \sqrt{\frac{\log(2/\delta)}{2m}}. \tag{24.2}$$

Еще одна интерпретация $\hat{\theta}$ – *оценка максимального правдоподобия*, которую мы сейчас формально объясним. Сначала запишем вероятность генерации выборки S:

$$\mathbb{P}[S = (x_1, \ldots, x_m)] = \prod_{i=1}^{m} \theta^{x_i}(1-\theta)^{1-x_i} = \theta^{\sum_i x_i}(1-\theta)^{\sum_i (1-x_i)}.$$

Определим *логарифмическое правдоподобие* S с параметром θ как логарифм этого выражения:

$$L(S;\theta) = \log(\mathbb{P}[S = (x_1, \ldots, x_m)]) = \log(\theta)\sum_i x_i + \log(1-\theta)\sum_i (1-x_i).$$

Оценка максимального правдоподобия – это такой параметр, который доставляет максимум правдоподобию:

$$\hat{\theta} \in \operatorname*{argmax}_{\theta} L(S;\theta). \tag{24.3}$$

Далее мы покажем, что в нашем случае выражение (24.1) является оценкой максимального правдоподобия. Чтобы убедиться в этом, приравняем нулю производную $L(S; \theta)$ по θ:

$$\frac{\sum_i x_i}{\theta} - \frac{\sum_i (1-x_i)}{1-\theta} = 0.$$

Решив это уравнение относительно θ, получим оценку (24.1).

24.1.1. Оценка максимального правдоподобия для непрерывных случайных величин

Пусть X – непрерывная случайная величина. Тогда для большинства $x \in \mathbb{R}$ имеем $P[X = x] = 0$, и потому данное выше определение правдоподобия оказывается тривиальным. Для преодоления этой технической трудности мы определим правдоподобие как логарифм *плотности* вероятности X в точке x, т. е. если дана обучающая выборка $S = (x_1, \ldots, x_m)$ примеров, независимо выбранных из распределения с плотностью $\mathcal{P}_\theta$, то правдоподобие S при заданном θ определяется как

$$L(S;\theta) = \log\left(\prod_{i=1}^{m} \mathcal{P}_\theta(x_i)\right) = \sum_{i=1}^{m} \log(\mathcal{P}_\theta(x_i)).$$

Как и раньше, оценкой максимального правдоподобия называется значение θ, доставляющее максимум $L(S; \theta)$.

В качестве примера рассмотрим нормальную случайную величину X, функция плотности вероятности которой зависит от параметра $\theta = (\mu, \sigma)$ и определена следующим образом:

$$\mathcal{P}_\theta = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

Мы можем переписать правдоподобие в виде

$$L(S;\theta) = -\frac{1}{2\sigma^2}\sum_{i=1}^{m}(x_i - m)^2 - m\log(\sigma\sqrt{2\pi}).$$

Чтобы найти параметр $\theta = (\mu, \sigma)$, оптимизирующий это выражение, приравняем нулю производные по μ и по σ. Получаем два уравнения:

$$\frac{d}{d\mu} L(S;\theta) = \frac{1}{\sigma^2}\sum_{i=1}^{m}(x_i - m) = 0;$$

$$\frac{d}{d\sigma} L(S;\theta) = \frac{1}{\sigma^3}\sum_{i=1}^{m}(x_i - m)^2 - \frac{m}{\sigma} = 0.$$

Решая их, получаем оценки максимального правдоподобия:

$$\hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{и} \quad \hat{\sigma} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_i - \hat{\mu})^2}.$$

Заметим, что оценка максимального правдоподобия не всегда является несмещенной. Например, оценка $\hat{\mu}$ несмещенная, но можно показать, что оценка дисперсии $\hat{\sigma}$ таковой не является (упражнение 24.1).

Упрощенная нотация

Для упрощения нотации мы в этой главе будем обозначать $\mathcal{P}[X = x]$ как вероятность того, что $X = x$ (для дискретных случайных величин), так и плотность распределения в точке x (для непрерывных величин).

24.1.2. Максимальное правдоподобие и минимизация эмпирического риска

Оценка максимального правдоподобия связана с принципом минимизации эмпирического риска (ERM), который мы всесторонне изучали в предыдущих главах. Напомним, что в принципе ERM имеется класс гипотез $\mathcal{H}$, и мы используем обучающий набор, чтобы выбрать гипотезу $h \in \mathcal{H}$, минимизирующую эмпирический риск. Покажем теперь, что оценка максимального правдоподобия – это ERM для определенной функции потерь.

Если дан параметр θ и наблюдение x, то определим потерю, соответствующую θ, в точке x как

$$\ell(\theta, x) = -\log(\mathcal{P}_\theta[x]). \tag{24.4}$$

То есть $\ell(\theta, x)$ – логарифмическое правдоподобие точке x со знаком минус в предположении, что данные имеют распределение $\mathcal{P}_\theta$. Эту функцию потерь часто называют логарифмической потерей. Из этого определения сразу следует, что принцип максимального правдоподобия эквивалентен минимизации эмпирического риска относительно функции потерь (24.4), т. е.

$$\operatorname*{argmin}_\theta \sum_{i=1}^{m} (-\log(\mathcal{P}_\theta[x_i])) = \operatorname*{argmin}_\theta \sum_{i=1}^{m} \log(\mathcal{P}_\theta[x_i]).$$

Если данные действительно имеют распределение $\mathcal{P}$ (не обязательно той параметрической формы, которую мы предполагаем), то истинный риск параметра θ принимает вид:

$$\begin{aligned} \mathop{\mathbb{E}}_{x}[\ell(\theta, x)] &= -\sum_{i=1} \mathcal{P}[x] \log(\mathcal{P}_\theta[x]) \\ &= \underbrace{\sum_x \mathcal{P}[x] \log\left(\frac{\mathcal{P}[x]}{\mathcal{P}_\theta[x]}\right)}_{D_{\mathrm{RE}}[\mathcal{P} \| \mathcal{P}_\theta]} + \underbrace{\sum_x \mathcal{P}[x] \log\left(\frac{1}{\mathcal{P}[x]}\right)}_{H(\mathcal{P})}. \end{aligned} \tag{24.5}$$

Здесь D_{RE} называется *относительной энтропией*, а H – *функцией энтропии*. Относительная энтропия измеряет расхождение между двумя распределениями вероятностей. Для дискретных величин она всегда неотрицательна и равна 0 только в том случае, когда два распределения совпадают. Отсюда следует, что истинный риск минимален, когда $\mathcal{P}_\theta = \mathcal{P}$.

Выражение (24.5) показывает, как наше предположение о порождении влияет на оценку плотности, даже в пределе, когда набор данных бесконечен. Из него видно, что если истинное распределение действительно имеет предполагаемую параметрическую форму, то, выбирая правильное значение параметра, мы можем сделать риск энтропией распределения. Но если распределение имеет иную форму, то даже самый лучший параметр дает модель низкого качества, а неоптимальность измеряется относительной энтропией.

24.1.3. Анализ обобщаемости

Насколько хороша оценка максимального правдоподобия при обучении на конечном наборе данных?

Чтобы ответить на этот вопрос, мы должны определить, как измерять качество приближенного решения в задаче об оценке плотности. В отличие от дискриминантного обучения, когда имеется однозначное понятие «потери», в порождающем обучении потерю модели можно определить разными способами. Предыдущий раздел подсказывает одного естественного кандидата: математическое ожидание логарифмической потери (24.5).

В некоторых ситуациях легко доказать, что принцип максимального правдоподобия гарантирует также и низкий риск. Рассмотрим, к примеру, задачу об оценивании среднего значения нормальной случайной величины с единичной дисперсией. Выше мы видели, что оценкой максимального правдоподобия является среднее $\hat{\mu} = (1/m)\sum_i x_i$. Обозначим μ^* оптимальный параметр. Тогда

$$\begin{aligned}
\mathop{\mathbb{E}}_{x\sim N(\mu^*,1)}[\ell(\hat{\mu},x)-\ell(\mu^*,x)] &= \mathop{\mathbb{E}}_{x\sim N(\mu^*,1)}\log\left(\frac{\mathcal{P}_{\mu^*}[x]}{\mathcal{P}_{\hat{\mu}}[x]}\right)\\
&= \mathop{\mathbb{E}}_{x\sim N(\mu^*,1)}\left(-\frac{1}{2}(x-\mu^*)^2+\frac{1}{2}(x-\hat{\mu})^2\right)\\
&= \frac{\hat{\mu}^2}{2}-\frac{(\mu^*)^2}{2}+(\mu^*-\hat{\mu})\mathop{\mathbb{E}}_{x\sim N(\mu^*,1)}[x]\\
&= \frac{\hat{\mu}^2}{2}-\frac{(\mu^*)^2}{2}+(\mu^*-\hat{\mu})\mu^*\\
&= \frac{1}{2}(\hat{\mu}-\mu^*)^2. \qquad (24.6)
\end{aligned}$$

Заметим далее, что $\hat{\mu}$ – среднее m нормальных величин и потому имеет нормальное распределение со средним μ^* и дисперсией σ^*/m. Отсюда можно вывести, что с вероятностью не менее $1-\delta$ справедливо неравенство $|\hat{\mu}-\mu^*| \le \epsilon$, где ϵ зависит от σ^*/m и δ.

Иногда оценка максимально правдоподобия приводит к очевидному переобучению. Рассмотрим, например, случайную величину Бернулли X и пусть $\mathcal{P}[X=1] = \theta^*$. Выше мы видели, что из неравенства Хёфдинга легко вывести гарантию $|\theta^* - \hat{\theta}|$, имеющую место с высокой вероятностью (см. (24.2)). Но если наша цель – получить малое значение ожидаемой логарифмической потери (24.5), то мы можем потерпеть неудачу. Предположим, например, что θ^* отлично от нуля, но очень мало. Тогда вероятность, что ни один элемент выборки размера m не равен 1, равна $(1-\theta^*)^m$, а это больше $e^{-2\theta^* m}$. Отсюда следует, что при $m \le \log(2)/(2\theta^*)$ вероятность того, что выборка состоит из одних нулей, составляет не менее 50%, и в таком случае правило максимального правдоподобия даст $\hat{\theta} = 0$. Но истинный риск оценки $\hat{\theta} = 0$ равен

$$\begin{aligned}
\mathop{\mathbb{E}}_{x\sim\theta^*}[\ell(\hat{\theta},x)] &= \theta^*\ell(\hat{\theta},1)+(1-\theta^*)\ell(\hat{\theta},0)\\
&= \theta^*\log(1/\hat{\theta})+(1-\theta^*)\log(1/(1-\hat{\theta}))\\
&= \theta^*\log(1/0) = \infty.
\end{aligned}$$

Этот простой пример показывает, что применять принцип максимального правдоподобия следует с осторожностью.

Чтобы избежать переобучения, можно использовать различные инструменты, встречавшиеся в этой книге ранее. Простая техника регуляризации описана в упражнении 24.2.

24.2. Наивная байесовская классификация

Наивный байесовский классификатор – классическая демонстрация того, как предположения о порождающем распределении и оценка параметров могут упростить процесс обучения. Рассмотрим проблему предсказания метки $y \in \{0, 1\}$ по вектору признаков $\mathbf{x} = (x_1, \ldots, x_d)$, где предполагается, что каждый элемент x_i равен 0 или 1. Напомним, что оптимальный байесовский классификатор определяется выражением

$$h_{\text{Bayes}}(\mathbf{x}) = \operatorname*{argmax}_{y \in \{0,1\}} \mathcal{P}[Y = y | X = \mathbf{x}].$$

Чтобы описать функцию вероятности $\mathcal{P}[Y = y|X = \mathbf{x}]$, нам нужно 2^d параметров, каждый из которых соответствует условной вероятности $\mathcal{P}[Y = 1|X = \mathbf{x}]$ для некоторого значения $\mathbf{x} \in \{0, 1\}^d$. Отсюда следует, что необходимое число примеров экспоненциально зависит от числа признаков.

В наивном байесовском подходе мы делаем (довольно наивное) предположение о том, что при данной метке признаки не зависят друг от друга, т. е.

$$\mathcal{P}[X = \mathbf{x}|Y = y] = \prod_{i=1}^{d} \mathcal{P}[X_i = x_i|Y = y].$$

При таком предположении из правила Байеса следует, что оптимальный байесовский классификатор можно еще упростить:

$$\begin{aligned} h_{\text{Bayes}}(\mathbf{x}) &= \operatorname*{argmax}_{y \in \{0,1\}} \mathcal{P}[Y = y | X = \mathbf{x}] \\ &= \operatorname*{argmax}_{y \in \{0,1\}} \mathcal{P}[Y = y] \mathcal{P}[X = \mathbf{x}|Y = y] \\ &= \operatorname*{argmax}_{y \in \{0,1\}} \mathcal{P}[Y = y] \prod_{i=1}^{d} \mathcal{P}[X_i = x_i|Y = y]. \end{aligned} \tag{24.7}$$

Это значит, что теперь нам нужно оценить только $2d + 1$ параметров. Сделанное предположение о порождающем распределении позволило значительно уменьшить количество подлежащих обучению параметров.

Если вдобавок параметры оцениваются по принципу максимального правдоподобия, то получающийся классификатор называется *наивным байесовским*.

24.3. Линейный дискриминантный анализ

Линейный дискриминантный анализ (ЛДА) – еще одна демонстрация того, как предположения о порождающем распределении упрощают процесс обучения. Как и в случае наивного байесовского классификатора, мы рассматриваем проблему предсказания метки $y \in \{0, 1\}$ по вектору признаков $\mathbf{x} = (x_1, \ldots, x_d)$. Но теперь

предположение выглядит следующим образом. Во-первых, мы предполагаем, что $\mathcal{P}[Y = 1] = \mathcal{P}[Y = 0] = 1/2$. Во-вторых, предполагается, что условная вероятность X при условии Y имеет нормальное распределение. Наконец, ковариационная матрица нормального распределения одинакова для обоих значений метки. Формально, пусть $\boldsymbol{\mu}_0, \boldsymbol{\mu}_1 \in \mathbb{R}^d$ и Σ – ковариационная матрица. Тогда функция плотности распределения имеет вид

$$\mathcal{P}[X = \mathbf{x}|Y = y] = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_y)^T \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_y)\right).$$

В предыдущем разделе мы показали, что в силу правила Байеса можно написать

$$h_{\text{Bayes}}(\mathbf{x}) = \operatorname*{argmax}_{y \in \{0,1\}} \mathcal{P}[Y = y]\,\mathcal{P}[X = \mathbf{x}|Y = y].$$

Это означает, что мы предсказываем $h_{\text{Bayes}}(\mathbf{x}) = 1$ тогда и только тогда, когда

$$\log\left(\frac{\mathcal{P}[Y = 1]\,\mathcal{P}[X = \mathbf{x}|Y = 1]}{\mathcal{P}[Y = 0]\,\mathcal{P}[X = \mathbf{x}|Y = 0]}\right) > 0.$$

Это отношение часто называют *логарифмическим отношением правдоподобия*. В нашем случае логарифмическое отношение правдоподобия принимает вид

$$½(\mathbf{x} - \boldsymbol{\mu}_0)^T\Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_0) - ½(\mathbf{x} - \boldsymbol{\mu}_1)^T\Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_1).$$

Мы можем переписать это в виде $\langle \mathbf{w}, \mathbf{x} \rangle + b$, где

$$\mathbf{w} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\Sigma^{-1} \quad \text{и} \quad b = ½(\boldsymbol{\mu}_0^T\Sigma^{-1}\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1^T\Sigma^{-1}\boldsymbol{\mu}_1). \tag{24.8}$$

Таким образом, мы получили, что при сделанных предположениях о порождающем распределении оптимальный байесовский классификатор будет линейным. Кроме того, для обучения этого классификатора нужно оценить параметры $\boldsymbol{\mu}_0$, $\boldsymbol{\mu}_1$ и Σ на основе данных, применяя, например, оценку максимального правдоподобия. Располагая такими оценками, мы можем вычислить значения $\mathbf{w}$ и b, применяя выражения (24.8).

24.4. Скрытые переменные и EM-алгоритм

В порождающих моделях мы предполагаем, что данные порождаются выборкой из определенного параметрического распределения на пространстве образцов $\mathcal{X}$. Иногда удобно выразить это распределение с помощью скрытых случайных величин. Естественный пример – смесь k нормальных распределений. То есть $\mathcal{X} = \mathbb{R}^d$, и предполагается, что каждый вектор $\mathbf{x}$ порождается следующим образом. Сначала выбираем случайное число из множества $\{1, \ldots, k\}$. Пусть Y – случайная величина, соответствующая этому выбору, обозначим $\mathcal{P}[Y = y] = c_y$. Затем выбираем $\mathbf{x}$ на основе значения Y из нормального распределения

$$\mathcal{P}[X = \mathbf{x}|Y = y] = \frac{1}{(2\pi)^{d/2}|\Sigma_y|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_y)^T \Sigma_y^{-1}(\mathbf{x} - \boldsymbol{\mu}_y)\right). \tag{24.9}$$

Таким образом, плотность X можно записать в виде:

$$\begin{aligned}\mathcal{P}[X=\mathbf{x}] &= \sum_{y=1}^{k}\mathcal{P}[Y=y]\,\mathcal{P}[X=\mathbf{x}|Y=y] \\ &= \sum_{y=1}^{k} c_y \frac{1}{(2\pi)^{d/2}|\Sigma_y|^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_y)^T\Sigma_y^{-1}(\mathbf{x}-\boldsymbol{\mu}_y)\right).\end{aligned}$$

Заметим, что Y – скрытая переменная, не наблюдаемая в данных. Тем не менее мы вводим Y, т. к. это помогает описать простую параметрическую форму вероятности X.

Вообще, пусть $\boldsymbol{\theta}$ – параметры совместного распределения X и Y (так, в предыдущем примере $\boldsymbol{\theta}$ включает c_y, $\boldsymbol{\mu}_y$ и Σ_y для всех $y = 1, \ldots, k$). Тогда логарифмическую вероятность наблюдения $\mathbf{x}$ можно записать в виде

$$\log(\mathcal{P}_{\boldsymbol{\theta}}[X=\mathbf{x}]) = \log\left(\sum_{y=1}^{k}\mathcal{P}_{\boldsymbol{\theta}}[X=\mathbf{x},Y=y]\right).$$

Если имеется выборка $S = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$, состоящая из независимых и одинаково распределенных образцов, то мы хотели бы найти значение $\boldsymbol{\theta}$, доставляющее максимум логарифмическому правдоподобию S

$$\begin{aligned}L(\boldsymbol{\theta}) &= \log\prod_{i=1}^{m}\mathcal{P}_{\boldsymbol{\theta}}[X=\mathbf{x}] \\ &= \sum_{i=1}^{m}\log\mathcal{P}_{\boldsymbol{\theta}}[X=\mathbf{x}_i] \\ &= \sum_{i=1}^{m}\log\left(\sum_{y=1}^{k}\mathcal{P}_{\boldsymbol{\theta}}[X=\mathbf{x}_i,Y=y]\right).\end{aligned}$$

Поэтому оценка максимального правдоподобия – это решение задачи максимизации

$$\operatorname*{argmax}_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}}\sum_{i=1}^{m}\log\left(\sum_{y=1}^{k}\mathcal{P}_{\boldsymbol{\theta}}[X=\mathbf{x}_i,Y=y]\right).$$

Во многих случаях суммирование под знаком логарифма делает эту задачу оптимизации вычислительно трудной. EM-алгоритм (Expectation-Maximization), предложенный Дэмпстером (Dempster), Лэрдом (Laird) и Рубином (Rubin), – это итеративная процедура поиска (локального) максимума $L(\boldsymbol{\theta})$. Хотя не гарантируется, что EM-алгоритм находит глобальный максимум, на практике он работает достаточно хорошо.

EM-алгоритм предназначен для случаев, когда, знай мы значения скрытых переменных Y, оптимизация максимального правдоподобия стала бы решаемой задачей. Точнее, мы определяем следующую функцию от матрицы размера $m \times k$ и параметров $\boldsymbol{\theta}$:

$$F(Q,\boldsymbol{\theta}) = \sum_{i=1}^{m}\sum_{y=1}^{k}Q_{i,y}\log(\mathcal{P}_{\boldsymbol{\theta}}[X=\mathbf{x}_i,Y=y]).$$

Если каждая строка Q определяет вероятность i-й скрытой переменной при условии $X = \mathbf{x}_i$, то $F(Q, \boldsymbol{\theta})$ можно интерпретировать как математическое ожидание логарифмического правдоподобия обучающего набора $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, взятое относительно выбора каждой метки y_i на основе i-й строки Q. В определении F суммирование вынесено из-под знака логарифма, и мы предполагаем, что это сделает задачу оптимизации относительно $\boldsymbol{\theta}$ разрешимой.

Предположение 24.1. Для любой матрицы $Q \in [0, 1]^{m,k}$ такой, что сумма элементов в каждой строке равна 1, задача оптимизации

$$\operatorname*{argmax}_{\boldsymbol{\theta}} F(Q, \boldsymbol{\theta})$$

эффективно разрешима.

На интуитивном уровне EM-алгоритм напоминает проблему «яйца и курицы». С одной стороны, если бы мы знали Q, то, согласно нашему предположению, задача оптимизации, состоящая в нахождении наилучших $\boldsymbol{\theta}$, разрешима. С другой стороны, если бы мы знали параметры $\boldsymbol{\theta}$, то могли положить $Q_{i,\,y}$ равным вероятности $Y = y$ при условии $X = \mathbf{x}_i$. Поэтому EM-алгоритм попеременно ищет $\boldsymbol{\theta}$ при заданном Q и Q при заданном $\boldsymbol{\theta}$. Формально говоря, EM ищет последовательность решений $(Q^{(1)}, \boldsymbol{\theta}^{(1)}), (Q^{(2)}, \boldsymbol{\theta}^{(2)}), \ldots$, так что на t-й итерации строится $(Q^{(t+1)}, \boldsymbol{\theta}^{(t+1)})$ путем выполнения двух шагов.

- Шаг вычисления ожидания (Expectation step)

$$Q_{i,y}^{(t+1)} = \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y | X = \mathbf{x}_i]. \tag{24.10}$$

  Этот шаг так называется потому, что дает новые вероятности скрытых переменных и тем самым определяет новую *ожидаемую* функцию логарифмической вероятности от $\boldsymbol{\theta}$.
- Шаг максимизации (Maximization step): положить $\boldsymbol{\theta}^{(t+1)}$ равным точке, в которой достигается максимум ожидаемой логарифмической вероятности, где математическое ожидание вычисляется согласно $Q^{(t+1)}$:

$$\boldsymbol{\theta}^{(t+1)} = \operatorname*{argmax}_{\boldsymbol{\theta}} F(Q^{(t+1)}, \boldsymbol{\theta}). \tag{24.11}$$

По нашему предположению, эту задачу оптимизации можно решить эффективно.

Начальные значения $\boldsymbol{\theta}^{(1)}$ и $Q^{(1)}$ обычно выбираются случайным образом, и процедура завершается, когда правдоподобие перестает существенно изменяться.

24.4.1. EM как алгоритм поочередной максимизации

Чтобы проанализировать EM-алгоритм, мы сначала рассмотрим его как алгоритм поочередной максимизации. Определим следующую целевую функцию

$$G(Q, \boldsymbol{\theta}) = F(Q, \boldsymbol{\theta}) - \sum_{i=1}^{m} \sum_{y=1}^{k} Q_{i,y} \log(Q_{i,y}).$$

Второй член в ней – сумма *энтропий* строк Q. Обозначим

$$\mathbb{Q} = \left\{ Q \in [0,1]^{m,k} : \forall i,\ \sum_{y=1}^{k} Q_{i,y} = 1 \right\}$$

множество матриц, строки которых задают вероятности на множестве $[k]$. Следующая лемма показывает, что EM-алгоритм максимизирует G, поочередно выполняя максимизации по обоим аргументам.

Лемма 24.2. *Процедуру EM можно переписать в виде*

$$Q^{(t+1)} = \operatorname*{argmax}_{Q \in \mathbb{Q}} G(Q, \boldsymbol{\theta}^{(t)}).$$

$$\boldsymbol{\theta}^{(t+1)} = \operatorname*{argmax}_{\boldsymbol{\theta}} G(Q^{(t+1)}, \boldsymbol{\theta}).$$

При этом $G(Q^{(t+1)}, \boldsymbol{\theta}^{(t)}) = L(\boldsymbol{\theta}^{(t)})$.

Доказательство. При заданном $Q^{(t+1)}$ мы, очевидно, имеем

$$\operatorname*{argmax}_{\boldsymbol{\theta}} G(Q^{(t+1)}, \boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} F(Q^{(t+1)}, \boldsymbol{\theta}).$$

Поэтому нужно только показать, что для любого $\boldsymbol{\theta}$ решение задачи $\operatorname{argmax}_{Q \in \mathbb{Q}} G(Q, \boldsymbol{\theta})$ получается, если положить $Q_{i,y} = \mathcal{P}_{\boldsymbol{\theta}}[Y = y | X = \mathbf{x}_i]$. Действительно, согласно неравенству Йенсена, для любого $Q \in \mathbb{Q}$ имеет место

$$\begin{aligned}
G(Q,\boldsymbol{\theta}) &= \sum_{i=1}^{m} \left(\sum_{y=1}^{k} Q_{i,y} \log \left(\frac{\mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i, Y = y]}{Q_{i,y}} \right) \right) \\
&\le \sum_{i=1}^{m} \left(\log \left(\sum_{y=1}^{k} Q_{i,y} \frac{\mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i, Y = y]}{Q_{i,y}} \right) \right) \\
&= \sum_{i=1}^{m} \log \left(\sum_{y=1}^{k} \mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i, Y = y] \right) \\
&= \sum_{i=1}^{m} \log(\mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i]) = L(\boldsymbol{\theta}),
\end{aligned}$$

тогда как для $Q_{i,y} = P_{\boldsymbol{\theta}}[Y = y | X = \mathbf{x}_i]$ имеем

$$\begin{aligned}
G(Q,\boldsymbol{\theta}) &= \sum_{i=1}^{m} \left(\sum_{y=1}^{k} \mathcal{P}_{\boldsymbol{\theta}}[Y = y | X = \mathbf{x}_i] \log \left(\frac{\mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i, Y = y]}{\mathcal{P}_{\boldsymbol{\theta}}[Y = y \mid X = \mathbf{x}_i]} \right) \right) \\
&= \sum_{i=1}^{m} \sum_{y=1}^{k} \mathcal{P}_{\boldsymbol{\theta}}[Y = y | X = \mathbf{x}_i] \log(\mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i]) \\
&= \sum_{i=1}^{m} \log(\mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i]) \sum_{y=1}^{k} \mathcal{P}_{\boldsymbol{\theta}}[Y = y | X = \mathbf{x}_i] \\
&= \sum_{i=1}^{m} \log(\mathcal{P}_{\boldsymbol{\theta}}[X = \mathbf{x}_i]) = L(\boldsymbol{\theta}).
\end{aligned}$$

Это показывает, что $Q_{i,y} = \mathcal{P}_{\boldsymbol{\theta}}[Y = y | X = \mathbf{x}_i]$ доставляет максимум $G(Q, \boldsymbol{\theta})$ на множестве $Q \in \mathbb{Q}$ и что $G(Q^{(t+1)}, \boldsymbol{\theta}^{(t)}) = L(\boldsymbol{\theta}^{(t)})$. □

Из этой леммы сразу вытекает следующая теорема.

Теорема 24.3. *Процедура EM никогда не уменьшает логарифмическое правдоподобие, т. е. для всех t,*

$$L(\boldsymbol{\theta}^{(t+1)}) \geq L(\boldsymbol{\theta}^{(t)}).$$

Доказательство. Согласно лемме, имеем

$$L(\boldsymbol{\theta}^{(t+1)}) = G(Q^{(t+2)}, \boldsymbol{\theta}^{(t+1)}) \geq G(Q^{(t+1)}, \boldsymbol{\theta}^{(t)}) = L(\boldsymbol{\theta}^{(t)}). \qquad \square$$

24.4.2. EM-алгоритм для смеси нормальных распределений (мягкий алгоритм k-средних)

Рассмотрим смесь k нормальных распределений, когда θ является тройкой $(\mathbf{c}, \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k\}, \{\Sigma_1, \ldots, \Sigma_k\})$, где $\mathcal{P}_{\theta}[Y = y] = c_y$ и $\mathcal{P}_{\theta}[X = \mathbf{x} | Y = y]$ определяется выражением (24.9). Для простоты будем предполагать, что $\Sigma_1 = \Sigma_2 = \ldots = \Sigma_k = I$, где I – единичная матрица. EM-алгоритм в этом случае принимает такой вид.

- Шаг вычисления ожидания: для любых $i \in [m]$ и $y \in [k]$ имеем

$$\begin{aligned}\mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y | X = \mathbf{x}_i] &= \frac{1}{Z_i} \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y] \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[X = \mathbf{x}_i | Y = y] \\ &= \frac{1}{Z_i} c_y^{(t)} \exp\left(-\frac{1}{2} \|\mathbf{x}_i - \boldsymbol{\mu}_y^{(t)}\|^2\right), \end{aligned} \tag{24.12}$$

  где Z_i – нормировочный коэффициент, который гарантирует, что сумма $\Sigma_y \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y | X = \mathbf{x}_i]$ равна 1.
- Шаг максимизации: нам нужно найти такое $\boldsymbol{\theta}^{t+1}$, при котором достигается максимум выражения (24.11). В нашем случае это означает максимизацию следующего выражения по $\mathbf{c}$ и $\boldsymbol{\mu}$:

$$\sum_{i=1}^{m} \sum_{y=1}^{k} \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y | X = \mathbf{x}_i] \left(\log(c_y) - \frac{1}{2} \|\mathbf{x}_i - \boldsymbol{\mu}_y\|^2\right). \tag{24.13}$$

  Приравнивая производную (24.13) по $\boldsymbol{\mu}_y$ к нулю и перегруппировывая члены, получаем:

$$\boldsymbol{\mu}_y = \sum_{i=1}^{m} \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y | X = \mathbf{x}_i] \mathbf{x}_i.$$

  То есть $\boldsymbol{\mu}_y$ – взвешенное среднее $\mathbf{x}_i$, в котором веса – вероятности, вычисленные на E-шаге. Для нахождения оптимального $\mathbf{c}$ нужна бóльшая аккуратность, потому что необходимо гарантировать, что $\mathbf{c}$ – вектор вероятностей. В упражнении 24.3 мы покажем, что решение имеет вид

$$c_y = \frac{\sum_{i=1}^{m} \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y | X = \mathbf{x}_i]}{\sum_{y'=1}^{k} \sum_{i=1}^{m} \mathcal{P}_{\boldsymbol{\theta}^{(t)}}[Y = y' | X = \mathbf{x}_i]}. \tag{24.14}$$

Интересно сравнить этот алгоритм с алгоритмом *k*-средних из главы 22. В алгоритме *k*-средних мы сначала относим каждый пример к некоторому кластеру в зависимости от расстояния $||\mathbf{x}_i - \boldsymbol{\mu}_y||$. Затем мы пересчитываем центры $\boldsymbol{\mu}_y$, присваивая каждому среднее всех примеров, отнесенных к соответствующему кластеру. А в EM-алгоритме мы вычисляем вероятность попадания каждого примера в каждый кластер, а затем пересчитываем центры на основе взвешенной суммы по всей выборке. Поэтому EM-алгоритм иногда называют «мягким алгоритмом *k*-средних».

24.5. Байесовское рассуждение

Оценка максимального правдоподобия – пример частотного подхода к теории вероятностей. Это означает, что мы считаем параметр θ фиксированным, а проблема заключается только в том, что мы не знаем его значения. Другой подход к оцениванию параметров называется байесовским рассуждением. В этом случае наша неуверенность относительно θ также моделируется средствами теории вероятностей. То есть мы считаем θ также случайной величиной и распределение $\mathcal{P}[\theta]$ называем *априорным* распределением. Как следует из названия, априорное распределение должно быть задано обучаемым до наблюдения данных.

В качестве примера снова рассмотрим фармацевтическую компанию, разработавшую новое лекарство. На основе прошлого опыта работающие в ней статистики полагают, что если лекарство дошло до стадии клинических испытаний на людях, то оно, скорее всего, будет эффективно. Эту априорную веру они моделируют, определив такую функцию плотности вероятности, что

$$\mathcal{P}(\theta) = \begin{cases} 0.8, & \text{если } \theta > 0.5 \\ 0.2, & \text{если } \theta \leq 0.5 \end{cases}. \tag{24.15}$$

Как и раньше, предполагается, что при заданном значении θ условная вероятность $\mathcal{P}[X = x|\theta]$ известна. В этом примере X принимает значения из множества {0, 1} и $\mathcal{P}[X = x|\theta] = \theta^x(1 - \theta)^{1-x}$.

Зная априорное распределение θ и условное распределение X при условии θ, мы располагаем полным знанием о распределении X, потому что распределение вероятности X можно записать в виде маргинального распределения:

$$\mathcal{P}[X = x] = \sum_{\theta} \mathcal{P}[X = x, \theta] = \sum_{\theta} \mathcal{P}[\theta]\mathcal{P}[X = x|\theta],$$

где последнее равенство следует из определения условной вероятности. Если θ – непрерывная случайная величина, то мы заменим $\mathcal{P}[\theta]$ функцией плотности вероятности, а сумму – интегралом:

$$\mathcal{P}[X = x] = \int_{\theta} \mathcal{P}[\theta]\mathcal{P}[X = x|\theta]d\theta.$$

Казалось бы, что если нам известно $\mathcal{P}[X = x]$, то обучающий набор $S = (x_1, \ldots, x_m)$ не скажет ничего нового, поскольку мы и так уже являемся экспертами и все знаем о вероятности новой точки X. Однако байесовский взгляд на вещи вводит

зависимость между S и X. Дело в том, что теперь мы рассматриваем θ как случайную величину. Новая точка X и предыдущие точки S независимы *только* при условии θ. Это отличается от частотной философии, в которой θ – параметр, который мы можем и не знать, но, поскольку это всего лишь параметр распределения, то новая точка X и предыдущие точки S независимы всегда. В байесовской теории, поскольку X и S уже не являются независимыми, нас интересует вычисление вероятности X при условии S, которую, по формуле полной вероятности, можно записать в виде:

$$\mathcal{P}[X = x|S] = \sum_{\theta} \mathcal{P}[X = x|\theta, S]\,\mathcal{P}[\theta|S] = \sum_{\theta} \mathcal{P}[X = x|\theta]\,\mathcal{P}[\theta|S].$$

Второе равенство следует из допущения о независимости X и S при условии θ. Применяя *формулу Байеса*, получаем

$$\mathcal{P}[\theta|S] = \frac{\mathcal{P}[S|\theta]\mathcal{P}[\theta]}{\mathcal{P}[S]},$$

а принимая во внимание предположение о независимости точек при условии θ, мы можем написать

$$\mathcal{P}[\theta|S] = \frac{\mathcal{P}[S|\theta]\mathcal{P}[\theta]}{\mathcal{P}[S]} = \frac{1}{\mathcal{P}[S]} \prod_{i=1}^{m} \mathcal{P}[X = x_i|\theta]\mathcal{P}[\theta].$$

Таким образом, мы получили следующее выражение для байесовского предсказания:

$$\mathcal{P}[X = x|S] = \frac{1}{\mathcal{P}[S]} \sum_{\theta} \mathcal{P}[X = x_i|\theta] \prod_{i=1}^{m} \mathcal{P}[X = x_i|\theta]\mathcal{P}[\theta]. \tag{24.16}$$

Возвращаясь к примеру фармацевтической компании, мы можем переписать $\mathcal{P}[X = x|S]$ в виде:

$$\mathcal{P}[X = x|S] = \frac{1}{\mathcal{P}[S]} \int \theta^{x+\sum_i x_i} (1-\theta)^{1-x+\sum_i (1-x_i)} \mathcal{P}[\theta] d\theta.$$

Интересно отметить, что если распределение $\mathcal{P}[\theta]$ равномерно, то получаем:

$$\mathcal{P}[X = x|S] = \propto \int \theta^{x+\sum_i x_i} (1-\theta)^{1-x+\sum_i (1-x_i)} d\theta.$$

Этот интеграл можно вычислить по частям:

$$\mathcal{P}[X = 1|S] = \frac{\left(\sum_i x_i\right) + 1}{m + 2}.$$

Напомним, что принцип максимального правдоподобия в этом случае предсказывает $\mathcal{P}[X = 1|\hat{\theta}] = (\sum_i x_i)/m$. Байесовское предсказание при равномерном априорном распределении напоминает предсказание максимального правдоподобия, но добавляет в обучающий набор «псевдопримеры», смещая предсказание в сторону равномерного априорного распределения.

Оценка апостериорного максимума

Во многих случаях трудно выразить интеграл (24.16) в замкнутой форме. Для его аппроксимации существует несколько численных методов. Другое популярное решение – искать то единственное значение θ, которое доставляет максимум $\mathcal{P}[\theta|S]$. Это значение называется оценкой *апостериорного максимума* (Maximum A Posteriori). Зная его, мы можем вычислить условную вероятность $X = x$.

24.6. Резюме

В рамках порождающего подхода к машинному обучению мы ставим задачу смоделировать распределение данных. В частности, при параметрическом оценивании функции плотности вероятности мы дополнительно предполагаем, что истинное распределение данных имеет определенную параметрическую форму и хотим оценить параметры модели. Мы описали несколько принципов параметрического оценивания, включая метод максимального правдоподобия, байесовское оценивание и оценку апостериорного максимума. Мы также рассказали о конкретных алгоритмах, реализующих оценку максимального правдоподобия при различных предположениях об истинном распределении данных, в т. ч. наивный байесовский классификатор, линейный дискриминантный анализ и EM-алгоритм.

24.7. Библиографические сведения

Принцип максимального правдоподобия изучался Рональдом Фишером в начала XX в. Байесовская статистика вытекает из формулы Байеса, названной в честь английского математика XVIII в. Томаса Байеса.

Есть немало прекрасных книг о порождающем и байесовском подходах к машинному обучению. См., например, Bishop, 2006; Koller & Friedman, 2009b; MacKay, 2003; Murphy, 2012; Barber, 2012.

24.8. Упражнения

24.1. Докажите, что оценка максимального правдоподобия дисперсии нормальной случайной величины смещена.

24.2. Регуляризация для максимального правдоподобия. Рассмотрим следующую задачу о минимизации регуляризированной потери:

$$\frac{1}{m}\sum_{i=1}^{m}\log(1/\mathcal{P}_\theta[x_i]) + \frac{1}{m}\big(\log(1/\theta) + \log(1/(1-\theta))\big).$$

- Покажите, что эта целевая функция была бы эквивалентна обычной эмпирической ошибке, если бы мы добавили в обучающий набор два псевдопримера. Сделайте отсюда вывод, что регуляризированная оценка максимального правдоподобия имела бы вид:

$$\hat{\theta} = \frac{1}{m+2}\left(1 + \sum_{i=1}^{m} x_i\right).$$

- Выведите оценку сверху для $|\hat{\theta} - \theta^*|$. *Указание*: перепишите в виде $|\hat{\theta} - \mathbb{E}[\hat{\theta}] + \mathbb{E}[\hat{\theta}] - \theta^*|$, затем воспользуйтесь неравенством треугольника и неравенством Хёфдинга.
- Воспользуйтесь этим результатом, чтобы ограничить сверху истинный риск. *Указание:* используйте тот факт, что теперь $\hat{\theta} \geq 1/(m+2)$, чтобы связать $|\hat{\theta} - \theta^*|$ с относительной энтропией.

24.3. Рассмотрим общую задачу оптимизации вида

$$\max_{\mathbf{c}} \sum_{y=1}^{k} \nu_y \log(c_y) \quad s.t. \quad c_y > 0, \sum_y c_y = 1,$$

где $\boldsymbol{\nu} \in \mathbb{R}_+^k$ – вектор неотрицательных весов.

- Убедитесь, что М-шаг мягкого алгоритма k-средних включает решение такой задачи оптимизации
- Пусть $c^* = 1/(\Sigma_y \nu_y)\boldsymbol{\nu}$. Докажите, что c^* – вектор вероятностей.
- Покажите, что эта задача оптимизации эквивалентна задаче

$$\min_{\mathbf{c}} D_{\mathrm{RE}}(\mathbf{c}^* \| \mathbf{c}) \quad s.t. \quad c_y > 0, \sum_y c_y = 1.$$

- Пользуясь свойствами относительной энтропии, сделайте вывод, что c^* – решение этой задачи оптимизации.

ЛАНЬ®

Глава 25

ОТБОР И ПОРОЖДЕНИЕ ПРИЗНАКОВ

ЛАНЬ®

В начале книги мы обсуждали абстрактную модель обучения, в которой априорные знания обучаемого закодированы исключительно в виде выбора класса гипотез. Однако существует еще один аспект моделирования, который мы до сих пор игнорировали: как представить пространство образцов $\mathcal{X}$? Например, в задаче об обучении классификатора плодов папайи мы предложили взять в качестве класса гипотез класс прямоугольников на двумерной плоскости цвет-мягкость. То есть сначала мы построили модель, представив папайю точкой на двумерной плоскости, описывающей цвет и мягкость плода. И лишь потом выбрали класс прямоугольников как класс отображений этой плоскости в множество меток. Преобразование объекта реального мира – «папайи» – в скаляр, представляющий его мягкость или цвет, называется *функцией признака*, или просто признаком. Вообще, любое измерение свойств реального объекта можно рассматривать как признак. Если $\mathcal{X}$ – подмножество векторного пространства, то элементы $x \in \mathcal{X}$ иногда называют *векторами признаков*. Важно понимать, что способ кодирования реальных объектов элементами пространства образцов $\mathcal{X}$ сам по себе является априорным знанием о проблеме.

Но, даже если мы уже имеем пространство образцов $\mathcal{X}$, представленное в виде подмножества векторного пространства, все равно может возникнуть желание преобразовать его в другое представление и на него наложить класс гипотез. То есть мы можем определить класс гипотез на $\mathcal{X}$ в виде композиции некоторого класса H с функцией признаков, которая отображает $\mathcal{X}$ в другое векторное пространство $\mathcal{X}'$. Нам уже встречались примеры таких композиций – в главе 15 мы видели, что ядерный метод SVM обучается композиции класса полупространств с функцией признаков ψ, которая отображает исходный образец из $\mathcal{X}$ в некоторое гильбертово пространство. И выбор ψ – это еще одна форма априорного знания о проблеме.

В этой главе мы изучим несколько методов построения хорошего набора признаков. Начнем с проблемы *отбора признаков*, когда имеется большой пул признаков, и наша цель – выбрать небольшое их количество для использования в предикторе. Затем мы обсудим *манипуляции и нормировку признаков*. Сюда относятся простые преобразования исходных признаков, которые могут уменьшить выборочную сложность алгоритма обучения, его смещение или его вычислительную сложность. И напоследок мы рассмотрим различные подходы к *обучению признаков* и попытаемся автоматизировать процесс построения признаков.

Мы еще раз подчеркиваем, что, хотя имеются общепринятые приемы обучения признаков, которые стоит попробовать, теорема об отсутствии бесплатных завтраков утверждает, что никакого верховного алгоритма их обучения не существует. Любой такой алгоритм обязательно потерпит неудачу на какой-то проблеме. Иными словами, успех алгоритма обучения признаков зависит (иногда неявно) от той или иной формы априорного предположения о распределении данных. Кроме того, относительное качество признаков сильно зависит от алгоритма обучения, в котором эти признаки будут впоследствии применены. Для иллюстрации приведем пример.

Пример 25.1. Рассмотрим проблему регрессии, в которой $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \mathbb{R}$ и используется квадратичная функция потерь. Предположим, что пример $(\mathbf{x}, y)$ в действительности генерируется следующим образом: сначала выбираем x_1 из равномерного распределения на отрезке $[-1, 1]$. Затем детерминированно полагаем $y = x_1^2$. А в качестве второго признака берем $x_2 = y + z$, где z – выбирается из равномерного распределения на отрезке $[-0{,}01, 0{,}01]$. Допустим, что мы хотели бы выбрать единственный признак. Интуитивно кажется, что первый признак следует предпочесть, потому что, зная его, метку всегда можно предсказать точно, тогда как по второму признаку это сделать нельзя. И, действительно, выбор первого признака будет правильным решением, если мы затем собираемся использовать полиномиальную регрессию степени не меньше 2. Но если обучаемым будет алгоритм линейной регрессии, то следует предпочесть *второй* признак, поскольку у оптимального линейного предиктора на основе первого признака риск будет больше, чем у предиктора на основе второго признака.

25.1. Отбор признаков

В этом разделе мы будем предполагать, что $\mathcal{X} = \mathbb{R}^d$, т. е. все образцы представлены векторами d признаков. Наша цель – обучить предиктор, зависящий только от k из d признаков. Чем меньше признаков использует предиктор, тем меньше ему нужно памяти и тем быстрее он будет работать. Кроме того, в таких приложениях, как медицинская диагностика, получение всех возможных «признаков» (например, результатов анализов) может стоить очень дорого, поэтому предиктор, которому нужно немного признаков, желателен – даже ценой незначительного ухудшения качества по сравнению с предиктором, использующим больше признаков. Наконец, ограничив класс гипотез таким, где число признаков невелико, мы, возможно, сумеем уменьшить ошибку оценивания и тем предотвратить переобучение.

В идеале можно было бы попробовать все подмножества k из d признаков и выбрать то, для которого получается наилучший предиктор. Но такой исчерпывающий поиск обычно вычислительно нереализуем. Ниже мы опишем три вычислительно реализуемых подхода к отбору признаков. Хотя эти методы не могут гарантировать нахождение оптимального подмножества, на практике они часто показывают хорошие результаты. Для некоторых из них можно дать формальные гарантии качества отобранных подмножеств при выполнении некоторых условий. Здесь мы эти гарантии обсуждать не будем.

25.1.1. Фильтры

Пожалуй, самый простой подход к отбору признаков – метод фильтрации, когда мы оцениваем отдельные признаки независимо от остальных, применяя некоторый критерий качества. Затем мы отбираем k признаков с наивысшей оценкой (можно также на основе оценок принимать решение о количестве признаков).

В литературе предложено много критериев для оценки качества признаков. Самый прямолинейный – оценивать признак по частоте ошибок предиктора, обученного только на этом признаке.

Для иллюстрации рассмотрим проблему линейной регрессии с квадратичной потерей. Пусть $\mathbf{v} = (x_{1,j}, \ldots, x_{m,j}) \in \mathbb{R}^m$ – вектор, состоящий из значений j-го признака для m образцов из обучающего набора, а $\mathbf{y} = (y_1, \ldots, y_m) \in \mathbb{R}^m$ – значения меток для этих самых m примеров. Эмпирическая квадратичная потеря линейного ERM-предиктора, в котором используется только j-й признак, равна

$$\min_{a,b\in\mathbb{R}} \frac{1}{m} \|a\mathbf{v} + b - \mathbf{y}\|^2,$$

где прибавление скаляра b к вектору $\mathbf{v}$ означает прибавление b ко всем элементам $\mathbf{v}$. Для решения этой задачи положим $\bar{v} = (1/m)\sum_{i=1}^m v_i$ – усредненное значение признака и $\bar{y} = (1/m)\sum_{i=1}^m y_i$ – усредненное значение меток. Очевидно (см. упражнение 25.1), что

$$\min_{a,b\in\mathbb{R}} \frac{1}{m} \|a\mathbf{v} + b - \mathbf{y}\|^2 = \min_{a,b\in\mathbb{R}} \frac{1}{m} \|a(\mathbf{v} + \bar{v}) + b - (\mathbf{y} - \bar{y})\|^2. \tag{25.1}$$

Приравнивая к нулю производную правой части по b, получаем, что $b = 0$. Поступая аналогичным образом для a (уже зная, что $b = 0$) получаем, что $a = \langle \mathbf{v} - \bar{v}, \mathbf{y} - \bar{y} \rangle / \|\mathbf{v} - \bar{v}\|^2$. Подставляя это значение назад в целевую функцию, получаем

$$\|\mathbf{y} - \bar{y}\|^2 - \frac{(\langle \mathbf{v} - \bar{v}, \mathbf{y} - \bar{y} \rangle)^2}{\|\mathbf{v} - \bar{v}\|^2}.$$

Ранжирование признаков по минимальной потере на них эквивалентно ранжированию по абсолютной величине следующей оценки (чем оценка больше, тем признак лучше):

$$\frac{\langle \mathbf{v} - \bar{v}, \mathbf{y} - \bar{y} \rangle}{\|\mathbf{v} - \bar{v}\| \|\mathbf{y} - \bar{y}\|} = \frac{\frac{1}{m}\langle \mathbf{v} - \bar{v}, \mathbf{y} - \bar{y} \rangle}{\sqrt{\frac{1}{m}\|\mathbf{v} - \bar{v}\|^2} \sqrt{\frac{1}{m}\|\mathbf{y} - \bar{y}\|^2}}. \tag{25.2}$$

Это выражение называется *коэффициентом корреляции Пирсона*. В числителе стоит эмпирическая оценка ковариации j-го признака и метки, $\mathbb{E}[(v - \mathbb{E}v)(y - \mathbb{E}y)]$, а в знаменателе – квадратный корень из эмпирической оценки дисперсии j-го признака, $\mathbb{E}[(v - \mathbb{E}v)^2]$, умноженный на дисперсию метки. Коэффициент Пирсона принимает значения из диапазона от -1 до 1, причем значение 1 или -1 означает, что существует линейное отображение $\mathbf{v}$ в $\mathbf{y}$ с нулевым эмпирическим риском.

Если коэффициент Пирсона равен нулю, значит, оптимальная линейная функция, отображающая $\mathbf{v}$ в $\mathbf{y}$, нулевая, т. е. *одного* вектора $\mathbf{v}$ недостаточно для предсказания $\mathbf{y}$. Но это не значит, что $\mathbf{v}$ – плохой признак, поскольку не исключено, что в сочетании с другими признаками $\mathbf{v}$ позволяет точно предсказать $\mathbf{y}$. Действительно, рассмотрим простой пример, когда метки генерируются функцией $y = x_1 + 2x_2$. Предположим еще, что x_1 выбирается из равномерного распределения на множестве $\{\pm 1\}$, а $x_2 = -\frac{1}{2}x_1 + \frac{1}{2}z$, где z также независимо выбирается из равномерного распределения на $\{\pm 1\}$. Тогда $\mathbb{E}[x_1] = \mathbb{E}[x_2] = \mathbb{E}[y] = 0$ и имеем также

$$\mathbb{E}[yx_1] = \mathbb{E}[x_2^1] + 2\mathbb{E}[x_2x_1] = \mathbb{E}[x_2^1] - \mathbb{E}[x_2^1] + \mathbb{E}[zx_1] = 0.$$

Поэтому для достаточно большого обучающего набора коэффициент корреляции Пирсона первого признака, скорее всего, будет близок к нулю, в силу чего он вряд ли будет отобран. Однако никакая функция не сможет надежно предсказать метку, не зная первый признак.

Существует много других оценочных функций, которые можно использовать в методе фильтрации. К наиболее известным относятся оценки взаимной информации, площади под кривой рабочей характеристики приемника (РХП) (receiver operating characteristic – ROC). Но всем им присущи те же проблемы, что в примере выше. Отсылаем читателя к работе Guyon and Elisseeff (2003).

25.1.2. Подходы на основе жадного отбора

Жадный отбор – еще один популярный подход к отбору признаков. В отличие от фильтрации, такие подходы неотделимы от соответствующего алгоритма обучения. Простейший пример – прямой жадный отбор. Мы начинаем с пустого множества признаков, а затем постепенно добавляем в него по одному признаку. Если обозначить текущее множество отобранных признаков I, то мы перебираем все $i \notin I$ и применяем алгоритм обучения к множеству признаков $I \cup \{i\}$. При каждом таком применении получается новый предиктор, и мы выбираем для добавления тот признак, которому соответствует предиктор с наименьшим риском (на обучающем или на контрольном наборе). Этот процесс продолжается, пока не будет отобрано k признаков, где k – заранее заданное значение, или не будет получен достаточно верный предиктор.

Пример 25.2 (ортогональное согласованное преследование). Для иллюстрации прямого жадного отбора применим его к проблеме линейной регрессии с квадратичной потерей. Пусть $X \in \mathbb{R}^{m,d}$ – матрица, строками которой являются m обучающих образцов. Пусть $\mathbf{y} \in \mathbb{R}^m$ – вектор m меток. Для любого $i \in [d]$ обозначим X_i – i-й столбец X. Для заданного множества $I \subset [d]$ обозначим X_I матрицу, образованную столбцами $\{X_i : i \in I\}$.

Прямой жадный отбор начинается с множества $I_0 = \emptyset$. На t-й итерации мы ищем индекс признака j_t, принадлежащий множеству

$$\arg\min_j \min_{\mathbf{w} \in \mathbb{R}^t} \|X_{I_{t-1} \cup \{j\}}\mathbf{x} - \mathbf{y}\|^2.$$

Затем производим обновление $I_t = I_{t-1} \cup \{j_t\}$.

Теперь опишем более эффективную реализацию прямого жадного отбора для линейной регрессии. Этот метод называется ортогональным согласованным преследованием (Orthogonal Matching Pursuit – OMP). Идея в том, чтобы хранить ортогональный базис уже агрегированных признаков. Пусть V_t – матрица, столбцы которой образуют ортонормированный базис пространства, натянутого на столбцы X_{I_t}. Очевидно, что

$$\min_{\mathbf{w}} ||X_{I_t}\mathbf{w} - \mathbf{y}||^2 = \min_{\boldsymbol{\theta} \in \mathbb{R}^t} ||V_t\boldsymbol{\theta} - \mathbf{y}||^2.$$

Будем хранить вектор $\boldsymbol{\theta}_t$, доставляющий минимум правой части. В начальный момент полагаем $I_0 = \emptyset$, $V_0 = \emptyset$, $\boldsymbol{\theta}_1$ – пустой вектор. На t-й итерации мы для каждого j производим разложение $X_j = \mathbf{v}_j + \mathbf{u}_j$, где $\mathbf{v}_j = V_{t-1}V_{t-1}^{\mathrm{T}}X_j$ – проекция X_j на подпространство, натянутое на векторы V_{t-1}, а $\mathbf{u}_j$ – часть X_j, ортогональная V_{t-1} (см. приложение С). Тогда

$$\begin{aligned}
&\min_{\boldsymbol{\theta},\alpha} ||\, V_{t-1}\boldsymbol{\theta} + \alpha\mathbf{u}_j - \mathbf{y}\,||^2 \\
&\quad = \min_{\boldsymbol{\theta},\alpha}[||\, V_{t-1}\boldsymbol{\theta} - \mathbf{y}||^2 + \alpha^2||\,\mathbf{u}_j||^2 + 2\alpha\langle \mathbf{u}_j, V_{t-1}\boldsymbol{\theta} - \mathbf{y}\rangle] \\
&\quad = \min_{\boldsymbol{\theta},\alpha}[||\, V_{t-1}\boldsymbol{\theta} - \mathbf{y}||^2 + \alpha^2||\,\mathbf{u}_j||^2 + 2\alpha\langle \mathbf{u}_j, -\mathbf{y}\rangle] \\
&\quad = \min_{\boldsymbol{\theta},\alpha}[||\, V_{t-1}\boldsymbol{\theta} - \mathbf{y}||^2] + \min_{\alpha}[\alpha^2||\,\mathbf{u}_j||^2 - 2\alpha\langle \mathbf{u}_j, \mathbf{y}\rangle] \\
&\quad = [||\, V_{t-1}\boldsymbol{\theta}_{t-1} - \mathbf{y}||^2 + \min_{\alpha}[\alpha^2||\,\mathbf{u}_j||^2 - 2\alpha\langle \mathbf{u}_j, \mathbf{y}\rangle] \\
&\quad = [||\, V_{t-1}\boldsymbol{\theta}_{t-1} - \mathbf{y}||^2 - \frac{(\langle \mathbf{u}_j, \mathbf{y}\rangle)^2}{||\,\mathbf{u}_j\,||^2}.
\end{aligned}$$

Отсюда следует, что нужно выбрать признак

$$j_t = \arg\min_j \frac{(\langle \mathbf{u}_j, \mathbf{y}\rangle)^2}{||\,\mathbf{u}_j\,||^2}$$

и далее закончить обновление

$$V_t = \left[V_{t-1}, \frac{\mathbf{u}_{j_t}}{||\,\mathbf{u}_{j_t}\,||^2}\right], \quad \boldsymbol{\theta}_t = \left[\boldsymbol{\theta}_{t-1}; \frac{\langle \mathbf{u}_{j_t}, \mathbf{y}\rangle}{||\,\mathbf{u}_{j_t}\,||^2}\right].$$

Процедура OMP хранит ортонормированный базис отобранных признаков. В приведенном выше описании свойство ортонормированности обеспечивается процедурой, напоминающей ортонормирование Грама–Шмидта. На практике метод Грама–Шмидта численно неустойчив. В показанном ниже псевдокоде используется метод сингулярного разложения SVD (см. раздел С.4), так что в конце каждой итерации мы получаем ортонормированный базис, не рискуя пасть жертвой численной неустойчивости.

Ортогональное согласованное преследование (OMP)

вход:
матрица данных $X \in \mathbb{R}^{m,d}$, вектор меток $\mathbf{y} \in \mathbb{R}^m$,
заданное количество признаков T
инициализация: $I_1 = \emptyset$
for $t = 1, \ldots, T$
с помощью метода SVD найти ортонормированный базис $V \in \mathbb{R}^{m,t-1}$ пространства X_{I_t} (для $t = 1$ взять в качестве V нулевую матрицу)
foreach $j \in [d] \backslash I_t$ положить $\mathbf{u}_j = X_j - VV^T X_j$
положить $j_t = \operatorname{argmax}_{j \notin I_t : \|\mathbf{u}_j\| > 0} (\langle \mathbf{u}_j, \mathbf{y} \rangle)^2 / \|\mathbf{u}_j\|^2$
обновить $I_{t+1} = I_t \cup \{j_t\}$
выход I_{T+1}

Более эффективные критерии жадного отбора

Обозначим $R(\mathbf{w})$ эмпирический риск вектора $\mathbf{w}$. На каждой итерации прямого метода жадного отбора и для каждого возможного j мы должны минимизировать $R(\mathbf{w})$ по всем векторам $\mathbf{w}$ с опорой $I_{t-1} \cup \{j\}$. Этот процесс может занимать много времени.

Проще выбрать j_t, который принадлежит множеству

$$\operatorname*{argmin}_j \min_{\eta \in \mathbb{R}} R(\mathbf{w}_{t-1} + \eta \mathbf{e}_j),$$

где $\mathbf{e}_j$ – вектор, все элементы которого, кроме j-го, равны 0, а j-й равен 1. Иначе говоря, мы сохраняем веса ранее выбранных координат без изменения, а оптимизируем только новую переменную. Таким образом, для каждого j нужно решить только задачу оптимизации по одной переменной, что гораздо проще, чем оптимизировать по t.

Еще более простой подход – ограничить $R(\mathbf{w})$ сверху «простой» функцией, а затем выбрать признак, который дает наибольшее уменьшение этой верхней границы. Например, если R – β-гладкая функция (см. выражение (12.5) в главе 12), то

$$R(\mathbf{w} + \eta \mathbf{e}_j) \le R(\mathbf{w}) + \eta \frac{\partial R(\mathbf{w})}{\partial w_j} + \beta \eta^2 / 2.$$

Минимизация правой части по η дает $-\frac{\partial R(\mathbf{w})}{\partial w_j} \cdot \frac{1}{\beta}$, и, подставляя это значение в неравенство, получаем

$$R(\mathbf{w}) - \frac{1}{2\beta} \left(\frac{\partial R(\mathbf{w})}{\partial w_j} \right)^2.$$

Это значение достигает минимума, когда частная производная $R(\mathbf{w})$ по w_j максимальна. Поэтому можно в качестве j_t выбрать индекс наибольшей координаты градиента $R(\mathbf{w})$ в точке $\mathbf{w}$.

Замечание 25.3 (AdaBoost как процедура прямого жадного отбора). Алгоритм AdaBoost, описанный в главе 10, можно интерпретировать как процедуру прямого жадного отбора относительно функции

$$R(\mathbf{w}) = \log\left(\sum_{i=1}^{m} \exp\left(-y_i \sum_{j=1}^{d} w_j h_j(\mathbf{x}_j)\right)\right)^2. \tag{25.3}$$

См. упражнение 25.3.

ЛАНЬ®

Обратное исключение

Еще один популярный подход к жадному отбору называется *обратным исключением* (backward elimination). В этом случае мы начинаем с полного множества признаков и удаляем признаки по одному. Если обозначить текущее множество отобранных признаков I, то мы перебираем все $i \in I$ и применяем алгоритм обучения к множеству $I \setminus \{i\}$. При каждом таком применении получается новый предиктор, и мы решаем удалить тот признак i, для которого предиктор, обученный на признаках $I \setminus \{i\}$, имеет наибольший риск (на обучающем или контрольном наборе).

Естественно, возможно много вариаций на тему этой идеи. Можно также объединить шаги прямого и обратного жадного отбора.

25.1.3. Нормы, индуцирующие разреженность

Задачу минимизации эмпирического риска при k признаках можно записать в виде

$$\min_{\mathbf{w}} L_S(\mathbf{w}) \quad \text{s.t.} \quad ||\mathbf{w}||_0 \le k,$$

где[1]

$$||\mathbf{w}||_0 = |\{i : w_i \neq 0\}|.$$

Иными словами, мы хотим, чтобы вектор $\mathbf{w}$ был разреженным, т. е. можно было измерять только признаки, соответствующие ненулевым элементам $\mathbf{w}$.

Эта задача оптимизации вычислительно трудна (Natarajan, 1995; Davis, Mallat & Avellaneda, 1997). Ее можно ослабить, заменив невыпуклую функцию $||\mathbf{w}||_0$ нормой ℓ_1, $||\mathbf{w}||_1 = \sum_{i=1}^{d} |w_i|$, и решить задачу

$$\min_{\mathbf{w}} L_S(\mathbf{w}) \quad \text{s.t.} \quad ||\mathbf{w}||_1 \le k_1. \tag{25.4}$$

где k_1 – параметр. Поскольку норма ℓ_1 – выпуклая функция, эту задачу можно решить эффективно при условии, что функция потерь тоже выпукла. Родственная задача – минимизировать сумму $L_S(\mathbf{w})$ плюс регуляризирующий член по норме ℓ_1:

[1] Функцию $||\cdot||_0$ часто называют нормой ℓ_0. Но, несмотря на нотацию, это не настоящая норма, т. к. не обладает свойством положительной однородности: $||a\mathbf{w}||_0 \neq |a|\,||\mathbf{w}||_0$.

$$\min_{\mathbf{w}} (L_S(\mathbf{w}) + \lambda ||\mathbf{w}||_1), \tag{25.5}$$

где λ – параметр регуляризации. Поскольку для любого k_1 существует такое λ, что задачи (25.4) и (25.5) имеют одинаковое решение, то в некотором смысле эти задачи эквивалентны.

Регуляризация по норме ℓ_1 часто индуцирует разреженные решения. Для иллюстрации начнем с простой задачи оптимизации

$$\min_{w \in \mathbb{R}} \left(\frac{1}{2} w^2 - xw + \lambda |w| \right). \tag{25.6}$$

Легко проверить (см. упражнение 25.2), что решением этой задачи является оператор «мягкой бинаризации»

$$w = \text{sign}(x)[|x| - \lambda]_+, \tag{25.7}$$

где $[a]_+ \stackrel{\text{def}}{=} \max\{a, 0\}$. Таким образом, при условии, что абсолютная величина x меньше λ, оптимальное решение будет равно нулю.

Далее рассмотрим одномерную проблему регрессии с квадратичной потерей:

$$\operatorname*{argmin}_{w \in \mathbb{R}^m} \left(\frac{1}{2m} \sum_{i=1}^{m} (x_i w - y_i)^2 + \lambda |w| \right).$$

Мы можем переписать ее в виде

$$\operatorname*{argmin}_{w \in \mathbb{R}^m} \left(\frac{1}{2m} \left(\frac{1}{m} \sum_i x_i^2 \right) w^2 - \left(\frac{1}{m} \sum_{i=1}^{m} x_i y_i \right) w - \lambda |w| \right).$$

Для простоты предположим, что $(1/m)\sum_i x_i^2 = 1$, и обозначим $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^m x_i y_i$; тогда оптимальное решение имеет вид

$$w = \text{sign}(\langle \mathbf{x}, \mathbf{y} \rangle)[|\langle \mathbf{x}, \mathbf{y} \rangle|/m - \lambda]_+.$$

Это решение равно нулю, если корреляция между признаком $\mathbf{x}$ и вектором меток $\mathbf{y}$ больше λ.

Замечание 25.4. В отличие от нормы ℓ_1, норма ℓ_2 не индуцирует разреженных решений. Действительно, рассмотрим вышеупомянутую проблему с регуляризацией по норме ℓ_2:

$$\operatorname*{argmin}_{w \in \mathbb{R}^m} \left(\frac{1}{2m} \sum_{i=1}^{m} (x_i w - y_i)^2 + \lambda w^2 \right).$$

Тогда оптимальное решение имеет вид

$$w = \frac{\langle \mathbf{x}, \mathbf{y} \rangle / m}{\|\mathbf{x}\|^2/m + 2\lambda}.$$

Это решение отлично от нуля, даже если корреляция между $\mathbf{x}$ и $\mathbf{y}$ очень мала. Напротив, выше было показано, что при регуляризации по норме ℓ_1 величина w

будет ненулевой, только если корреляция между **x** и **y** больше параметра регуляризации λ.

Включение регуляризации по норме ℓ_1 в проблему линейной регрессии с квадратичной потерей дает алгоритм LASSO, определяемый как

$$\arg\min_{\mathbf{w}} \left(\frac{1}{2m} \|X\mathbf{w} - \mathbf{y}\|^2 + \lambda \|\mathbf{w}\|_1 \right). \tag{25.8}$$

При определенных допущениях о распределении и параметре регуляризации λ метод LASSO находит разреженные решения (см., например, работу Zhao & Yu (2006) и приведенные в ней ссылки). Еще одно преимущество нормы ℓ_1 заключается в том, что вектор с малой нормой ℓ_1 можно «разредить» (см., например, работу (Shalev-Shwartz, Zhang and Srebro, 2010) и приведенные в ней ссылки).

25.2. Манипулирование и нормировка признаков

Под манипулированием или нормировкой признаков понимается простое преобразование, применяемое к каждому исходному признаку. Такие преобразования иногда уменьшают ошибки аппроксимации или оценивания нашего класса гипотез, а иногда приводят к более быстрому алгоритму. Как и в задаче об отборе признаков, здесь нет абсолютно «хороших» или «плохих» преобразований, просто любое применяемое преобразование следует соотносить с алгоритмом обучения, который мы планируем применить к результирующему вектору признаков, а также с априорными предположениями о проблеме.

В обоснование нормировки рассмотрим проблему линейной регрессии с квадратичной потерей. Пусть $X \in \mathbb{R}^{m,d}$ – матрица, строками которой являются векторы образцов, а $\mathbf{y} \in \mathbb{R}^m$ – вектор меток. Напомним, что гребневая регрессия возвращает вектор

$$\arg\min_{\mathbf{w}} \left[\frac{1}{m} \|X\mathbf{w} - \mathbf{y}\|^2 + \lambda \|\mathbf{w}\|^2 \right] = (2\lambda m I + X^\top X)^{-1} X^\top \mathbf{y}.$$

Предположим, что $d = 2$ и истинное распределение данных устроено следующим образом. Сначала выбираем y из равномерного распределения на множестве $\{\pm 1\}$. Затем полагаем $x_1 = y + 0{,}5\alpha$, где α выбрано из равномерного распределения на множестве $\{\pm 1\}$, а $x_2 = 0{,}0001y$. Заметим, что оптимальный вектор весов $\mathbf{w}^* = [0; 10000]$, а $L_D(\mathbf{w}^*) = 0$. Однако целевая функция гребневой регрессии в точке $\mathbf{w}^*$ принимает значение $\lambda 10^8$. С другой стороны, в точке $\mathbf{w} = [1; 0]$ целевая функция гребневой регрессии, скорее всего, будет близка к $0{,}25 + \lambda$. Отсюда следует, что при $\lambda > 0{,}25/(10^8 - 1) \approx 0.25 \times 10^{-8}$ целевая функция гребневой регрессии оказывается меньше на неоптимальном решении $\mathbf{w} = [1; 0]$. Так как λ обычно должно быть не меньше $1/m$ (см. анализ в главе 13), то получается, что в этом примере мы с большой вероятностью найдем неоптимальное решение, если количество примеров меньше 10^8.

Смысл этого примера в том, что признаки имеют совершенно разный масштаб. Эту проблему может решить нормировка. Существует много способов нор-

мировки признаков, один из самых простых – привести все признаки к диапазону от −1 до 1. Так, в приведенном выше примере, если мы разделим каждый признак на его максимальное значение, то получим $x_1 = (y + 0{,}5\alpha)/1{,}5$, $x_2 = y$. Тогда для $\lambda \leq 10^{-3}$ решение проблемы гребневой регрессии будет очень близко к $\mathbf{w}^*$.

Далее, границы обобщаемости, которые мы вывели в главе 13 для минимизации регуляризированной потери, зависят от нормы оптимального вектора $\mathbf{w}^*$ и от максимальной нормы векторов-образцов[1]. Поэтому в примере выше до нормировки признаком было $\|\mathbf{w}^*\|^2 = 10^8$, а после нормировки стало $\|\mathbf{w}^*\|^2 = 1$. Максимальная норма вектора-образца осталась приблизительно прежней, следовательно, нормировка существенно улучшает ошибку оценивания.

Нормировка признаков может также уменьшить время работы алгоритма обучения. Например, в разделе 14.5.3 было показано, как использовать алгоритм стохастического градиентного спуска (СГС) для решения задачи минимизации регуляризированной потери. Количество итераций, необходимое для сходимости СГС, также зависит от нормы $\mathbf{w}^*$ и от максимальной нормы $\|\mathbf{x}\|$. Следовательно, и в этом случае нормировка может заметно уменьшить время работы СГС.

Далее мы продемонстрируем, как простое преобразование признаков, например, отсечение, иногда позволяет уменьшить ошибку аппроксимации нашего класса гипотез. Снова рассмотрим проблему линейной регрессии с квадратичной потерей. Пусть $a > 1$ – большое число. Предположим, что метка y выбирается из равномерного распределения на множестве $\{\pm 1\}$, а затем единственному признаку x присваивается значение y с вероятностью $(1 - 1/a)$ и значение ay с вероятностью $1/a$. Это значит, что в большинстве случаев признак ограничен, но с очень малой вероятностью принимает очень большое значение. Тогда для любого w ожидаемая квадратичная потеря w равна

$$L_{\mathcal{D}}(w) = \mathbb{E}\frac{1}{2}(wx - y)^2 = \left(1 - \frac{1}{a}\right)\frac{1}{2}(wy - y)^2 + \frac{1}{a}\frac{1}{2}(wy - y)^2.$$

Решая это уравнение относительно w, находим, что $w^* = (2a - 1) / (a^2 + a - 1)$, т. е. стремится к нулю, когда a стремится к бесконечности. Поэтому значение целевой функции в точке w^* стремится к 0,5, когда a стремится к бесконечности. Например, при $a = 100$ получаем $L_{\mathcal{D}}(w^*) \geq 0{,}48$. Предположим далее, что мы применили отсечение, т. е. преобразование $x \mapsto \text{sign}(x)\min\{1, |x|\}$. При таком преобразовании w^* переходит в 1, а $L_{\mathcal{D}}(w^*) = 0$. Этот простой пример показывает, что простое преобразование может оказывать существенное влияние на ошибку аппроксимации.

Конечно, нетрудно придумать примеры, когда то же самое преобразование признаков только ухудшает качества и увеличивает ошибку аппроксимации.

[1] Точнее, выведенные в главе 1 границы для минимизации регуляризированной потери зависят от $\|\mathbf{w}^*\|^2$ и от липшицевости или гладкости функции потерь. Для линейных предикторов с функцией потери вида $\ell(\mathbf{w}, (x, y)) = \varphi(\langle \mathbf{w}, \mathbf{x} \rangle, y)$, где φ — выпуклая и либо 1-липшицева, либо 1-гладкая по первому аргументу, имеем, что ℓ либо $\|\mathbf{x}\|$-липшицева, либо $\|\mathbf{x}\|^2$-гладкая. Например, для квадратичной потери $\varphi(a, y) = 1/2(a - y)^2$ и $\ell(\mathbf{w}, (\mathbf{x}, y)) = 1/2(\langle \mathbf{w}, \mathbf{x} \rangle - y)^2$ является $\|\mathbf{x}\|^2$-гладкой по первому аргументу.

Это не удивительно, поскольку, как мы уже говорили, преобразования признаков должны опираться на априорные предположения о проблеме. В примере выше в роли априорного предположения, наводящего на мысль об использовании отсечения, могло выступать предположение о том, что признаки, которые принимают значения выше заранее заданного порога, не несут дополнительной полезной информации, и потому их можно усечь до этого порогового значения.

25.2.1. Примеры преобразований признаков

Перечислим несколько употребительных преобразований признаков. Обычно их имеет смысл комбинировать (например, центрирование + масштабирование). Ниже мы будем обозначать $\mathbf{f} = (f_1, \ldots, f_m) \in \mathbb{R}^m$ вектор значений признака f на m обучающих примерах, а $\bar{f} = (1/m) \sum_{i=1}^{m} f_i$ – эмпирическое среднее этого признака на тех же примерах.

Центрирование

Это преобразование делает среднее значение признака равным нулю: $f_i \leftarrow f_i - \bar{f}$.

Приведение к единичному диапазону

Это преобразование приводит все значения признаков к диапазону [0,1]. Формально, положим $f_{\max} = \max_i f_i$ и $f_{\min} = \min_i f_i$. Тогда преобразование имеет вид $f_i \leftarrow (f_i - f_{\min})/(f_{\max} - f_{\min})$. Аналогично можно привести значения признака к диапазону [−1, 1] с помощью преобразования $f_i \leftarrow 2\,(f_i - f_{\min})/(f_{\max} - f_{\min}) - 1$. Разумеется, легко заменить эти отрезки на $[0, b]$ или $[-b, b]$, где b – заданный пользователем параметр.

Стандартизация

В результате этого преобразования признак будет иметь нулевое среднее и единичную дисперсию. Формально, обозначим $\nu = (1/m) \sum_{i=1}^{m} (f_i - \bar{f})^2$ эмпирическую дисперсию признака. Тогда следует положить $f_i \leftarrow (f_i - \bar{f})/\sqrt{\nu}$.

Отсечение

Это преобразование отсекает большие и малые значения признака, например: $f_i \leftarrow \text{sign}(f_i) \max\{b, |f_i|\}$, где b – заданный пользователем параметр.

Сигмоидное преобразование

Как следует из названия, это преобразование применяет сигмоидную функцию к признаку, например: $f_i \leftarrow 1/(1 + \exp(bf_i))$, где b – заданный пользователем параметр. Это преобразование можно рассматривать как «мягкий» вариант отсечения: он слабо влияет на значения, близкие к нулю, но ведет себя подобно отсечению для значений, далеких от нуля.

Логарифмическое преобразование

Это преобразование $f_i \leftarrow \log(b + f_i)$, где b – заданный пользователем параметр. Оно широко используется для признак-«счетчиков». Например, предположим, что признак представляет количество вхождений слова в текстовый документ. Тогда разность между количеством вхождений 0 и 1 гораздо важнее, чем между 1000 и 1001 вхождением.

Замечание 25.5. В вышеупомянутых преобразованиях каждый признак преобразуется на основе его значения в обучающем наборе, независимо от значений других признаков. Иногда желательно задавать параметр преобразования с учетом других признаков. Важный пример – применение к признакам такого масштабирования, чтобы эмпирическое среднее некоторой нормы образцов стало равно 1.

25.3. Обучение признаков

До сих пор мы обсуждали отбор признаков и манипуляции с ними. В этих случаях мы начинаем с заранее определенного векторного пространства $\mathbb{R}^d$, представляющего признаки. Затем мы выбираем подмножество этих признаков (отбор признаков) или преобразуем отдельные признаки (преобразование признаков). В этом разделе мы опишем *обучение признаков*, когда вначале имеется некоторое пространство образцов $\mathcal{X}$ и требуется обучить функцию $\psi : \mathcal{X} \to \mathbb{R}^d$, отображающую образцы из $\mathcal{X}$ в d-мерные векторы признаков.

Смысл обучения признаков состоит в том, чтобы автоматизировать процесс поиска хорошего представления пространства входов. Согласно теореме об отсутствии бесплатных завтраков, мы должны включить какие-то априорные знания о распределении данных, если хотим построить хорошее представление признаков. В этом разделе мы познакомимся с несколькими подходами к обучению признаков и покажем, при каких условиях на истинное распределение данных описанные методы могут оказаться полезны.

В этой книге мы уже видели несколько полезных способов построения признаков. Например, в случае полиномиальной регрессии мы отображали исходные образцы в векторное пространство всех одночленов (см. раздел 9.2.2 в главе 9). Выполнив это отображение, мы обучали *линейный* предиктор на построенных признаках. Чтобы автоматизировать этот процесс, нужно было бы обучиться такому преобразованию $\psi : \mathcal{X} \to \mathbb{R}^d$, что композиция класса линейных предикторов с ψ дает хороший класс гипотез для рассматриваемой задачи.

Ниже мы опишем технику построения признаков, которая называется *обучение словаря*.

25.3.1. Обучение словаря с помощью автокодировщиков

Идея обучения словаря проистекает из часто используемого представления документов в виде «мешка слов». Если задан словарь $D = \{w_1, \ldots, w_k\}$, где каждый образец w_i – строка, представляющая одно слово, и имеется документ $(p_1, \ldots, p_d)$, где p_i – встречающиеся в нем слова, то мы представляем документ в виде вектора

$\mathbf{x} \in \{0, 1\}^k$, где $x_i = 1$, если $w_i = p_j$ для некоторого $j \in [d]$, и $x_i = 0$ в противном случае. Эмпирически замечено, что во многих задачах обработки текстов применение линейных предикторов к такому представлению дает отличные результаты. Интуитивно каждое слово можно представлять себе как признак, измеряющий некоторое отличительное свойство документа. Если заданы помеченные примеры (например, темы документов), то алгоритм обучения ищет линейный предиктор, которые взвешивает эти признаки, так чтобы правильная комбинация вхождений слов позволяла предсказать метку.

В области обработки текстов у слов и словаря имеется естественная семантика, но в других приложениях такого интуитивно понятного представления образца может и не быть. Рассмотрим, к примеру, распознавание объектов в компьютерном зрении. Здесь образцом является изображение, а наша цель – распознать, какие объекты присутствуют в этом изображении. Применение линейного предиктора к пиксельному представлению изображения не дает хорошего классификатора. Нам нужно отображение ψ, которое переводило бы пиксельное представление в мешок «визуальных слов», представляющих содержание изображения. Таким «визуальным словом» могло бы быть «в изображении присутствует глаз». Если бы у нас было подобное представление, то мы могли бы применить к нему линейный предиктор, чтобы обучить классификатор распознавать, например, лица. Возникает вопрос, как можно обучиться такому словарю «визуальных слов», чтобы представление изображения в виде мешка слов было полезно для предсказания объектов, присутствующих в изображении.

Первый, наивный, подход к обучению словаря опирается на алгоритм кластеризации (см. главу 22). Предположим, что мы обучили функцию $c : \mathcal{X} \to \{1, \ldots, k\}$, где $c(\mathbf{x})$ – кластер, которому принадлежит $\mathbf{x}$. Тогда кластеры можно интерпретировать как «слова», а образцы – как «документы». При этом документ $\mathbf{x}$ отображается в вектор $\psi(\mathbf{x}) \in \{0, 1\}^k$, где $\psi(\mathbf{x})_i = 1$ тогда и только тогда, когда $\mathbf{x}$ принадлежит i-му кластеру. Легко видеть, что применение линейного предиктора к $\psi(\mathbf{x})$ эквивалентно назначению одной и той же метки всем образцам, попавшим в один кластер. Более того, если кластеризация основана на расстоянии до центра кластера (как, например, в алгоритме k-средних), то линейный предиктор, построенный по $\psi(\mathbf{x})$, дает кусочно-постоянный предиктор по $\mathbf{x}$.

Алгоритм k-средних и метод PCA можно рассматривать как частные случаи более общего подхода к обучению словаря – *автокодировщиков*, когда мы обучаем две функции: «кодировщик» $\psi : \mathbb{R}^d \to \mathbb{R}^k$ и «декодер» $\varphi : \mathbb{R}^k \to \mathbb{R}^d$. Цель процесса обучения – найти такие две функции, чтобы ошибка реконструкции $\sum_i ||\mathbf{x}_i - \varphi(\psi(\mathbf{x}_i))||^2$ была как можно меньше. Конечно, можно тривиально положить $k = d$, и тогда ψ и φ станут тождественными отображениями, а реконструкция будет идеальной. Поэтому нужно наложить какие-то ограничения на ψ и φ. В случае PCA мы требуем, чтобы $k < d$ и чтобы ψ и φ были линейными функциями. В алгоритме k-средних k необязательно должно быть меньше d, но ψ и φ зависят от k центроидов, $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$: $\psi(\mathbf{x})$ возвращает индикаторный вектор, принадлежащий $\{0, 1\}^k$, который показывает ближайший к $\mathbf{x}$ центроид, а φ принимает индикаторный вектор и возвращает представленный им центроид.

Важное свойство построения в алгоритме k-средних, благодаря которому k может быть больше d, заключается в том, что ψ отображает образцы в *разреженные* векторы. Действительно, в алгоритме k-средних только одна координата

$\psi(\mathbf{x})$ отлична от нуля. Поэтому естественно обобщить это построение, потребовав, чтобы значениями ψ были векторы, содержащие не более s ненулевых элементов, где s – небольшое целое число. В частности, мы позволим функциям ψ и φ зависеть от $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$. Функция ψ отображает образец $\mathbf{x}$ в вектор $\psi(\mathbf{x}) \in \mathbb{R}^k$, где $\psi(\mathbf{x})$ содержит не более s ненулевых элементов. Функция $\varphi(\mathbf{v})$ определяется как $\sum_{i=1}^{k} v_i \boldsymbol{\mu}_i$. Как и раньше, наша цель – добиться малой ошибки реконструкции, поэтому мы можем определить

$$\psi(\mathbf{x}) = \operatorname*{argmin}_{\mathbf{v}} ||\mathbf{x} - \varphi(\mathbf{v})||^2 \quad \text{s.t.} \quad ||\mathbf{v}||_0 \leq s,$$

где $||\mathbf{v}||_0 = |\{j: v_j \neq 0\}|$. Заметим, что если $s = 1$ и мы дополнительно налагаем ограничение $||\mathbf{v}||_1 = 1$, то получается функция кодирования из алгоритма k-средних, т. е. $\psi(\mathbf{x})$ – индикаторный вектор, который определяет ближайший к $\mathbf{x}$ центроид. Для больших значений s сформулированная задача оптимизации становится вычислительно трудной. Поэтому на практике мы иногда используем регуляризацию по норме ℓ_1 вместо ограничения на разреженность и определяем ψ следующим образом:

$$\psi(\mathbf{x}) = \operatorname*{argmin}_{\mathbf{v}} [||\mathbf{x} - \varphi(\mathbf{v})||^2 + \lambda ||\mathbf{v}||_1],$$

где $\lambda > 0$ – параметр регуляризации. В любом случае проблема обучения словаря теперь сводится к нахождению таких векторов $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$, при которых ошибка реконструкции $\sum_{i=1}^{m} ||\mathbf{x}_i - \varphi(\psi(\mathbf{x}))||^2$ как можно меньше. Даже если ψ определена с использованием ℓ_1-регуляризации, эта задача все равно вычислительно трудна (как и задача k-средних). Но существует несколько эвристических алгоритмов поиска, дающих удовлетворительные решения. Правда, эти алгоритмы выходят за рамки книги.

25.4. Резюме

Во многих алгоритмах машинного обучения представление образцов с помощью признаков считается естественным. Тем не менее, выбор представления требует тщательного обдумывания. Мы обсудили различные подходы к отбору признаков, познакомились с фильтрами, алгоритмами жадного отбора и нормами, индуцирующими разреженность. Затем мы привели несколько примеров преобразований признаков и продемонстрировали их полезность. Наконец, мы обсудили обучение признаков и, в частности, обучение словаря. Мы показали, что и отбор признаков, и манипулирование ими, и обучение зависят от априорных знаний о данных.

25.5. Библиографические сведения

В работе Guyon and Elisseeff (2003) дан обзор нескольких процедур отбора признаков и, в частности, описаны многочисленные типы фильтров.

Прямые процедуры отбора признаков для минимизации выпуклой целевой функции при наличии полиэдрального ограничения восходят к алгоритму Фран-

ка–Вульфа (Frank & Wolfe, 1956). Связь с усилением изучалась несколькими авторами, в т. ч. Warmuth, Liao & Ratsch, 2006; Warmuth, Glocer & Vishwanathan, 2008; Shalev-Shwartz & Singer, 2008. Согласованное преследование изучалось специалистами по обработке сигналов (Mallat & Zhang, 1993). В нескольких работах проанализированы методы жадного отбора при различных условиях. См., например, статью Shalev-Shwartz, Zhang, and Srebro (2010) и приведенные в ней ссылки.

Использование нормы ℓ_1 как суррогата разреженности имеет долгую историю (см., например, работу Tibshirani (1996) и приведенные в ней ссылки), была проделана большая работа по пониманию связи между нормой ℓ_1 и разреженностью. Эта проблема тесно связана со сжатым измерением сигнала (см. главу 23). Идея разреживания предикторов с малой нормой ℓ_1 восходит к Морэ (Maurey) (Pisier 1980–1981). В разделе 26.4 мы покажем также, что низкую норму ℓ_1 можно использовать для ограничения сверху ошибки оценивания предиктора.

Обучение признаков и словаря активно изучалось в последнее время в контексте глубоких нейронных сетей. См., например, работы Le Cun & Bengio, 1995; Hinton et al., 2006; Ranzato et al., 2007; Collobert & Weston, 2008; Lee et al., 2009; Le et al., 2012; Bengio, 2009 и приведенные в них ссылки.

25.6. Упражнения

25.1. Докажите равенство (25.1). *Указание*. Пусть a^*, b^* доставляют минимум левой части. Найдите такие a, b, при которых целевая функция в правой части меньше, чем в левой. Сделайте то же самое в обратном направлении.

25.2. Докажите, что (25.7) является решением задачи (25.6).

25.3. AdaBoost как прямой алгоритм жадного отбора. Вспомните алгоритм AdaBoost из главы 10. В этом упражнении мы дадим еще одну интерпретацию AdaBoost как прямого алгоритма жадного отбора.

- Пусть дано множество m образцов $\mathbf{x}_1, \ldots, \mathbf{x}_m$ и класс гипотез $\mathcal{H}$ конечной VC-размерности. Покажите, что существуют d и $h_1, \ldots, h_d$ такие, что для любого $h \in \mathcal{H}$ существует $i \in [d]$ такое, что $h_i(\mathbf{x}_j) = h(\mathbf{x}_j)$ для всех $j \in [m]$.
- Пусть $R(\mathbf{w})$ определено, как в (25.3). При заданном векторе $\mathbf{w}$ определим функцию $f_{\mathbf{w}}$ как

$$f_{\mathbf{w}}(\cdot) = \sum_{i=1}^{d} w_i h_i(\cdot).$$

  Пусть $\mathbf{D}$ – следующее распределение на $[m]$:

$$D_i = \frac{\exp(-y_i f_{\mathbf{w}}(\mathbf{x}_i))}{Z},$$

  где Z – нормировочный коэффициент, гарантирующий, что $\mathbf{D}$ – вектор вероятностей. Покажите, что

$$\frac{\partial R(\mathbf{w})}{w_j} = -\sum_{i=1}^{m} D_i y_i h_j(\mathbf{x}_i).$$

Кроме того, обозначив $\epsilon_j = \sum_{i=1}^{m} D_i \mathbb{1}_{[h_j(\mathbf{x}_i) \neq y_i]}$, докажите, что

$$\frac{\partial R(\mathbf{w})}{w_j} = 2\epsilon_j - 1.$$

Сделайте отсюда вывод, что если $\epsilon_j \leq ½ - \gamma$, то $\left|\frac{\partial R(\mathbf{w})}{w_j}\right| \geq \gamma/2$.

- Покажите, что шаг обновления в алгоритме AdaBoost гарантирует, что $R(\mathbf{w}^{(t+1)}) - R(\mathbf{w}^{(t)}) \leq \log(\sqrt{1 - 4\gamma^2})$. *Указание:* воспользуйтесь доказательством теоремы 10.2.

ЛАНЬ®

ЛАНЬ®

Часть *IV*

ДОПОЛНИТЕЛЬНЫЕ ГЛАВЫ

РАДЕМАХЕРОВСКАЯ СЛОЖНОСТЬ

ЛАНЬ®

В главе 4 мы показали, что равномерная сходимость – достаточное условие обучаемости. В этой главе мы изучим радемахеровскую сложность, измеряющую скорость равномерной сходимости. Мы приведем границы обобщаемости, основанные на этой метрике.

26.1. Радемахеровская сложность

Напомним определение ϵ-репрезентативной выборки из главы 4.

Определение 26.1 (ϵ-репрезентативная выборка). Обучающий набор S называется ϵ-репрезентативным (относительно множества примеров Z, класса гипотез $\mathcal{H}$, функции потерь ℓ и распределения $\mathcal{D}$), если

$$\sup_{h \in \mathcal{H}} |L_{\mathcal{D}}(h) - L_S(h)| \leq \epsilon.$$

Мы показали, что если S – $\epsilon/2$-репрезентативная выборка, то правило ERM является ϵ-согласованным, т. е. $L_{\mathcal{D}}(\text{ERM}_{\mathcal{H}}(S)) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$.

Чтобы упростить нотацию, обозначим

$$\mathcal{F} \stackrel{\text{def}}{=} \ell \circ \mathcal{H} \stackrel{\text{def}}{=} \{z \mapsto \ell(h, z) : h \in \mathcal{H}\},$$

и для данной функции $f \in \mathcal{F}$ определим

$$L_{\mathcal{D}}(f) = \mathop{\mathbb{E}}_{z \sim \mathcal{D}}[f(z)], \quad L_S(f) = \frac{1}{m} \sum_{i=1}^{m} f(z_i).$$

Будем называть представительностью S относительно $\mathcal{F}$ наибольшее расхождение между истинной и эмпирической ошибкой функции f, т. е.

$$\text{Rep}_{\mathcal{D}}(\mathcal{F}, S) \stackrel{\text{def}}{=} \sup_{f \in \mathcal{F}} (L_{\mathcal{D}}(f) - L_S(f)). \tag{26.1}$$

Предположим теперь, что требуется оценить представительность S, используя только саму выборку S. Возникает простая мысль: разобьем S на два непересекающихся множества, $S = S_1 \cup S_2$, будем называть S_1 контрольным набором, а S_2 – обучающим набором. Тогда представительность S можно оценить как:

$$\sup_{f\in\mathcal{F}}(L_{S_1}(f) - L_{S_2}(f)). \tag{26.2}$$

Это выражение можно переписать более компактно, если определить $\sigma = (\sigma_1, \ldots, \sigma_m) \in \{\pm 1\}^m$ как такой вектор, что $S_1 = \{z_i : \sigma_i = 1\}$ и $S2 = \{z_i : \sigma_i = -1\}$. Если еще предположить, что $|S_1| = |S_2|$, то выражение (26.2) можно переписать в виде

$$\frac{2}{m}\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}\sigma_i f(z_i). \tag{26.3}$$

Рademахеровская сложность улавливает эту идею, рассматривая математическое ожидание выражения (26.3) относительно случайного выбора σ. Формально, обозначим $\mathcal{F} \circ S$ множество всех возможных комбинаций значений, которые функции $f \in \mathcal{F}$ могут принимать на выборке S, т. е.

$$\mathcal{F} \circ S = \{(f(z_1), \ldots, f(z_m)) : f \in \mathcal{F}\}.$$

Пусть случайные величины, принадлежащие σ, независимы и одинаково распределены с распределением $\mathbb{P}[\sigma_i = 1] = \mathbb{P}[\sigma_i = -1] = ½$. Тогда рademахеровская сложность $\mathcal{F}$ относительно S определяется следующим образом:

$$R(\mathcal{F} \circ S) \stackrel{\text{def}}{=} \frac{1}{m}\mathop{\mathbb{E}}_{\sigma\sim\{\pm 1\}^m}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}\sigma_i f(z_i)\right]. \tag{26.4}$$

Вообще для заданного множества векторов $A \subset \mathbb{R}^m$ определим

$$R(A) \stackrel{\text{def}}{=} \frac{1}{m}\mathop{\mathbb{E}}_{\sigma}\left[\sup_{\mathbf{a}\in A}\sum_{i=1}^{m}\sigma_i a_i\right]. \tag{26.4}$$

Следующая лемма ограничивает математическое ожидание представительности S удвоенной рademахеровской сложностью.

Лемма 26.2

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[\mathrm{Rep}_{\mathcal{D}}(\mathcal{F}, S)] \le 2\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} R(\mathcal{F} \circ S).$$

Доказательство. Пусть $S' = \{z'_1, \ldots, z'_m\}$ – еще одна независимая и одинаково распределенная выборка. Очевидно, что для всех $f \in \mathcal{F}$, $L_{\mathcal{D}}(f) = \mathbb{E}_{S'}[L_{S'}(f)]$. Следовательно, для любого $f \in \mathcal{F}$ имеем

$$(L_{\mathcal{D}}(f) - L_S(f)) = \mathop{\mathbb{E}}_{S'}[L_{S'}(f)] - L_S(f) = \mathop{\mathbb{E}}_{S'}[L_{S'}(f) - L_S(f)].$$

Взяв верхнюю грань обеих частей по $f \in \mathcal{F}$ и воспользовавшись тем фактом, что верхняя грань математического ожидания меньше математического ожидания верхней грани, получаем

$$\begin{aligned}\sup_{f\in\mathcal{F}}(L_{\mathcal{D}}(f) - L_S(f)) &= \sup_{f\in\mathcal{F}}\mathop{\mathbb{E}}_{S'}[L_{S'}(f) - L_S(f)].\\ &\le \mathop{\mathbb{E}}_{S'}[\sup_{f\in\mathcal{F}}(L_{S'}(f) - L_S(f))].\end{aligned}$$

Берем математическое ожидание обеих частей по S:

$$\mathop{\mathbb{E}}_{S}\left[\sup_{f\in\mathcal{F}}(L_{\mathcal{D}}(f)-L_S(f))\right]\le \mathop{\mathbb{E}}_{S,S'}\left[\sup_{f\in\mathcal{F}}(L_{S'}(f)-L_S(f))\right]$$
$$=\frac{1}{m}\mathop{\mathbb{E}}_{S,S'}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}(f(z_i')-f(z_i))\right]. \tag{26.6}$$

Далее заметим, что для любого j случайные величины z_j и z_j' независимы и одинаково распределены. Поэтому замена одной на другую не оказывает влияния на математическое ожидание:

$$\mathop{\mathbb{E}}_{S,S'}\left[\sup_{f\in\mathcal{F}}\left((f(z_j')-f(z_j))+\sum_{i\ne j}(f(z_i')-f(z_i))\right)\right]$$
$$=\mathop{\mathbb{E}}_{S,S'}\left[\sup_{f\in\mathcal{F}}\left((f(z_j)-f(z_j'))+\sum_{i\ne j}(f(z_i')-f(z_i))\right)\right]. \tag{26.7}$$

Пусть σ_j – случайная величина такая, что $\mathbb{P}[\sigma_j = 1] = \mathbb{P}[\sigma_j = -1] = ½$. Из равенства (26.7) получаем

$$\mathop{\mathbb{E}}_{S,S',\sigma_j}\left[\sup_{f\in\mathcal{F}}\left(\sigma_j(f(z_j')-f(z_j))+\sum_{i\ne j}(f(z_i')-f(z_i))\right)\right]$$
$$=\frac{1}{2}(\text{левая часть (26.7)})+\frac{1}{2}(\text{правая часть (26.7)})$$
$$=\mathop{\mathbb{E}}_{S,S'}\left[\sup_{f\in\mathcal{F}}\left((f(z_j')-f(z_j))+\sum_{i\ne j}(f(z_i')-f(z_i))\right)\right]. \tag{26.8}$$

Повторяя это для всех j, мы получим:

$$\mathop{\mathbb{E}}_{S,S'}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}(f(z_i')-f(z_i))\right]=\mathop{\mathbb{E}}_{S,S',\sigma}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^{m}\sigma_i(f(z_i')-f(z_i))\right]. \tag{26.9}$$

Наконец,

$$\sup_{f\in\mathcal{F}}\sum_i\sigma_i(f(z_i')-f(z_i))\le\sup_{f\in\mathcal{F}}\sum_i\sigma_i f(z_i')+\sup_{f\in\mathcal{F}}\sum_i-\sigma_i f(z_i),$$

и, поскольку вероятность σ такая же, как вероятность $-\sigma$, правую часть (26.9) можно ограничить сверху значением

$$\mathop{\mathbb{E}}_{S,S',\sigma}\left[\sup_{f\in\mathcal{F}}\sum_i\sigma_i f(z_i')+\sup_{f\in\mathcal{F}}\sum_i\sigma_i f(z_i)\right]$$
$$=m\mathop{\mathbb{E}}_{S'}[R(\mathcal{F}\circ S')]+m\mathop{\mathbb{E}}_{S}[R(\mathcal{F}\circ S)]=2m\mathop{\mathbb{E}}_{S}[R(\mathcal{F}\circ S)].$$ □

Из этой леммы сразу же следует, что правило ERM находит гипотезу, которая по математическому ожиданию близка к оптимальной гипотезе в классе $\mathcal{H}$.

Теорема 26.3. *Имеет место неравенство*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(\mathrm{ERM}_{\mathcal{H}}(S)) - L_S(\mathrm{ERM}_{\mathcal{H}}(S))] \le 2\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} R(\ell\circ\mathcal{H}\circ S).$$

Кроме того, для любой $h^*\in\mathcal{H}$

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(\mathrm{ERM}_{\mathcal{H}}(S)) - L_S(h^*)] \le 2\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} R(\ell\circ\mathcal{H}\circ S).$$

Кроме того, если $h^* = \mathrm{argmin}_h L_{\mathcal{D}}(h)$, *то для любого* $\delta\in(0, 1)$ *с вероятностью не менее* $1-\delta$ *для случайной выборки* S *имеем*

$$L_{\mathcal{D}}(\mathrm{ERM}_{\mathcal{H}}(S)) - L_{\mathcal{D}}(h^*) \le \frac{2\mathbb{E}_{S'\sim\mathcal{D}^m} R(\ell\circ\mathcal{H}\circ S)}{\delta}.$$

Доказательство. Первое неравенство вытекает непосредственно из леммы 26.2. Второе неравенство справедливо, потому что для любой фиксированной гипотезы h^*

$$L_{\mathcal{D}}(h^*) = \mathop{\mathbb{E}}_{S}[L_S(h^*)] \ge \mathop{\mathbb{E}}_{S}[L_S(\mathrm{ERM}_{\mathcal{H}}(S))].$$

Третье неравенство следует из предыдущего в силу неравенства Маркова (отметим, что случайная величина $L_{\mathcal{D}}(\mathrm{ERM}_{\mathcal{H}}(S)) - L_{\mathcal{D}}(h^*)$ неотрицательна). □

Далее мы выведем границы, аналогичные границам в теореме 26.3, но с лучшей зависимостью от параметра уверенности δ. Для этого сначала сформулируем следующее утверждение о концентрации для функций с ограниченной разностью.

Лемма 26.4. (неравенство Макдайрмида). *Пусть* V *– некоторое множество и* $f : V_m \to \mathbb{R}$ *– функция* m *переменных такая, что для некоторого* $c > 0$, *для всех* $i\in[m]$ *и для всех* $x_1, \ldots, x_m, x_i'\in V$ *имеет место неравенство*

$$|f(x_1, \ldots, x_m) - f(x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_m)| \le c.$$

Пусть $X_1, \ldots, X_m$ *– m независимых случайных величин, принимающих значения из V. Тогда с вероятностью не менее* $1-\delta$ *справедливо неравенство*

$$|f(X_1, \ldots, X_m) - \mathbb{E}[f(X_1, \ldots, X_m)]| \le c\sqrt{\ln\left(\frac{2}{\delta}\right)m/2}.$$

Из неравенства Макдайрмида мы можем вывести границы обобщаемости с лучшей зависимостью от параметра уверенности.

Теорема 26.5. *Предположим, что для любых* z *и* $h\in\mathcal{H}$ *имеет место* $|\ell(h, z)| \le c$. *Тогда*

1) *с вероятностью не менее* $1-\delta$ *для всех* $h\in\mathcal{H}$

$$L_{\mathcal{D}}(h) - L_S(h) \le 2\mathop{\mathbb{E}}_{S'\sim\mathcal{D}^m} R(\ell\circ\mathcal{H}\circ S') + c\sqrt{\frac{2\ln(2/\delta)}{m}}.$$

В частности, это справедливо для $h = \mathrm{ERM}_{\mathcal{H}}(S)$;

2) *с вероятностью не менее* $1 - \delta$ *для всех* $h \in \mathcal{H}$

$$L_{\mathcal{D}}(h) - L_S(h) \le 2R(\ell \circ \mathcal{H} \circ S) + c\sqrt{\frac{2\ln(4/\delta)}{m}}.$$

В частности, это справедливо для $h = \mathrm{ERM}_{\mathcal{H}}(S)$;

3) *с вероятностью не менее* $1 - \delta$ *для всех* $h \in \mathcal{H}$

$$L_{\mathcal{D}}(\mathrm{ERM}_{\mathcal{H}}(S)) - L_S(h) \le 2R(\ell \circ \mathcal{H} \circ S) + c\sqrt{\frac{2\ln(8/\delta)}{m}}.$$

Доказательство. Сначала заметим, что случайная величина $\mathrm{Rep}_{\mathcal{D}}(\mathcal{F}, S) = \sup_{h \in \mathcal{H}}(L_{\mathcal{D}}(h) - L_S(h))$ удовлетворяет условию ограниченных разностей из леммы 26.4 с константой $2c/m$. Объединяя границу из леммы 26.4 с леммой 26.2, получаем, что с вероятностью не менее $1 - \delta$

$$\mathrm{Rep}_{\mathcal{D}}(\mathcal{F}, S) \le \mathbb{E}\mathrm{Rep}_{\mathcal{D}}(\mathcal{F}, S) + c\sqrt{\frac{2\ln(2/\delta)}{m}} \le 2\mathop{\mathbb{E}}_{S'}R(\ell \circ \mathcal{H} \circ S') + c\sqrt{\frac{2\ln(2/\delta)}{m}}.$$

Первое неравенство теоремы следует из определения $\mathrm{Rep}_{\mathcal{D}}(\mathcal{F}, S)$. Для доказательства второго неравенства заметим, что случайная величина $(\ell \circ \mathcal{H} \circ S)$ также удовлетворяет условию ограниченных разностей из леммы 26.4 с константой $2c/m$. Поэтому второе неравенство вытекает из первого, леммы 26.4 и леммы о границе объединения. Наконец, для доказательства последнего неравенства обозначим $h_S = \mathrm{ERM}_{\mathcal{H}}(S)$ и заметим, что

$$\begin{aligned} &L_{\mathcal{D}}(h_S) - L_{\mathcal{D}}(h^*) \\ &\quad = L_{\mathcal{D}}(h_S) - L_S(h_S) + L_S(h_S) - L_S(h^*) + L_S(h^*) - L_{\mathcal{D}}(h^*) \\ &\quad \le (L_{\mathcal{D}}(h_S) - L_S(h_S)) + (L_S(h^*) - L_{\mathcal{D}}(h^*)). \end{aligned} \tag{26.10}$$

Первое слагаемое в правой части ограничено в силу второго неравенства теоремы. Для оценки второго слагаемого используем тот факт, что h^* не зависит от S, поэтому, в силу неравенства Хёфдинга, с вероятностью не менее $1 - \delta/2$ имеет место неравенство

$$L_S(h^*) - L_{\mathcal{D}}(h^*) \le c\sqrt{\frac{2\ln(4/\delta)}{m}}. \tag{26.11}$$

Применяя лемму о границе объединения, получаем требуемый результат. □

Эта теорема утверждает, что если величина $R(\ell \circ \mathcal{H} \circ S)$ мала, то класс $\mathcal{H}$ можно обучить по правилу ERM. Важно подчеркнуть, что границы в последних двух пунктах теоремы зависят от конкретного обучающего набора S. То есть мы используем S как для обучения гипотезы из $\mathcal{H}$, так и для оценки ее качества. Границы такого типа называются *зависимыми от данных*.

26.1.1. Исчисление Радемахера

Обсудим некоторые свойства радемахеровской сложности. Это поможет нам вывести простые границы $R(\ell \circ \mathcal{H} \circ S)$ для представляющих интерес частных случаев.

Следующая лемма – прямое следствие определение.

Лемма 26.6. *Для любого* $A \subset \mathbb{R}^m$, *скаляра* $c \in \mathbb{R}$ *и вектора* $\mathbf{a}_0 \in \mathbb{R}^m$ *имеем*

$$R(\{c\mathbf{a} + \mathbf{a}_0 : \mathbf{a} \in A\}) \leq |c|\, R(A).$$

Эта лемма означает, что сложность выпуклой оболочки A совпадает со сложностью A.

Лемма 26.7. *Пусть* A – *подмножество* $\mathbb{R}^m$, *и пусть* $A' = \{\sum_{j=1}^{N} \alpha_j \mathbf{a}^{(j)} : N \in \mathcal{H}, \forall j \mathbf{a}^{(j)} \in A, \alpha_j \geq 0, \|\alpha\|_1 = 1\}$. *Тогда* $R(A') = R(A)$.

Доказательство. Основная идея вытекает из того факта, что для любого вектора $\mathbf{v}$ имеет место равенство

$$\sup_{\boldsymbol{\alpha} \geq 0 : \|\boldsymbol{\alpha}\|_1 = 1} \sum_{j=1}^{N} \alpha_j v_j = \max_j v_j.$$

Следовательно,

$$\begin{aligned} mR(A') &= \mathbb{E}_{\boldsymbol{\sigma}} \sup_{\boldsymbol{\alpha} \geq 0 : \|\boldsymbol{\alpha}\|_1 = 1} \sup_{\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(N)}} \sum_{i=1}^{m} \sigma_j \sum_{j=1}^{N} \alpha_j a_i^{(j)} \\ &= \mathbb{E}_{\boldsymbol{\sigma}} \sup_{\boldsymbol{\alpha} \geq 0 : \|\boldsymbol{\alpha}\|_1 = 1} \sum_{j=1}^{N} \alpha_j \sup_{\mathbf{a}^{(j)}} \sum_{i=1}^{m} \sigma_i a_i^{(j)} \\ &= \mathbb{E}_{\boldsymbol{\sigma}} \sup_{\mathbf{a} \in A} \sum_{i=1}^{m} \sigma_i a_i \\ &= mR(A), \end{aligned}$$

что и завершает доказательство. □

Следующая лемма, доказанная Массартом (Massart), утверждает, что радемахеровская сложность конечного множества логарифмически зависит от размера множества.

Лемма 26.8 (лемма Массарта). Пусть $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_N\}$ – конечное множество векторов в $\mathbb{R}^m$. Определим $\bar{\mathbf{a}} = (1/N) \sum_{i=1}^{N} \mathbf{a}_i$. Тогда

$$R(A) \leq \max_{\mathbf{a} \in A} \|\mathbf{a} - \bar{\mathbf{a}}\| \frac{\sqrt{2\log(N)}}{m}.$$

Доказательство. В силу леммы 26.6, можно без ограничения общности считать, что $\bar{\mathbf{a}} = 0$. Пусть $\lambda > 0$ и $A' = \{\lambda \mathbf{a}_1, \ldots, \lambda \mathbf{a}_N\}$. Ограничим сверху радемахеровскую сложность следующим образом:

$$
\begin{aligned}
mR(A') &= \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\max_{\mathbf{a}\in A'}\langle \boldsymbol{\sigma}, \mathbf{a}\rangle\right] = \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\log\left(\max_{\mathbf{a}\in A'} e^{\langle \boldsymbol{\sigma}, \mathbf{a}\rangle}\right)\right] \\
&\le \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\log\left(\sum_{\mathbf{a}\in A'} e^{\langle \boldsymbol{\sigma}, \mathbf{a}\rangle}\right)\right] \\
&\le \log\left[\mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left(\sum_{\mathbf{a}\in A'} e^{\langle \boldsymbol{\sigma}, \mathbf{a}\rangle}\right)\right] \quad \text{// неравенство Йенсена} \\
&= \log\left(\sum_{\mathbf{a}\in A'}\prod_{i=1}^{m}\mathop{\mathbb{E}}_{\sigma_i}[e^{\sigma_i a_i}]\right),
\end{aligned}
$$

где последнее равенство вытекает из независимости рademахеровских переменных. Далее, согласно лемме А.6, для любых $a_i \in \mathbb{R}$

$$
\mathop{\mathbb{E}}_{\sigma_i} e^{\sigma_i a_i} = \frac{\exp(a_i) + \exp(-a_i)}{2} \le \exp(a_i^2/2),
$$

и потому

$$
\begin{aligned}
mR(A') &\le \log\left(\sum_{\mathbf{a}\in A'}\prod_{i=1}^{m}\exp\left(\frac{a_i^2}{2}\right)\right) = \log\left(\sum_{\mathbf{a}\in A'}\exp(\|\mathbf{a}\|^2/2)\right) \\
&\le \log\left(|A'|\max_{\mathbf{a}\in A'}\exp(\|\mathbf{a}\|^2/2)\right) = \log(|A'|)\max_{\mathbf{a}\in A'}(\|\mathbf{a}\|^2/2).
\end{aligned}
$$

Поскольку $R(A) = (1/\lambda)\, R(A')$, получаем отсюда, что

$$
R(A) \le \frac{\log(|A|) + \lambda^2 \max_{\mathbf{a}\in A}(\|\mathbf{a}\|^2/2)}{\lambda m}.
$$

Полагая $\lambda = \sqrt{2\log(|A|)/\max_{\mathbf{a}\in A}\|\mathbf{a}\|^2}$ и перегруппировывая члены, завершаем доказательство. □

Следующая лемма показывает, что композиция A с липшицевой функцией не приводит к резкому росту рademахеровской сложности. Доказательство предложено Какаде (Kakade) и Тевари (Tewari).

Лемма 26.9 (лемма о сжатии). *Для любого $i \in [m]$ пусть $\varphi_i : \mathbb{R} \to \mathbb{R}$ – ρ-липшицева функция, т. е. для любых $\alpha, \beta \in \mathbb{R}$ имеет место неравенство $|\varphi_i(\alpha) - \varphi_i(\beta)| \le \rho|\alpha - \beta|$. Для $\mathbf{a} \in \mathbb{R}^m$ обозначим $\varphi(\mathbf{a})$ вектор $(\varphi_1(a_1), \ldots, \varphi_m(y_m))$ и пусть $\varphi \circ A = \{\varphi(\mathbf{a}) : a \in A\}$. Тогда $R(\varphi \circ A) \le \rho R(A)$.*

Доказательство. Для простоты докажем лемму для случая $\rho = 1$. Если $\rho \ne 1$, то достаточно определить $\varphi' = (1/\rho)\varphi$, а затем воспользоваться леммой 26.6. Положим $A_i = \{(a_1, \ldots, a_{i-1}, \varphi_i(a_i), a_{i+1}, \ldots, a_m) : \mathbf{a} \in A\}$. Очевидно, достаточно доказать, что для любого множества A и любого i имеет место $R(A_i) \le R(A)$. Без ограничения общности мы докажем это утверждение для $i = 1$, а чтобы упростить обозначения, будем опускать нижний индекс в φ_1. Имеем

$$
\begin{aligned}
mR(A_1) &= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{a}\in A_1}\sum_{i=1}^{m}\sigma_i a_i\right] \\
&= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{a}\in A}\sigma_1\varphi(a_1)+\sum_{i=2}^{m}\sigma_i a_i\right] \\
&= \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}\left[\sup_{\mathbf{a}\in A}\left(\varphi(a_1)+\sum_{i=2}^{m}\sigma_i a_i\right)+\sup_{\mathbf{a}\in A}\left(-\varphi(a_1)+\sum_{i=2}^{m}\sigma_i a_i\right)\right] \\
&= \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}\left[\sup_{\mathbf{a},\mathbf{a}'\in A}\left(\varphi(a_1)-\varphi(a_1')+\sum_{i=2}^{m}\sigma_i a_i+\sum_{i=2}^{m}\sigma_i a_i'\right)\right] \\
&\le \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}\left[\sup_{\mathbf{a},\mathbf{a}'\in A}\left(|a_1-a_1'|+\sum_{i=2}^{m}\sigma_i a_i+\sum_{i=2}^{m}\sigma_i a_i'\right)\right], \qquad (26.12)
\end{aligned}
$$

где в последнем неравенстве мы воспользовались предположением о липшицевости φ. Далее заметим, что абсолютную величину $|a_1 - a_1'|$ в этом выражении можно опустить, поскольку $\mathbf{a}$ и $\mathbf{a}'$ берутся из одного и того же множества A, а остальная часть выражения под знаком верхней грани не меняется от замены $\mathbf{a}$ на $\mathbf{a}'$. Поэтому

$$
mR(A_1) = \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}\left[\sup_{\mathbf{a},\mathbf{a}'\in A}\left(a_1-a_1'+\sum_{i=2}^{m}\sigma_i a_i+\sum_{i=2}^{m}\sigma_i a_i'\right)\right]. \qquad (26.13)
$$

Но, пользуясь теми же тождествами, что в (26.12), легко видеть, что правая часть (26.13) в точности равна $mR(A)$, что и завершает доказательство. □

26.2. Рademахеровская сложность линейных классов

В этом разделе мы проанализируем рademахеровскую сложность линейных классов. Чтобы упростить вывод, с самого начала определим такие два класса:

$$
\mathcal{H}_1 = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \|\mathbf{w}\|_1 \le 1\},\ \mathcal{H}_2 = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \|\mathbf{w}\|_2 \le 1\}. \qquad (26.14)
$$

Следующая лемма ограничивает рademахеровскую сложность класса $\mathcal{H}_2$. Векторы $\mathbf{x}_i$ могут принадлежать любому гильбертову пространству (даже бесконечномерному), а граница не зависит от размерности этого пространства. Эти свойства полезны для анализа ядерных методов.

Лемма 26.10. *Пусть* $S = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$ *– векторы в гильбертовом пространстве. Определим* $\mathcal{H}_2 \circ S = \{(\langle \mathbf{w}, \mathbf{x}_1\rangle, \ldots, \langle \mathbf{w}, \mathbf{x}_m\rangle) : \|\mathbf{w}\|_2 \le 1\}$. *Тогда*

$$
R(\mathcal{H}_2 \circ S) \le \frac{\max_i \|\mathbf{x}_i\|_2}{\sqrt{m}}.
$$

Доказательство. В силу неравенства Коши–Буняковского мы знаем, что для любых векторов $\mathbf{w}$, $\mathbf{v}$ справедливо соотношение $\langle \mathbf{w}, \mathbf{v}\rangle \le \|\mathbf{w}\|\,\|\mathbf{v}\|$. Поэтому

$$
\begin{aligned}
mR(\mathcal{H}_2 \circ S) &= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{a}\in\mathcal{H}_2\circ S}\sum_{i=1}^{m}\sigma_i a_i\right] \\
&= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{w}:\|\mathbf{w}\|\le 1}\sum_{i=1}^{m}\sigma_i\langle \mathbf{w}, \mathbf{x}_i\rangle\right] \\
&= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{w}:\|\mathbf{w}\|\le 1}\left\langle \mathbf{w}, \sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\rangle\right] \\
&\le \mathbb{E}_{\sigma}\left[\left\|\sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\|_2\right]. \qquad (26.15)
\end{aligned}
$$

Применение неравенства Йенсена дает

$$
\mathbb{E}_{\sigma}\left[\left\|\sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\|_2\right] = \mathbb{E}_{\sigma}\left[\left(\left\|\sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\|_2^2\right)^{1/2}\right] \le \left(\mathbb{E}_{\sigma}\left[\left\|\sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\|_2^2\right]\right)^{1/2}. \qquad (26.16)
$$

Наконец, поскольку величины $\sigma_1, \ldots, \sigma_m$ независимы, имеем

$$
\begin{aligned}
\mathbb{E}_{\sigma}\left[\left\|\sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\|_2^2\right] &= \mathbb{E}_{\sigma}\left[\sum_{i=1}^{m}\sigma_i\sigma_j\langle \mathbf{x}_i, \mathbf{x}_j\rangle\right] \\
&= \sum_{i\ne j}\langle \mathbf{x}_i, \mathbf{x}_j\rangle\mathbb{E}_{\sigma}[\sigma_i\sigma_j] + \sum_{i=1}^{m}\langle \mathbf{x}_i, \mathbf{x}_j\rangle\mathbb{E}_{\sigma}[\sigma_i^2] \\
&= \sum_{i=1}^{m}\|\mathbf{x}_i\|_2^2 \le m\max_i\|\mathbf{x}_i\|_2^2.
\end{aligned}
$$

Для завершения доказательства достаточно объединить (26.15) и (26.16). □

Далее мы ограничим рademахеровскую сложность $\mathcal{H}_1 \circ S$.

Лемма 26.11. *Пусть* $S = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$ *– векторы в* $\mathbb{R}^n$. Тогда

$$
R(\mathcal{H}_1 \circ S) \le \max_i\|\mathbf{x}_i\|_\infty\sqrt{\frac{2\log(2n)}{m}}.
$$

Доказательство. Согласно неравенству Гёльдера, для любых векторов $\mathbf{w}$, $\mathbf{v}$ справедливо соотношение $\langle \mathbf{w}, \mathbf{v}\rangle \le \|\mathbf{w}\|_1\|\mathbf{v}\|_\infty$. Следовательно,

$$
\begin{aligned}
mR(\mathrm{H}_1 \circ S) &= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{a}\in\mathcal{H}_1\circ S}\sum_{i=1}^{m}\sigma_i a_i\right] \\
&= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{w}:\|\mathbf{w}\|_1\le 1}\sum_{i=1}^{m}\sigma_i\langle \mathbf{w}, \mathbf{x}_i\rangle\right] \\
&= \mathbb{E}_{\sigma}\left[\sup_{\mathbf{w}:\|\mathbf{w}\|_1\le 1}\left\langle \mathbf{w}, \sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\rangle\right] \\
&\le \mathbb{E}_{\sigma}\left[\left\|\sum_{i=1}^{m}\sigma_i\mathbf{x}_i\right\|_\infty\right]. \qquad (26.17)
\end{aligned}
$$

Для любого $j \in [n]$ положим $\mathbf{v}_j = (x_{1,j}, \ldots, x_{m,j}) \in \mathbb{R}^m$. Заметим, что $\|\mathbf{v}_j\|_2 \le \sqrt{m}\max_i \|\mathbf{x}_i\|_\infty$. Положим $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_n, -\mathbf{v}_1, \ldots, -\mathbf{v}_n\}$. Правая часть неравенства (26.17) равна $mR(V)$. По лемме Массарта (лемма 26.8) имеем

$$R(V) \le \max_i \|\mathbf{x}_i\|_\infty \sqrt{2\log(2n)/m},$$

что и требовалось доказать. □

26.3. Границы обобщаемости метода SVM

В этом разделе мы воспользуемся радемахеровской сложностью, чтобы вывести границы обобщаемости обобщенных линейных предикторов с ограничениями на евклидову норму. Мы покажем, как отсюда вытекают границы обобщаемости метода SVM с жестким и мягким зазором.

Мы будем рассматривать следующую общую постановку задачи с ограничениями. Пусть $\mathcal{H} = \{\mathbf{w} : \|\mathbf{w}\|_2 \le B\}$ – наш класс гипотез, а $Z = \mathcal{X} \times \mathcal{Y}$ – множество примеров. Предположим, что функция потерь $\ell : \mathcal{H} \times Z \to \mathbb{R}$ имеет вид

$$\ell(\mathbf{w}, (\mathbf{x}, y)) = \varphi(\langle \mathbf{w}, \mathbf{x}\rangle, y), \tag{26.18}$$

где $\varphi : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$ такова, что для любого $y \in \mathcal{Y}$ скалярная функция $a \mapsto \varphi(a, y)$ является ρ-липшицевой. Например, кусочно-линейную функцию потерь $\ell(\mathbf{w}, (\mathbf{x}, y)) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}$ можно записать в виде (26.18), полагая $\varphi(a, y) = \max\{0, 1 - ya\}$ и заметив, что φ является 1-липшицевой для всех $y \in \{\pm 1\}$. Другой пример – функция потерь по абсолютной величине, $\ell(\mathbf{w}, (\mathbf{x}, y)) = |\langle \mathbf{w}, \mathbf{x}\rangle - y|$, которую можно записать в виде (26.18), взяв $\varphi(a, y) = |a - y|$, тоже являющуюся 1-липшицевой для всех $y \in \mathbb{R}$.

Следующая теорема ограничивает ошибку обобщаемости всех предикторов из класса $\mathcal{H}$ на основе их эмпирической ошибки.

Теорема 26.12. *Предположим, что $\mathcal{D}$ – распределение на $\mathcal{X} \times \mathcal{Y}$ такое, что с вероятностью 1 имеет место $\|\mathbf{x}\|_2 \le R$. Пусть $\mathcal{H} = \{\mathbf{w} : \|\mathbf{w}\|_2 \le B\}$ и $\ell : \mathcal{H} \times Z \to \mathbb{R}$ – функция потерь вида (26.18) такая, что для любого $y \in \mathcal{Y}$ функция $a \mapsto \varphi(a, y)$ является ρ-липшицевой и удовлетворяет условию $\max_{a \in [-BR, BR]} |\varphi(a, y)| \le c$. Тогда для любого $\delta \in (0, 1)$ с вероятностью не менее $1 - \delta$ для независимой и одинаково распределенной выборки размера m*

$$\forall \mathbf{w} \in \mathcal{H}_i, \quad L_{\mathcal{D}}(\mathbf{w}) \le L_S(\mathbf{w}) + \frac{2\rho BR}{\sqrt{m}} + c\sqrt{\frac{2\ln(2/\delta)}{m}}.$$

Доказательство. Пусть $F = \{(\mathbf{x}, y) \mapsto \varphi(\langle \mathbf{w}, \mathbf{x}\rangle, y) : \mathbf{w} \in \mathcal{H}\}$. Мы покажем, что с вероятностью 1 $R(F \circ S) \le \rho BR/\sqrt{m}$, и тогда теорема будет следовать из теоремы 26.5. Действительно, множество $F \circ S$ можно записать в виде

$$F \circ S = \{(\varphi(\langle \mathbf{w}, \mathbf{x}_1\rangle, y_1), \ldots, \varphi(\langle \mathbf{w}, \mathbf{x}_m\rangle, y_m)) : \mathbf{w} \in \mathcal{H}\},$$

и верхняя граница $R(F \circ S)$ вытекает из лемм 26.9 и 26.10 и предположения о том, что $\|\mathbf{x}\|_2 \le R$ с вероятностью 1. □

Далее мы выведем границу обобщаемости для SVM с жестким зазором (Hard-SVM), основываясь на только что доказанной теореме. Для простоты откажемся от члена смещения и будем рассматривать постановку проблемы Hard-SVM в таком виде:

$$\operatorname*{argmin}_{\mathbf{w}} \|\mathbf{w}\|^2 \quad \text{при ограничениях} \quad \forall i,\, y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1. \tag{26.19}$$

Теорема 26.13. *Рассмотрим распределение $\mathcal{D}$ на $\mathcal{X} \times \{\pm 1\}$ такое, что существует вектор $\mathbf{w}^*$, для которого $\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}[y\langle \mathbf{w}^*, \mathbf{x}\rangle \geq 1] = 1, u\|\mathbf{x}\|_2 \leq R$ с вероятностью 1. Обозначим $\mathbf{w}_S$ решение задачи (26.19). Тогда с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ имеет место неравенство*

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}[y \neq sign(\langle \mathbf{w}_S, \mathbf{x}\rangle) \leq \frac{2R\|\mathbf{w}^*\|}{\sqrt{m}} + (1 + R\|\mathbf{w}^*\|)\sqrt{\frac{2\ln(2/\delta)}{m}}.$$

Доказательство. В доказательстве этой теореме предполагается использование рамповой функции потерь (см. раздел 15.2.3). Отметим, что область значений рамповой функции потерь – отрезок [0, 1] и что она является 1-липшицевой. Поскольку рамповая потеря ограничивает бинарную функцию потерь сверху, имеем

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}[y \neq sign(\langle \mathbf{w}_S, \mathbf{x}\rangle) \leq L_{\mathcal{D}}(\mathbf{w}_S).$$

Пусть $B = \|\mathbf{w}^*\|_2$ и рассмотрим класс $\mathcal{H} = \{\mathbf{w} : \|\mathbf{w}\|_2 \leq B\}$. Из определения Hard-SVM и нашего предположения о распределении вытекает, что $\mathbf{w}_S \in \mathcal{H}$ с вероятностью 1 и что $L_S(\mathbf{w}_S) = 0$. Поэтому согласно теореме 26.12

$$L_{\mathcal{D}}(\mathbf{w}_S) \leq L_S(\mathbf{w}_S) + \frac{2BR}{\sqrt{m}} + \sqrt{\frac{2\ln(2/\delta)}{m}}. \qquad \square$$

Замечание 26.1. Из теоремы 26.13 следует, что выборочная сложность алгоритма Hard-SVM растет как $(R^2\|\mathbf{w}^*\|^2)/\epsilon^2$. Применив более тонкий анализ и предположение о разделимости, можно улучшить порядок этой границы до $(R^2\|\mathbf{w}^*\|^2)/\epsilon$.

Граница в доказанной выше теореме зависит от нормы $\|\mathbf{w}^*\|$, которая нам неизвестна. Далее мы выведем границу, которая зависит от нормы выхода алгоритма SVM, поэтому ее можно будет вычислить, зная только сам обучающий набор. Доказательство аналогично выводу границ для структурной минимизации риска (SRM).

Теорема 26.14. *Предположим, что выполняются условия теоремы 26.13. Тогда с вероятностью не менее $1 - \delta$ для случайной выборки $S \sim \mathcal{D}^m$ имеет место неравенство*

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}[y \neq sign(\langle \mathbf{w}_S, \mathbf{x}\rangle) \leq \frac{4R\|\mathbf{w}_S\|}{\sqrt{m}} + \sqrt{\frac{\ln\left(\frac{4\log_2(\|\mathbf{w}_S\|)}{\delta}\right)}{m}}.$$

Доказательство. Для любого целого i обозначим $B_i = 2^i$, $\mathcal{H}_i = \{\mathbf{w} : \|\mathbf{w}\| \leq B_i\}$ и положим $\delta_i = \delta/(2i^2)$. Зафиксируем i. Тогда по теореме 26.12 с вероятностью не менее $1 - \delta_i$

$$\forall \mathbf{w} \in \mathcal{H}_i, \quad L_{\mathcal{D}}(\mathbf{w}) \leq L_S(\mathbf{w}) + \frac{2B_i R}{\sqrt{m}} + \sqrt{\frac{2\ln(2/\delta_i)}{m}}.$$

Применяя лемму о границе объединения и пользуясь тем фактом, что $\sum_{i=1}^{\infty} \delta_i \leq \delta$, получаем, что с вероятностью не менее $1 - \delta$ это неравенство справедливо для всех i. Поэтому для любого $\mathbf{w}$, если положить $i = \lceil \log_2(\|\mathbf{w}\|) \rceil$, то $\mathbf{w} \in \mathcal{H}_i$, $B_i \leq 2\|\mathbf{w}\|$ и $2/\delta_i = (2i)^2/\delta \leq (4\log_2(\|\mathbf{w}\|))^2/\delta$. Следовательно,

$$\begin{aligned} L_{\mathcal{D}}(\mathbf{w}) &\leq L_S(\mathbf{w}) + \frac{2B_i R}{\sqrt{m}} + \sqrt{\frac{2\ln(2/\delta_i)}{m}} \\ &\leq L_S(\mathbf{w}) + \frac{4\|\mathbf{w}\| R}{\sqrt{m}} + \sqrt{\frac{4(\ln(4\log_2(\|\mathbf{w}\|) + \ln(1/\delta))}{m}}. \end{aligned}$$

В частности, это справедливо для $\mathbf{w}_S$, что и завершает наше доказательство. □

Замечание 26.2. Отметим, что все выведенные нами границы не зависят от размерности $\mathbf{w}$. Это свойство используется при обучении SVM с ядрами, когда размерность $\mathbf{w}$ может быть очень велика.

26.4. Границы обобщаемости для предикторов с малой нормой ℓ_1

В предыдущем разделе мы вывели границы обобщаемости для линейных предикторов с ограничением на норму ℓ_2. В этом разделе мы рассмотрим следующую общую постановку задачи с ограничением на норму ℓ_1. Пусть $\mathcal{H} = \{\mathbf{w} : \|\mathbf{w}\|_1 \leq B\}$ – наш класс гипотез и $Z = \mathcal{X} \times \mathcal{Y}$ – множество примеров. Предположим, что функция потерь $\ell : \mathcal{H} \times Z \to \mathbb{R}$ имеет вид (26.18), где функция $\varphi : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$ является ρ-липшицевой по первому аргументу. Следующая теорема дает верхнюю границу ошибки обобщения всех предикторов из $\mathcal{H}$ на основе их эмпирической ошибки.

Теорема 26.15. *Предположим, что $\mathcal{D}$ – распределение на $\mathcal{X} \times \mathcal{Y}$ такое, что с вероятностью 1 имеет место $\|\mathbf{x}\|_\infty \leq R$. Пусть $\mathcal{H} = \{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\|_1 \leq B\}$ и $\ell : \mathcal{H} \times Z \to \mathbb{R}$ – функция потерь вида (26.18) такая, что для любого $y \in \mathcal{Y}$ функция $a \mapsto \varphi(a, y)$ является ρ-липшицевой и удовлетворяет условию $\max_{a \in [-BR, BR]} |\varphi(a, y)| \leq c$. Тогда для любого $\delta \in (0, 1)$ с вероятностью не менее $1 - \delta$ для независимой и одинаково распределенной выборки размера m*

$$\forall \mathbf{w} \in \mathcal{H}_i, \quad L_{\mathcal{D}}(\mathbf{w}) \leq L_S(\mathbf{w}) + 2\rho BR\sqrt{\frac{2\log(2d)}{m}} + c\sqrt{\frac{2\ln(2/\delta)}{m}}.$$

Доказательство. Доказательство проводится точно так же, как для теоремы 26.12, только опирается на лемму 26.11, а не 26.10. □

Интересно сравнить две границы: из теоремы 26.12 и теоремы 26.15. Если не считать дополнительного множителя $\log(d)$ в теореме 26.15, то, на первый взгляд, они похожи. Однако параметры B и R имеют в них разный смысл. В теореме 26.12

параметр B налагает ограничение на норму ℓ_2 вектора $\mathbf{w}$, а параметр R отражает предположение о малой величине нормы ℓ_2 образцов. С другой стороны, в теореме 26.15 параметр B налагает ограничение норму ℓ_1 вектора $\mathbf{w}$, а параметр R отражает предположение о малой величине нормы ℓ_∞ образцов (оно слабее, чем предположение о малой норме ℓ_2). Поэтому выбор ограничения должен зависеть от наших априорных знаний о множестве образцов и от априорных предположений о хороших предикторах.

26.5. Библиографические сведения

Использование радемахеровской сложности для ограничения скорости равномерной сходимости изучалось в работах Koltchinskii & Panchenko, 2000; Bartlett & Mendelson, 2001; Bartlett & Mendelson, 2002. В качестве дополнительной литературы рекомендуем, например, Bousquet, 2002; Boucheron, Bousquet & Lugosi, 2005; Bartlett, Bousquet & Mendelson, 2005. Наше доказательство леммы о концентрации заимствовано из записок к лекциям Какаде и Тевари. В работе Kakade, Sridharan and Tewari (2008) описан унифицированный подход к выводу границ радемахеровской сложности линейных классов при различных предположениях о нормах.

ЛАНЬ®

ЧИСЛА ПОКРЫТИЯ

В этой главе мы опишем еще один способ измерения сложности множеств – числа покрытия.

27.1. Покрытие

Определение 27.1 (покрытие)**.** Пусть $A \subset \mathbb{R}^m$ – множество векторов. Говорят, что A r-покрыто множеством A' относительно евклидовой метрики, если для всех $\mathbf{a} \in A$ существует $\mathbf{a}' \in A'$, для которого $||\mathbf{a} - \mathbf{a}'|| \leq r$. Обозначим $N(r, A)$ мощность наименьшего множества A', являющегося r-покрытием A.

Пример 27.1 (подпространство)**.** Предположим, что $A \subset \mathbb{R}^m$, положим $c = \max_{\mathbf{a} \in A} ||\mathbf{a}||$ и предположим, что A находится в d-мерном подпространстве $\mathbb{R}^m$. Тогда $N(r, A) \leq (2c\sqrt{d}/r)^d$. Чтобы убедиться в этом, рассмотрим ортонормированный базис подпространства $\mathbf{v}_1, \ldots, \mathbf{v}_d$. Тогда любой вектор $\mathbf{a} \in A$ можно представить в виде $\mathbf{a} = \sum_{i=1}^d \alpha_i \mathbf{v}_i$, где $||\alpha||_\infty \leq ||\alpha||_2 = ||\mathbf{a}||_2 \leq c$. Пусть $\epsilon \in \mathbb{R}$, рассмотрим множество

$$A' = \left\{ \sum_{i=1}^d \alpha_i' \mathbf{v}_i : \forall i, \alpha_i' \in \{-c, -c+\epsilon, -c+2\epsilon, \ldots, c\} \right\}.$$

Если задан $\mathbf{a} \in A$ такой, что $\sum_{i=1}^d \alpha_i \mathbf{v}_i$, где $||\alpha||_\infty \leq c$, то существует $\mathbf{a}' \in A'$ такой, что

$$\|\mathbf{a} - \mathbf{a}'\|^2 = \left\| \sum_i (\alpha_i' - \alpha_i) \mathbf{v}_i \right\|^2 \leq \epsilon^2 \sum_i \|\mathbf{v}_i\|^2 \leq \epsilon^2 d.$$

Выберем $\epsilon = r/\sqrt{d}$, тогда $||\mathbf{a} - \mathbf{a}'|| \leq r$ и, следовательно, A' – r-покрытие A. Поэтому

$$N(r, A) \leq |A'| = \left(\frac{2c}{\epsilon}\right)^d = \left(\frac{2c\sqrt{d}}{r}\right)^d.$$

27.1.1. Свойства

Следующая лемма – прямое следствие определения.

Лемма 27.2. *Для любого $A \subset \mathbb{R}^m$, скаляра $c > 0$ и вектора $\mathbf{a}_0 \in \mathbb{R}^m$ справедливо утверждение*

$$\forall r > 0 N(r, \{c\mathbf{a} + \mathbf{a}_0 : \mathbf{a} \in A\}) \le N(cr, A).$$

Далее мы докажем принцип сжатия.

Лемма 27.3. *Для любого $i \in [m]$ пусть $\varphi_i : \mathbb{R} \to \mathbb{R}$ – ρ-липшицева функция, т. е. для любых $\alpha, \beta \in \mathbb{R}$ имеет место неравенство $|\varphi_i(\alpha) - \varphi_i(\beta)| \le \rho|\alpha - \beta|$. Для любого $\mathbf{a} \in \mathbb{R}^m$ обозначим $\varphi(\mathbf{a})$ вектор $(\varphi_1(a_1), \ldots, \varphi_m(a_m))$ и положим $\varphi \circ A = \{\varphi(\mathbf{a}) : \mathbf{a} \in A\}$. Тогда*

$$N(\rho r, \varphi \circ A) \le N(r, A).$$

Доказательство. Определим $B = \varphi \circ A$. Пусть A' – r-покрытие A и $B' = \varphi \circ A'$. Тогда для любого $\mathbf{a} \in A$ существует $\mathbf{a}' \in A'$, для которого $\|\mathbf{a} - \mathbf{a}'\| \le r$. Следовательно,

$$\|\varphi(\mathbf{a}) - \varphi(\mathbf{a}')\|^2 = \sum_i (\varphi_i(a_i) - \varphi_i(a_i'))^2 \le \rho^2 \sum_i (a_i - a_i')^2 \le (\rho r)^2.$$

Поэтому B' является (ρr)-покрытием B. □

27.2. От покрытия к рademахеровской сложности через сцепление

Следующая лемма дает верхнюю оценку рademахеровской сложности A в терминах чисел покрытия $N(r, A)$. Эта техника называется *сцеплением* (chaining) и приписывается Дадли (Dudley).

Лемма 27.4. *Пусть $c = \min_{\bar{\mathbf{a}}} \max_{\mathbf{a} \in A} \|\mathbf{a} - \bar{\mathbf{a}}\|$. Тогда для любого целого $M > 0$*

$$R(A) \le \frac{c2^{-M}}{\sqrt{m}} + \frac{6c}{m} \sum_{k=1}^{M} 2^{-k} \sqrt{\log(N(c2^{-k}, A))}.$$

Доказательство. Пусть $\bar{\mathbf{a}}$ доставляет минимум целевой функции в определении c. С помощью леммы 26.6 мы можем проанализировать рademахеровскую сложность в предположении, что $\bar{\mathbf{a}} = \mathbf{0}$.

Рассмотрим множество $B_0 = \{\mathbf{0}\}$ и заметим, что оно является c-покрытием A. Пусть $B_1, \ldots, B_M$ – такие множества, что каждое B_k соответствует минимальному $(c2^{-k})$-покрытию A. Обозначим $\mathbf{a}^* = \operatorname{argmax}_{\mathbf{a} \in A} \langle \sigma, \mathbf{a} \rangle$ (если точек максимума больше одной, выберем произвольную, а если их не существует вовсе, выбираем $\mathbf{a}^*$ так, чтобы $\langle \sigma, \mathbf{a}^* \rangle$ было достаточно близко к верхней грани). Отметим, что $\mathbf{a}^*$ – функция от σ. Для любого k обозначим $\mathbf{b}^{(k)}$ ближайшего соседа $\mathbf{a}^*$, принадлежащего B_k (следовательно, $\mathbf{b}^{(k)}$ – тоже функция σ). Согласно неравенству треугольника,

$$\|\mathbf{b}^{(k)} - \mathbf{b}^{(k-1)}\| \le \|\mathbf{b}^{(k)} - \mathbf{a}^*\| + \|\mathbf{a}^* - \mathbf{b}^{(k-1)}\| \le c(2^{-k} + 2^{-(k-1)}) = 3c2^{-k}.$$

Для любого k определим множество

$$\hat{B}_k = \{(\mathbf{a} - \mathbf{a}') : \mathbf{a} \in B_k, \mathbf{a}' \in B^{k-1}, \|\mathbf{a} - \mathbf{a}'\| \le 3c2^{-k}\}.$$

Теперь можно написать

$$\begin{aligned} R(A) &= \frac{1}{m}\mathbb{E}\langle \boldsymbol{\sigma}, \mathbf{a}^*\rangle \\ &= \frac{1}{m}\mathbb{E}\left[\langle \boldsymbol{\sigma}, \mathbf{a}^* - \mathbf{b}^{(M)}\rangle + \sum_{k=1}^{M}\langle \boldsymbol{\sigma}, \mathbf{b}^{(k)} - \mathbf{b}^{(k-1)}\rangle\right] \\ &\le \frac{1}{m}\mathbb{E}[\|\sigma\|\,\|\mathbf{a}^* - \mathbf{b}^{(M)}\|] + \sum_{k=1}^{M}\frac{1}{m}\mathbb{E}\left[\sup_{\mathbf{a}\in \hat{B}_k}\langle \boldsymbol{\sigma}, \mathbf{a}\rangle\right]. \end{aligned}$$

Поскольку $\|\sigma\| = \sqrt{m}$ и $\|\mathbf{a}^* - \mathbf{b}^{(M)}\| \le c2^{-M}$, то первое слагаемое не превосходит $(c/\sqrt{m})2^{-M}$. Кроме того, по лемме Массарта

$$\frac{1}{m}\mathbb{E}\sup_{\mathbf{a}\in \hat{B}_k}\langle \boldsymbol{\sigma}, \mathbf{a}\rangle \le 3c2^{-k}\frac{\sqrt{2\log(N(c2^{-k}, A)^2)}}{m} = 6c2^{-k}\frac{\sqrt{\log(N(c2^{-k}, A))}}{m}.$$

Следовательно,

$$R(A) \le \frac{c2^{-M}}{\sqrt{m}} + \frac{6c}{m}\sum_{k=1}^{M}2^{-k}\sqrt{\log(N(c2^{-k}, A))}. \qquad \square$$

Как следствие, получаем такую лемму.

Лемма 27.5. *Предположим, что существуют $\alpha, \beta > 0$ такие, что для любого $k \ge 1$ имеет место неравенство*

$$\sqrt{\log(N(c2^{-k}, A))} \le \alpha + \beta k.$$

Тогда

$$R(A) \le \frac{6c}{m}(\alpha + 2\beta).$$

Доказательство. Эта граница вытекает из леммы 27.4, если положить $M \to \infty$ и заметить, что $\sum_{k=1}^{\infty}2^{-k} = 1$ и $\sum_{k=1}^{\infty}k2^{-k} = 2$. $\square$

Пример 27.2. Рассмотрим множество A, расположенное в d-мерном подпространстве $\mathbb{R}^m$ и такое, что $c = \max_{\mathbf{a}\in A}\|\mathbf{a}\|$. Мы показали, что $N(r, A) \le (2c\sqrt{d}/r)^d$. Поэтому для любого k имеем

$$\begin{aligned} \sqrt{\log(N(c2^{-k}, A))} &\le \sqrt{d\log(2^{k+1}\sqrt{d})} \\ &\le \sqrt{d\log(2\sqrt{d})} + \sqrt{kd} \\ &\le \sqrt{d\log(2\sqrt{d})} + \sqrt{d}k. \end{aligned}$$

Теперь применение леммы 27.5 дает

$$R(A) \le \frac{6c}{m}\left(\sqrt{d\log(2\sqrt{d})} + 2\sqrt{d}\right) = O\left(\frac{c\sqrt{d\log(d)}}{m}\right).$$

27.3. Библиографические сведения

Техника сцепления предложена в работе Dudley (1987). За подробными сведениями о числах покрытия и других мерах сложности, которые можно использовать для ограничения скорости равномерной сходимости, отсылаем читателя к работе Anthony & Bartlet, 1999.

ЛАНЬ®

ЛАНЬ®

Глава 28

ДОКАЗАТЕЛЬСТВО ФУНДАМЕНТАЛЬНОЙ ТЕОРЕМЫ ТЕОРИИ ОБУЧЕНИЯ

ЛАНЬ®

В этой главе мы докажем теорему 6.8 из главы 6. Напомним условия этой теоремы, которые будут подразумеваться на всем протяжении этой главы: класс гипотез $\mathcal{H}$ состоит из функций, отображающих множество образцов $\mathcal{X}$ в {0, 1}, используется бинарная функция потерь и $\text{VCdim}(\mathcal{H}) = d < \infty$.

Мы выведем верхнюю границу для реализуемого и агностического случая, а также нижнюю границу для агностического случая. Вывод нижней границы для реализуемого случая оставляем читателю в качестве упражнения.

28.1. Верхняя граница для агностического случая

ЛАНЬ®

Для вывода верхней границы мы должны доказать, что существует такое C, что $\mathcal{H}$ допускает агностическое PAC-обучение с выборочной сложностью

$$m_{\mathcal{H}}(\epsilon, \delta) \leq C \frac{d + \ln(1/\delta)}{\epsilon^2}.$$

Мы докажем чуть более слабую оценку:

$$m_{\mathcal{H}}(\epsilon, \delta) \leq C \frac{d \log(d/\epsilon) + \ln(1/\delta)}{\epsilon^2}. \tag{28.1}$$

Доказательство более точной границы требует больших усилий, включающих более тщательный анализ радемахеровской сложности с помощью техники «сцепления». Это выходит за рамки книги.

Для доказательства неравенства (28.1) достаточно показать, что применение правила ERM с выборочной сложностью

$$m \geq 4 \frac{32d}{\epsilon^2} \cdot \log\left(\frac{64d}{\epsilon^2}\right) + \frac{8}{\epsilon^2} \cdot (8d \log(\epsilon/d) + 2\log(4/\delta))$$

дает ϵ, δ-обучаемого для $\mathcal{H}$. Мы докажем этот результат, основываясь на теореме 26.5.

Пусть $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ – обучающий набор для классификации. Напомним, что по лемме Зауэра-Шеллаха, если $\mathrm{VCdim}(\mathcal{H}) = d$, то

$$|\{(h(\mathbf{x}_1), \ldots, h(\mathbf{x}_m)) : h \in \mathcal{H}\}| \leq \left(\frac{em}{d}\right)^d.$$

Обозначим $A = \{(\mathbb{1}_{[h(\mathbf{x}_1)\neq y_1]}, \ldots, \mathbb{1}_{[h(\mathbf{x}_m)\neq y_m]}) : h \in \mathcal{H}\}$. Очевидно, что

$$|A| \leq \left(\frac{em}{d}\right)^d.$$

ЛАНЬ®

Объединяя с леммой 26.8, получаем следующую границу радемахеровской сложности:

$$R(A) \leq \sqrt{\frac{2d\log(em/d)}{m}}.$$

Пользуясь теоремой 26.5, получаем, что с вероятностью не менее $1 - \delta$ для любого $h \in \mathcal{H}$ имеет место неравенство

$$L_{\mathcal{D}}(h) - L_S(h) \leq \sqrt{\frac{8d\log(em/d)}{m}} + \sqrt{\frac{2\log(2/\delta)}{m}}.$$

Повторяя это рассуждение для бинарной потери со знаком минус и применяя лемму о границе объединения, получаем, что с вероятностью не менее $1 - \delta$ для любого $h \in \mathcal{H}$ справедлива оценка

$$\begin{aligned} |L_{\mathcal{D}}(h) - L_S(h)| &\leq \sqrt{\frac{8d\log(em/d)}{m}} + \sqrt{\frac{2\log(4/\delta)}{m}} \\ &\leq 2\sqrt{\frac{8d\log(em/d) + 2\log(4/\delta)}{m}}. \end{aligned}$$

Чтобы эта величина была меньше ϵ, необходимо, чтобы

$$m \geq \frac{4}{\epsilon^2} \cdot (8d\log(m) + 8d\log(e/d) + 2\log(4/\delta)).$$

По лемме A.2, достаточным условием этого неравенства является

$$m \geq 4\frac{32d}{\epsilon^2} \cdot \log\left(\frac{64d}{\epsilon^2}\right) + \frac{8}{\epsilon^2} \cdot (8d\log(e/d) + 2\log(4/\delta)).$$

28.2. Нижняя граница для агностического случая

В этом разделе мы докажем, что существует такое C, что класс $\mathcal{H}$ допускает агностическое PAC-обучениес выборочной сложностью

$$m_{\mathcal{H}}(\epsilon, \delta) \leq C\frac{d + \ln(1/\delta)}{\epsilon^2}.$$

Доказательство будет состоять из двух частей. Сначала мы покажем, что $m(\epsilon, \delta) \geq 0{,}5\log(1/(4\delta))/\epsilon^2$, а затем – что для любого $\delta \leq 1/8$ справедливо неравенство $m(\epsilon, \delta) \geq 8d/\epsilon^2$. В совокупности эти два утверждения доказывают теорему.

ЛАНЬ®

28.2.1. Доказательство того, что $m(\epsilon, \delta) \geq 0{,}5\log(1/(4\delta))/\epsilon^2$

Сначала покажем, что для любого $\epsilon < 1/\sqrt{2}$ и любого $\delta \in (0, 1)$ имеет место неравенство $m(\epsilon, \delta) \geq 0{,}5\log(1/(4\delta))/\epsilon^2$. Для этого покажем, что для $m \leq 0{,}5\log(1/(4\delta))/\epsilon^2$ класс $\mathcal{H}$ не допускает обучения.

Выберем один пример, который разбивается классом $\mathcal{H}$. То есть пусть c – такой пример, что существуют гипотезы $h_+, h_- \in \mathcal{H}$, для которых $h_+(c) = 1$ и $h_-(c) = -1$. Определим два распределения $\mathcal{D}_+$ и $\mathcal{D}_-$ такие, что для $b \in \{\pm 1\}$ имеем

$$\mathcal{D}_b(\{(x,y)\}) = \begin{cases} \dfrac{1+yb\epsilon}{2}, & \text{если } x = c \\ 0, & \text{в противном случае} \end{cases}.$$

То есть вся масса распределения сконцентрирована в двух примерах: $(c, 1)$ и $(c, -1)$, так что вероятность (c, b) равна $(1 + b\epsilon)/2$, а вероятность$(c, -b)$ равна $(1 - b\epsilon)/2$.

Пусть A – произвольный алгоритм. Любой обучающий набор, выбранный из $\mathcal{D}_b$, имеет вид $S = (c, y_1), \ldots, (c, y_m)$. Следовательно, он полностью характеризуется вектором $\mathbf{y} = (y_1, \ldots, y_m) \in \{\pm 1\}^m$. Получив обучающий набор S, алгоритм A возвращает гипотезу $h : \mathcal{X} \to \{\pm 1\}$. Поскольку ошибка A относительно $\mathcal{D}_b$ зависит только от $h(c)$, мы можем рассматривать A как отображение $\{\pm 1\}^m$ в $\{\pm 1\}$. Поэтому обозначим $A(\mathbf{y})$ значение из $\{\pm 1\}$, соответствующее предсказанию $h(c)$, где h – гипотеза, которую A выводит, получив на входе обучающий набор $S = (c, y_1), \ldots, (c, y_m)$.

Заметим, что для любой гипотезы h

$$L_{\mathcal{D}_b}(h) = \frac{1 - h(c)b\epsilon}{2}.$$

В частности, оптимальная байесовская гипотеза равна h_b и

$$L_{\mathcal{D}_b}(A(\mathbf{y})) - L_{\mathcal{D}_b}(h_b) = \frac{1 - A(\mathbf{y})b\epsilon}{2} - \frac{1-\epsilon}{2} = \begin{cases} \epsilon, & \text{если } A(\mathbf{y}) \neq b \\ 0, & \text{в противном случае} \end{cases}.$$

Зафиксируем A. Для $b \in \{\pm 1\}$ обозначим $Y_b = \{\mathbf{y} \in \{0, 1\}^m : A(\mathbf{y}) \neq b\}$. Распределение $\mathcal{D}_b$ индуцирует вероятность P_b на $\{\pm 1\}^m$. Отсюда

$$\mathbb{P}[L_{\mathcal{D}_b}(A(\mathbf{y})) - L_{\mathcal{D}_b}(h_b) = \epsilon] = \mathcal{D}_b(Y^b) = \sum_{\mathbf{y}} P_b[\mathbf{y}]\mathbb{1}_{[A(\mathbf{y}) \neq b]}.$$

Обозначим $N^+ = \{\mathbf{y}: |\{i: y_i = 1\}| \geq m/2\}$ и $N^- = \{\pm 1\}^m \backslash N^+$. Заметим, что для любого $\mathbf{y} \in N^+$ имеем $P_+[\mathbf{y}] \geq P_-[\mathbf{y}]$ и для любого $\mathbf{y} \in N^-$ имеем $P_-[\mathbf{y}] \geq P_+[\mathbf{y}]$. Следовательно,

$$\begin{aligned}
&\max_{b\in\{\pm 1\}} \mathbb{P}[L_{\mathcal{D}_b}(A(\mathbf{y})) - L_{\mathcal{D}_b}(h_b) = \epsilon] \\
&\quad = \max_{b\in\{\pm 1\}} \sum_{\mathbf{y}} P_b[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq b]} \\
&\quad \geq \frac{1}{2}\sum_{\mathbf{y}} P_+[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq +]} + \frac{1}{2}\sum_{\mathbf{y}} P_-[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq -]} \\
&\quad = \frac{1}{2}\sum_{\mathbf{y}\in N^+} (P_+[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq +]} + P_-[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq -]}) \\
&\qquad + \frac{1}{2}\sum_{\mathbf{y}\in N^-} (P_+[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq +]} + P_-[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq -]}) \\
&\quad \geq \frac{1}{2}\sum_{\mathbf{y}\in N^+} (P_-[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq +]} + P_-[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq -]}) \\
&\qquad + \frac{1}{2}\sum_{\mathbf{y}\in N^-} (P_+[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq +]} + P_+[\mathbf{y}] \mathbb{1}_{[A(\mathbf{y})\neq -]}) \\
&\quad = \frac{1}{2}\sum_{\mathbf{y}\in N^+} P_-[\mathbf{y}] + \frac{1}{2}\sum_{\mathbf{y}\in N^-} P_+[\mathbf{y}].
\end{aligned}$$

Заметим далее, что $\sum_{\mathbf{y}\in N^+} P_-[\mathbf{y}] = \sum_{\mathbf{y}\in N^-} P_+[\mathbf{y}]$ и что оба значения равны вероятности того, что биномиальная случайная величина с параметрами $(m, (1-\epsilon)/2)$ будет иметь значение, превышающее $m/2$. По лемме B.11, эта вероятность ограничена снизу величиной

$$\frac{1}{2}\left(1-\sqrt{1-\exp(-m\epsilon^2/(1-\epsilon^2))}\right) \geq \frac{1}{2}\left(1-\sqrt{1-\exp(-2m\epsilon^2)}\right),$$

где мы предполагаем, что $\epsilon^2 \leq 1/2$. Отсюда следует, что если $m \leq 0{,}5\log(1/(4\delta))/\epsilon^2$, то существует такое b, что

$$\begin{aligned}
&\mathbb{P}[L_{\mathcal{D}_b}(A(\mathbf{y})) - L_{\mathcal{D}_b}(h_b) = \epsilon] \\
&\qquad \geq \frac{1}{2}\left(1-\sqrt{1-\sqrt{4\delta}}\right) \geq \delta,
\end{aligned}$$

где последнее неравенство получается стандартными алгебраическими преобразованиями. На этом доказательство завершается.

28.2.2. Доказательство того, что $m(\epsilon, 1/8) \geq 8d/\epsilon^2$

Теперь мы покажем, что для любого $\epsilon < 1/(8\sqrt{2})$ имеет место неравенство $m(\epsilon, \delta) \geq 8d/\epsilon^2$.

Положим $\rho = 8\epsilon$ и заметим, что $\rho \in (0, 1/\sqrt{2})$. Мы построим семейство распределений следующим образом. Во-первых, пусть $C = \{c_1, \ldots, c_d\}$ – множество d образцов, разбиваемое классом $\mathcal{H}$. Во-вторых, для каждого вектора $(b_1, \ldots, b_d) \in \{\pm 1\}^d$ определим распределение $\mathcal{D}_b$ такое, что

$$\mathcal{D}_b(\{(x,y)\}) = \begin{cases} \frac{1}{d}\cdot\frac{1+yb\epsilon}{2}, & \text{если } \exists i: x = c_i \\ 0, & \text{в противном случае} \end{cases}.$$

Иначе говоря, чтобы выбрать пример из $\mathcal{D}_b$, мы сначала случайно и равномерно выбираем элемент $c_i \in C$, а затем присваиваем метке значение b_i с вероятностью $(1 + \rho)/2$ или $-b_i$ с вероятностью $(1 - \rho)/2$.

Легко проверить, что оптимальный байесовский предиктор для $\mathcal{D}_b$ – это гипотеза $h \in \mathcal{H}$ такая, что $h(c_i) = b_i$ для всех $i \in [d]$, и ее ошибка оставляет $(1 - \rho)/2$. Кроме того, для любой функции $f : \mathcal{X} \to \{\pm 1\}$легко проверить, что

$$L_{\mathcal{D}_b}(f) = \frac{1+\rho}{2}\cdot\frac{|\{i \in \{d\} : f(c_i) \neq b_i\}|}{d} + \frac{1-\rho}{2}\cdot\frac{|\{i \in \{d\} : f(c_i) = b_i\}|}{d}.$$

Поэтому

$$L_{\mathcal{D}_b}(f) - \min_{h\in\mathcal{H}} L_{\mathcal{D}_b}(h) = \rho\cdot\frac{|\{i \in \{d\} : f(c_i) \neq b_i\}|}{d}. \tag{28.2}$$

Далее зафиксируем некоторый алгоритм обучения A. Как и в доказательстве теоремы об отсутствии бесплатных завтраков, имеем

$$\max_{\mathcal{D}_b : b\in\{\pm 1\}^d} \mathop{\mathbb{E}}_{S\sim\mathcal{D}_b^m}\left[L_{\mathcal{D}_b}(A(S)) - \min_{h\in\mathcal{H}} L_{\mathcal{D}_b}(h)\right] \tag{28.3}$$

$$\geq \mathop{\mathbb{E}}_{\mathcal{D}_b : b\sim U(\{\pm 1\}^d)} \mathop{\mathbb{E}}_{S\sim\mathcal{D}_b^m}\left[L_{\mathcal{D}_b}(A(S)) - \min_{h\in\mathcal{H}} L_{\mathcal{D}_b}(h)\right] \tag{28.4}$$

$$= \mathop{\mathbb{E}}_{\mathcal{D}_b : b\sim U(\{\pm 1\}^d)} \mathop{\mathbb{E}}_{S\sim\mathcal{D}_b^m}\left[\rho\cdot\frac{|\{i \in [d] : A(S)(c_i) \neq b_i|}{d}\right] \tag{28.5}$$

$$= \frac{\rho}{d}\sum_{i=1}^{d} \mathop{\mathbb{E}}_{\mathcal{D}_b : b\sim U(\{\pm 1\}^d)} \mathop{\mathbb{E}}_{S\sim\mathcal{D}_b^m} \mathbb{1}_{[A(S)(c_i)\neq b_i]}, \tag{28.6}$$

где последнее равенство следует из (28.2). Кроме того, по определению $\mathcal{D}_b$, чтобы выбрать $S \sim \mathcal{D}_b$, мы сначала выбираем $(j_1, \ldots, j_m) \sim U([d])^m$, полагаем $x_r = c_{j_i}$ и наконец выбираем y_r такое, что $P[y_r = b_{j_i}] = (1 + \rho)/2$. Упростим нотацию и будем использовать обозначение $y \sim b$ для выборки из распределения $P[y = b] = (1 + \rho)/2$. Тогда правая часть равенства (28.6) равна

$$\frac{\rho}{d}\sum_{i=1}^{d} \mathop{\mathbb{E}}_{j\sim U([d])^m} \mathop{\mathbb{E}}_{b\sim U(\{\pm 1\})^d} \mathop{\mathbb{E}}_{\forall r, y_r\sim b_{j_r}} \mathbb{1}_{[A(S)(c_i)\neq b_i]}. \tag{28.7}$$

Далее мы проведем рассуждение, состоящее из двух шагов. Сначала покажем, что из всех алгоритмов обучения A выражение (28.7) (а значит, и (28.4)) минимизирует алгоритм максимального правдоподобия, который будем обозначать A_{ML}. Формально, для любого $iA_{ML}(S)(c_i)$ определяется большинство голосов за метки из множества $\{y_r : r \in [m], x_r = c_i\}$. Затем мы ограничим снизу выражение (28.7) для A_{ML}.

Лемма 28.1. *Из всех алгоритмов минимум выражению (28.4) доставляет алгоритм максимального правдоподобия A_{ML}, определяемый следующим образом:*

$$\forall i, \quad A_{ML}(S)(c_i) = sign\left(\sum_{r:x_r=c_i} y_r\right).$$

Доказательство. Зафиксируем некоторое $j \in [d]^m$. Заметим, что при заданных j и $y \in \{\pm 1\}^m$ обучающий набор S полностью определен. Поэтому мы можем писать $A(j, y)$ вместо $A(S)$. Зафиксируем также $i \in [d]$. Обозначим $b^{\neg i}$ последовательность $(b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_m)$. Кроме того, для любого $y \in \{\pm 1\}^m$ обозначим y^I элементы y, соответствующие индексам, для которых $j_r = i$, $y^{\neg I}$ – остальные элементы y. Имеем

$$\begin{aligned}
&\mathop{\mathbb{E}}_{b\sim U(\{\pm 1\}^d)} \mathop{\mathbb{E}}_{\forall r, y_r \sim b_{j_r}} \mathbb{1}_{[A(S)(c_i)\neq b_i]} \\
&= \frac{1}{2} \sum_{b_i \in \{\pm 1\}} \mathop{\mathbb{E}}_{b^{\neg i}\sim U(\{\pm 1\}^{d-1})} \sum_{y} P[y \mid b^{\neg i}, b_i] \mathbb{1}_{[A(j,y)(c_i)\neq b_i]} \\
&= \mathop{\mathbb{E}}_{b^{\neg i}\sim U(\{\pm 1\}^{d-1})} \sum_{y^{\neg I}} P[y^{\neg I} \mid b^{\neg i}] \frac{1}{2} \sum_{y^I} \left(\sum_{b_i \in \{\pm 1\}} P[y^I \mid b_i] \mathbb{1}_{[A(j,y)(c_i)\neq b_i]} \right).
\end{aligned}$$

Сумма в скобках достигает минимума, когда $A(j, y)(c_i)$ доставляет максимум $P[y^I|b_i]$ на $b_i \in \{\pm 1\}$, а это и есть правило максимального правдоподобия. Для завершения доказательства нужно просто повторить это рассуждение для всех i. □

Зафиксируем i. Для любого j обозначим $n_i(j) = \{|t : j_t = i|\}$ количество образцов, в которых образец равен c_i. По правилу максимального правдоподобия величина

$$\mathop{\mathbb{E}}_{b\sim U(\{\pm 1\}^d)} \mathop{\mathbb{E}}_{\forall r, y_r \sim b_{j_r}} \mathbb{1}_{[A_{ML}(S)(c_i)\neq b_i]},$$

в точности равна вероятности того, что биномиальная случайная величина с параметрами $(n_i(j), (1 - \rho)/2)$ окажется больше $n_i(j)/2$. По лемме B.11 и в силу предположения, что $\rho^2 \le 1/2$, имеем

$$P[B \ge n_i(j)/2] \ge \frac{1}{2}\left(1 - \sqrt{1 - e^{-2n_i(j)\rho^2}}\right).$$

Таким образом, мы показали, что

$$\begin{aligned}
&\frac{\rho}{d} \sum_{i=1}^{d} \mathop{\mathbb{E}}_{j\sim U([d])^m} \mathop{\mathbb{E}}_{b\sim U(\{\pm 1\}^d)} \mathop{\mathbb{E}}_{\forall r, y_r \sim b_{j_r}} \mathbb{1}_{[A(S)(c_i)\neq b_i]} \\
&\quad \ge \frac{\rho}{2d} \sum_{i=1}^{d} \mathop{\mathbb{E}}_{j\sim U([d])^m} \left(1 - \sqrt{1 - e^{-2\rho^2 n_i(j)}}\right) \\
&\quad \ge \frac{\rho}{2d} \sum_{i=1}^{d} \mathop{\mathbb{E}}_{j\sim U([d])^m} \left(1 - \sqrt{2\rho^2 n_i(j)}\right),
\end{aligned}$$

где в последнем неравенстве использовано неравенство $1 - e^{-a} \le a$.

Поскольку квадратный корень – выпуклая функция, мы можем применить неравенство Йенсена и получить, что это выражение ограничено снизу величиной

$$\geq \frac{\rho}{2d}\sum_{i=1}^{d}\left(1-\sqrt{2\rho^2 \mathop{\mathbb{E}}_{j\sim U([d])^m} n_i(j)}\right)$$
$$= \frac{\rho}{2d}\sum_{i=1}^{d}\left(1-\sqrt{2\rho^2 m/d}\right)$$
$$\geq \frac{\rho}{d}\left(1-\sqrt{2\rho^2 m/d}\right).$$

При условии, что $m < d/(8\rho^2)$, этот член будет больше $\rho/4$.

В итоге мы показали, что если $m < d/(8\rho^2)$, то для любого алгоритма существует такое распределение, что

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[L_{\mathcal{D}}(A(S)) - \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h)\right] \geq \rho/4.$$

Наконец, положим $\Delta = (1/\rho)(L_{\mathcal{D}}(A(S)) - \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h))$ и заметим, что $\Delta \in [0, 1]$ (см. равенство (28.5)). Поэтому в силу леммы B.1 имеем

$$\mathbb{P}\left[L_{\mathcal{D}}(A(S)) - \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) > \epsilon\right] = \mathbb{P}\left[\Delta > \frac{\epsilon}{\rho}\right] \geq \mathbb{E}[\Delta] - \frac{\epsilon}{\rho}$$
$$\geq \frac{1}{4} - \frac{\epsilon}{\rho}.$$

Выбрав $\rho = 8$, заключаем, что если $m < 8d/\epsilon^2$, то с вероятностью не менее 1/8 будем иметь $L_{\mathcal{D}}(A(S)) - \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) \geq \epsilon$.

28.3. Верхняя граница для реализуемого случая

В этом разделе мы докажем, что существует такое C, что класс $\mathcal{H}$ является PAC-обучаемым с выборочной сложностью

$$m_{\mathcal{H}}(\epsilon, \delta) \leq C\frac{d\ln(1/\epsilon) + \ln(1/\delta)}{\epsilon}.$$

Для этого мы покажем, что при $m \geq C\dfrac{d\ln(1/\epsilon) + \ln(1/\delta)}{\epsilon}$ $\mathcal{H}$ допускает обучение по правилу ERM. Мы докажем это утверждение, пользуясь понятием ϵ-сети.

Определение 28.2 (ϵ-сеть). Пусть $\mathcal{X}$ – множество образцов. Множество $S \subset \mathcal{X}$ называется ϵ-сетью для $\mathcal{H} \subset 2^{\mathcal{X}}$ относительно распределения $\mathcal{D}$ на $\mathcal{X}$, если

$$\forall h \in \mathcal{H}: \mathcal{D}(h) \geq \epsilon \Rightarrow h \cap S \neq \emptyset.$$

Теорема 28.3. *Пусть* $\mathcal{H} \subset 2^{\mathcal{X}}$ *и* $\mathrm{VCdim}(\mathcal{H}) = d$. *Зафиксируем* $\epsilon \in (0, 1)$, $\delta \in (0,1/4)$, *и пусть*

$$m \geq \frac{8}{\epsilon}\left(2d\log\left(\frac{16e}{\epsilon}\right) + \log\left(\frac{2}{\delta}\right)\right).$$

Тогда с вероятностью на менее $1-\delta$ *случайная выборка* $S \sim \mathcal{D}^m$ *является* ϵ*-сетью для* $\mathcal{H}$.

Доказательство. Обозначим

$$B = \{S \subset \mathcal{X} : |S| = m, \exists h \in \mathcal{H}, \mathcal{D}(h) \geq \epsilon, h \cap S = \emptyset\}$$

множество множеств, не являющихся ϵ-сетями. Требуется ограничить $\mathbb{P}[S \in B]$. Определим

$$B' = \{(S, T) \subset \mathcal{X} : |S| = |T| = m, \exists h \in \mathcal{H}, \mathcal{D}(h) \geq \epsilon, h \cap S = \emptyset, |T \cap h| > \epsilon m/2\}. \quad \square$$

ЛАНЬ®

Утверждение 1

$$\mathbb{P}[S \in B] \leq 2\mathbb{P}[(S, T) \in B'].$$

Доказательство утверждения 1. Поскольку S и T выбраны независимо, можно написать

$$\mathbb{P}[(S,T) \in B'] = \mathop{\mathbb{E}}_{(S,T)\sim\mathcal{D}^{2m}}[\mathbb{1}_{[(S,T)\in B']}] = \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\mathop{\mathbb{E}}_{T\sim\mathcal{D}^m}[\mathbb{1}_{[(S,T)\in B']}]\right].$$

Заметим, что из того, что $(S, T) \in B'$, следует, что $S \in B$, и, значит, $\mathbb{1}_{[(S,T)\in B']} = \mathbb{1}_{[(S,T)\in B']}\mathbb{1}_{[S\in B]}$, откуда

$$\begin{aligned}\mathbb{P}[(S,T) \in B'] &= \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\mathop{\mathbb{E}}_{T\sim\mathcal{D}^m}\mathbb{1}_{[(S,T)\in B']}\mathbb{1}_{[S\in B]} \\ &= \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\mathbb{1}_{[S\in B]}\mathop{\mathbb{E}}_{T\sim\mathcal{D}^m}\mathbb{1}_{[(S,T)\in B']}.\end{aligned}$$

Зафиксируем некоторое S. Тогда либо $\mathbb{1}_{[S\in B]} = 0$, либо $S \in B$, и тогда существует h_S такое, что $\mathcal{D}(h_S) \geq \epsilon$ и $|h_S \cap S| = 0$. Отсюда следует, что достаточным условием для $(S, T) \in B'$, является $|T \cap h_S| > \epsilon m/2$. Стало быть, если $S \in B$, то имеем

$$\mathop{\mathbb{E}}_{T\sim\mathcal{D}^m}\mathbb{1}_{[(S,T)\in B']} \geq \mathop{\mathbb{P}}_{T\sim\mathcal{D}^m}\left[|T \cap h_S| > \frac{\epsilon m}{2}\right].$$

Но поскольку мы теперь предполагаем, что $S \in B$, то знаем, что $\mathcal{D}(h_S) = \rho \geq \epsilon$. Поэтому $|T \cap h_S|$ – биномиальная случайная величина с параметрами ρ (вероятность успеха в одном испытании) и m (количество испытаний). Согласно неравенству Чернова,

$$\mathbb{P}\left[|T \cap h_S| \leq \frac{\epsilon m}{2}\right] \leq e^{-\frac{2}{m\rho}(m\rho - m\rho/2)^2} = e^{-m\rho/2} \leq e^{-m\epsilon/2} \leq e^{-d\log(1/\delta)/2} = \delta^{d/2} \leq 1/2.$$

Таким образом,

$$\mathbb{P}\left[|T \cap h_S| > \frac{\epsilon m}{2}\right] = 1 - \mathbb{P}\left[|T \cap h_S| \leq \frac{\epsilon m}{2}\right] \geq 1 - \mathbb{P}\left[|T \cap h_S| \leq \frac{\rho m}{2}\right] \geq 1/2.$$

Объединяя все вышесказанное, *мы завершаем доказательство утверждения 1*. $\square$

Утверждение 2 (симметризация):

$$\mathbb{P}[(S, T) \in B'] \le e^{-\epsilon m/4} \tau_{\mathcal{H}}(2m).$$

Доказательство утверждения 2. Для упрощения обозначений положим $\alpha = m/2$ и для последовательности $A = (x_1, \ldots, x_{2m})$ обозначим $A_0 = (x_1, \ldots, x_m)$. Согласно определению B', имеем

$$\begin{aligned}\mathbb{P}[A \in B'] &= \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} \max_{h \in \mathcal{H}} \mathbb{1}_{[\mathcal{D}(h) \ge \epsilon]} \mathbb{1}_{[|h \cap A_0 = 0]} \mathbb{1}_{[|h \cap A \ge \alpha]} \\ &\le \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} \max_{h \in \mathcal{H}} \mathbb{1}_{[|h \cap A_0 = 0]} \mathbb{1}_{[|h \cap A \ge \alpha]}.\end{aligned}$$

Теперь обозначим $\mathcal{H}_A$ эффективное число различных гипотез на A, т. е. $\mathcal{H}_A = \{h \cap A : h \in \mathcal{H}\}$. Тогда

$$\begin{aligned}\mathbb{P}[A \in B'] &= \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} \max_{h \in \mathcal{H}_A} \mathbb{1}_{[|h \cap A_0 = 0]} \mathbb{1}_{[|h \cap A \ge \alpha]} \\ &\le \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} \sum_{h \in \mathcal{H}_A} \mathbb{1}_{[|h \cap A_0 = 0]} \mathbb{1}_{[|h \cap A \ge \alpha]}.\end{aligned}$$

Пусть $J = \{\mathbf{j} \subset [2m] : |\mathbf{j}| = m\}$. Для любого $\mathbf{j} \in J$ и $A = (x_1, \ldots, x_{2m})$ определим $A_\mathbf{j} = (x_{j_1}, \ldots, x_{j_m})$. Поскольку элементы A независимы и одинаково распределены, то для любого $\mathbf{j} \in J$ и любой функции $f(A, A_0)$ имеет место $\mathbb{E}_{A \sim \mathcal{D}^{2m}}[f(A, A_0)] = \mathbb{E}_{A \sim \mathcal{D}^{2m}}[f(A, A_\mathbf{j})]$. Так как это справедливо для любого $\mathbf{j}$, то справедливо и для математического ожидания $\mathbf{j}$, случайно выбранного из J. В частности, это справедливо для функции $f(A, A_0) = \sum_{h \in \mathcal{H}_A} \mathbb{1}_{[|h \cap A_0| = 0]} \mathbb{1}_{[|h \cap A| = 0 \ge \alpha]}$. Таким образом, мы получаем

$$\begin{aligned}\mathbb{P}[A \in B'] &= \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} \mathop{\mathbb{E}}_{j \sim J} \sum_{h \in \mathcal{H}_A} \mathbb{1}_{[|h \cap A_\mathbf{j} = 0]} \mathbb{1}_{[|h \cap A \ge \alpha]} \\ &= \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} \sum_{h \in \mathcal{H}_A} \mathbb{1}_{[|h \cap A \ge \alpha]} \mathop{\mathbb{E}}_{j \sim J} \mathbb{1}_{[|h \cap A_\mathbf{j} = 0]}.\end{aligned}$$

Теперь зафиксируем такое A, что $|h \cap A| \ge \alpha$. Тогда $\mathbb{E}_j \mathbb{1}_{[|h \cap A_\mathbf{j}| = 0]}$ – это вероятность того, что при выборе m шаров из ящика, содержащего по меньшей мере α красных шаров, мы не достанем ни одного красного шара. Эта вероятность не превосходит

$$(1 - \alpha/(2m))^m = (1 - \epsilon/4)^m \le e^{-m/4}.$$

Отсюда получаем, что

$$\mathbb{P}[A \in B'] = \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} \sum_{h \in \mathcal{H}_A} e^{-\epsilon m/4} \le e^{-\epsilon m/4} \mathop{\mathbb{E}}_{A \sim \mathcal{D}^{2m}} |\mathcal{H}_A|.$$

Теперь *доказательство утверждения 2* следует из определения функции роста.

Завершение доказательства. По лемме Зауэра $\tau_{\mathcal{H}}(2m) \le (2em/d)^d$. Объединяя это с двумя доказанными выше утверждениями, получаем, что

$$\mathbb{P}[S \in B] \le 2(2em/d)^d e^{-\epsilon m/4}.$$

Мы хотим, чтобы правая часть этого неравенства была не больше δ, т. е.

$$2(2em/d)\mathrm{d}e^{-\epsilon m/4} \le \delta.$$

После перегруппировки членов получаем такое требование:

$$m \geq \frac{4}{\epsilon}(d\log(2em/d)+\log(2/\delta)) = \frac{4d}{\epsilon}\log(m)+\frac{4}{\epsilon}(d\log(2e/d)+\log(2/\delta)).$$

По лемме A.2 достаточным условием для выполнения этого требования является

$$m \geq \frac{16d}{\epsilon}\log\left(\frac{8d}{\epsilon}\right)+\frac{8}{\epsilon}(d\log(2e/d)+\log(2/\delta)).$$

А достаточным условием этого неравенства является

$$\begin{aligned} m &\geq \frac{16d}{\epsilon}\log\left(\frac{8d}{\epsilon}\right)+\frac{16}{\epsilon}\left(d\log(2e/d)+\frac{1}{2}\log(2/\delta)\right) \\ &= \frac{16d}{\epsilon}\left(\log\left(\frac{8d2e}{d\epsilon}\right)\right)+\frac{8}{\epsilon}\log(2/\delta) \\ &= \frac{8}{\epsilon}\left(2d\log\left(\frac{16e}{\epsilon}\right)+\log\left(\frac{2}{\delta}\right)\right), \end{aligned}$$

и на этом доказательство завершается. □

28.3.1. От ϵ-сетей к PAC-обучаемости

Теорема 28.4. *Пусть $\mathcal{H}$ – класс гипотез на $\mathcal{X}$, для которого* VCdim($\mathcal{H}$) = d. *Пусть $\mathcal{D}$ – распределение на $\mathcal{X}$, и $c \in \mathcal{H}$ – целевая гипотеза. Зафиксируем $\epsilon, \delta \in (0, 1)$, и пусть m определено, как в теореме 28.3. Тогда с вероятностью не менее $1 - \delta$ для m независимых и одинаково распределенных примеров из $\mathcal{X}$ с метками, назначенными гипотезой c, истинная ошибка любой ERM-гипотезы не превосходит ϵ.*

Доказательство. Определим класс $\mathcal{H}_c = \{c \Delta h: h \in \mathcal{H}\}$, где $c \Delta h = (h \backslash c) \cup (c \backslash h)$. Легко проверить, что если некоторое $A \subset \mathcal{X}$ разбивается классом $\mathcal{H}$, то оно разбивается также классом $\mathcal{H}_c$, и наоборот. Следовательно, VCdim($\mathcal{H}$) = VCdim($\mathcal{H}_c$). Но тогда по теореме 28.3 с вероятностью не менее $1 - \delta$ выборка S является ϵ-сетью для $\mathcal{H}_c$. Заметим, что $L_{\mathcal{D}}(h) = \mathcal{D}(h \Delta c)$. Поэтому для любой гипотезы $h \in \mathcal{H}$, для которой $L_{\mathcal{D}}(h) \geq \epsilon$, имеем $|(h \Delta c) \cap S| > 0$, откуда следует, что h не может быть ERM-гипотезой. Что и требовалось доказать. □

Глава 29

МНОГОКЛАССОВАЯ ОБУЧАЕМОСТЬ

В главе 17 мы познакомились с проблемой многоклассовой категоризации, цель которой – обучить предиктор $h : \mathcal{X} \to [k]$. В этой главе мы займемся PAC-обучаемостью многоклассовых предикторов относительно бинарной потери. Как и в главе 6, нашей основной целью будет:

- охарактеризовать классы многоклассовых гипотез, допускающие обучение в (многоклассовой) модели PAC;
- количественно оценить выборочную сложность таких классов гипотез.

В свете фундаментальной теоремы теории обучения (теорема 6.8) естественно поискать обобщение VC-размерности на классы многоклассовых гипотез. В разделе 29.1 мы представим такое обобщение, *размерность Натараджана*, и сформулируем обобщение фундаментальной теоремы на основе этой размерности. Затем мы покажем, как вычислить размерность Натараджана нескольких важных классов гипотез.

Напомним, что смысл фундаментальной теоремы теории обучения состоит в том, что класс гипотез, состоящий из бинарных классификаторов, допускает обучение (относительно бинарной потери) тогда и только тогда, когда он обладает свойством равномерной сходимости, и в этом случае для обучения годится любой ERM-обучаемый. В главе 13 (упражнение 13.2) мы показали, что эта эквивалентность нарушается для некоторых выпуклых проблем обучения. Последний раздел этой главы посвящен доказательству того, что эквивалентность между обучаемостью и равномерной сходимостью нарушается даже в многоклассовых проблемах с бинарной потерей, очень похожих на бинарную классификацию. Мы построим класс гипотез, допускающий обучение конкретным ERM-обучаемым, но такой, что другие ERM-обучаемые могут терпеть неудачу, и для которого свойство равномерной сходимости не имеет места.

29.1. Размерность Натараджана

В этом разделе мы определим размерность Натараджана – обобщение VC-размерности на многоклассовые предикторы. В этом разделе мы считаем, что $\mathcal{H}$ – класс гипотез, состоящий из многоклассовых предикторов, т. е. любая $h \in \mathcal{H}$ – функция из $\mathcal{X}$ в $[k]$.

Прежде чем определять размерность Натараджана, мы обобщим определение разбиения.

Определение 29.1 (разбиение – многоклассовый вариант). Говорят, что множество $C \subset \mathcal{X}$ разбивается классом $\mathcal{H}$, если существуют две функции $f_0, f_1 : C \to [k]$ такие, что:

- для всех $x \in C$ $f_0(x) \neq f_1(x)$;
- для любого $B \subset C$ существует функция $h \in \mathcal{H}$ такая, что

$$\forall x \in B h(x) = f_0(x) \text{ и } \forall x \in C \setminus B h(x) = f_1(x).$$

Определение 29.2 (размерность Натараджана). Размерностью Натараджана класса $\mathcal{H}$, обозначаемой $\mathrm{Ndim}(\mathcal{H})$, называется максимальный размер разбитого им множества $C \subset \mathcal{X}$.

Нетрудно видеть, что при наличии всего двух классов $\mathrm{Ndim}(\mathcal{H}) = \mathrm{VCdim}(\mathcal{H})$. Поэтому размерность Натараджана обобщает VC-размерность. Далее мы покажем, что размерность Натараджана позволяет обобщить фундаментальную теорему статистического обучения с бинарной на многоклассовую классификацию.

ЛАНЬ®

29.2. Фундаментальная многоклассовая теорема

Теорема 29.3 (фундаментальная многоклассовая теорема). *Существуют абсолютные константы C_1, $C_2 > 0$ такие, что выполняется следующее утверждение. Для любого класса гипотез $\mathcal{H}$, состоящего из функций, отображающих $\mathcal{X}$ в $[k]$, размерность Натараджана которого равна d, имеет место:*

1) *$\mathcal{H}$ обладает свойством равномерной сходимости с выборочной сложностью*

$$C_1 \frac{d + \log(1/\delta)}{\epsilon^2} \leq m^{\mathrm{UC}}_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d \log(k) + \log(1/\delta)}{\epsilon^2};$$

2) *$\mathcal{H}$ допускает агностическое PAC-обучение с выборочной сложностью*

$$C_1 \frac{d + \log(1/\delta)}{\epsilon^2} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d \log(k) + \log(1/\delta)}{\epsilon^2};$$

3) *$\mathcal{H}$ допускает PAC-обучение (в предположении реализуемости) с выборочной сложностью*

$$C_1 \frac{d + \log(1/\delta)}{\epsilon} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d \log\left(\frac{kd}{\epsilon}\right) + \log(1/\delta)}{\epsilon}.$$

29.2.1. О доказательстве теоремы 29.3

Нижние границы в теореме 29.3 можно вывести путем сведения к фундаментальной теореме бинарной классификации (см. упражнение 29.5).

Верхние границы можно доказать, рассуждая примерно так же, как при доказательстве фундаментальной теоремы бинарной классификации в главе 28 (см.

упражнение 29.4). Единственная часть доказательства, требующая модификации, – применение леммы Зауэра. Поскольку она относится только к бинарным классам, то должна быть заменена следующей леммой Натараджана.

Лемма 29.4 (Натараджана). $|\mathcal{H}| \le |\mathcal{X}|^{\mathrm{Ndim}(\mathcal{H})} \cdot k^{2\mathrm{Ndim}(\mathcal{H})}$.

Доказательство леммы Натараджана проводится в том же духе, что леммы Зауэра, и мы оставляем его читателю в качестве упражнения (см. упражнение 29.3).

29.3. Вычисление размерности Натараджана

В этом разделе мы покажем, как вычислить (или оценить) размерность Натараджана нескольких популярных классов, некоторые из которых изучались в главе 17. Как будет видно из этих вычислений, размерность Натараджана часто пропорциональна количеству параметров, требуемых для определения гипотезы.

29.3.1. Метод «один против всех»

В главе 17 мы рассматривали два способа свести многоклассовую категоризацию к бинарной классификации: «один против всех» и «все пары». В этом разделе мы вычислим размерность Натараджана в методе «один против всех».

Напомним, что этот метод подразумевает, что для каждой метки обучается бинарный классификатор, который отличает эту метку от всех остальных. Поэтому естественно возникает мысль рассмотреть классы многоклассовых гипотез следующего вида. Пусть $\mathcal{H}_{\mathrm{bin}} \subset \{0, 1\}^{\mathcal{X}}$ – класс бинарных гипотез. Для любой гипотезы $\bar{h} = (h_1, \ldots, h_k) \in (\mathcal{H}_{\mathrm{bin}})^k$ определим $T(\bar{h}) : \mathcal{X} \to [k]$

$$T(\bar{h})(x) = \operatorname*{argmax}_{i \in [k]} h_i(x).$$

Если существуют две метки, доставляющие максимум $h_i(x)$, то выбираем наименьшую. Кроме того, положим

$$\mathcal{H}_{\mathrm{bin}}^{\mathrm{OvA,k}} = \{T(\bar{h}) : \bar{h} \in (\mathcal{H}_{\mathrm{bin}})^k\}.$$

Какой «должна» быть размерность Натараджана класса $\mathcal{H}_{\mathrm{bin}}^{\mathrm{OvA,k}}$? Интуитивно понятно, что для определения гипотезы в $\mathcal{H}_{\mathrm{bin}}$ нам нужно $d = \mathrm{VCdim}(\mathcal{H}_{\mathrm{bin}})$ параметров. Для определения гипотезы в $\mathcal{H}_{\mathrm{bin}}^{\mathrm{OvA,k}}$ необходимо определить k гипотез $\mathcal{H}_{\mathrm{bin}}$. Поэтому должно хватить kd параметров. Следующая лемма подтверждает это интуитивную догадку.

Лемма 29.5. *Если* $d = \mathrm{VCdim}(\mathcal{H}_{\mathrm{bin}})$, *то*

$$\mathrm{Ndim}(\mathcal{H}_{\mathrm{bin}}^{\mathrm{OvA,k}}) \le 3kd \log(kd).$$

Доказательство. Пусть $C \subset \mathcal{X}$ – разбитое классом множество. По определению разбиения (для многоклассовых гипотез)

$$|(\mathcal{H}_{\mathrm{bin}}^{\mathrm{OvA,k}})_C| \ge 2^{|C|}.$$

С другой стороны, каждая гипотеза в $\mathcal{H}_{\text{bin}}^{\text{OvA,k}}$ определяется с помощью k гипотез из $\mathcal{H}_{\text{bin}}$. Поэтому

$$|(\mathcal{H}_{\text{bin}}^{\text{OvA,k}})_C| \le |(\mathcal{H}_{\text{bin}})_C|^{k|}$$

По лемме Зауэра, $|(\mathcal{H}_{\text{bin}})_C| \le |C|^d$. Отсюда

$$2^{|C|} \le |(\mathcal{H}_{\text{bin}}^{\text{OvA,k}})_C| \le |C|^{dk}.$$

Для завершения доказательства следует прологарифмировать и применить лемму A.1. □

Насколько точна граница из леммы 29.5? Нетрудно видеть, что для некоторых классов $\text{Ndim}(\mathcal{H}_{\text{bin}}^{\text{OvA,k}})$ может быть гораздо меньше dk (см. упражнение 29.1). Однако существует несколько естественных бинарных классов $\mathcal{H}_{\text{bin}}$ (например, полупространства), для которых $\text{Ndim}(\mathcal{H}_{\text{bin}}^{\text{OvA,k}}) = \Omega(dk)$ (см. упражнение 29.6).

29.3.2. Сведение многоклассовой категоризации к бинарной классификации в общем случае

То же рассуждение, с помощью которого мы доказали лемму 29.5, можно использовать для нахождения верхней границы размерности Натараджана в более общем случае сведения многих классов к двум. При любом таком сведении на данных обучается несколько бинарных классификаторов. Затем метка нового предъявленного образца предсказывается путем применения некоторых правил, принимающих в расчет метки, предсказанные бинарными классификаторами. Частными случаями такого сведения являются методы «один против всех» и «все пары».

Предположим, что таким методом мы обучили l бинарных классификаторов на основе класса $\mathcal{H}_{\text{bin}}$ и что $r : \{0, 1\}^l \to [k]$ – правило, которое определяет (многоклассовую) метку по предсказаниям бинарных классификаторов. Класс гипотез, соответствующий этому методу, можно определить следующим образом. Для каждой гипотезы $\bar{h} = (h_1, \ldots, h_l) \in (\mathcal{H}_{\text{bin}})^l$ определим $R(\bar{h}) : \mathcal{X} \to [k]$:

$$R(\bar{h})(x) = r(h_1(x), \ldots, h_l(x)).$$

Наконец, положим

$$\mathcal{H}_{\text{bin}}^r = \{R(\bar{h}) : \bar{h} \in (\mathcal{H}_{\text{bin}})^l\}.$$

По аналогии с леммой 29.5 можно доказать следующую лемму.

Лемма 29.6. *Если* $d = \text{VCdim}(\mathcal{H}_{\text{bin}})$, *то*

$$\text{Ndim}(\mathcal{H}_{\text{bin}}^r) \le 3ld\log(ld).$$

Доказательство оставляем читателю (см. упражнение 29.2).

29.3.3. Линейные многоклассовые предикторы

Далее рассмотрим класс линейных многоклассовых предикторов (см. раздел 17.2). Пусть $\Psi : \mathcal{X} \times [k] \to \mathbb{R}^d$ – зависящее от класса отображение в пространство признаков и обозначим

$$\mathcal{H}_\Psi = \left\{ x \mapsto \operatorname*{argmax}_{i\in[k]} \langle \mathbf{w}, \Psi(x,i)\rangle : \mathbf{w} \in \mathbb{R}^d \right\}. \tag{29.1}$$

Любая гипотеза из $\mathcal{H}_\Psi$ определяется d параметрами, т. е. вектором $\mathbf{w} \in \mathbb{R}^d$. Поэтому следует ожидать, что размерность Натараджана будет ограничена сверху числом d. И действительно справедлива

Теорема 29.7. $\mathrm{Ndim}(\mathcal{H}_\Psi) \le d$.

Доказательство. Пусть $C \subset \mathcal{X}$ – разбитое множество и $f_0, f_1 : C \to [k]$ – две функции, свидетельствующие о разбиении. Нам нужно показать, что $|C| \le d$. Для любого $x \in C$ обозначим $\rho(x) = \Psi(x, f_0(x)) - \Psi(x, f_1(x))$. Мы утверждаем, что множество $\rho(C) \stackrel{\text{def}}{=} \{\rho(x) : x \in C\}$ содержит $|C|$ элементов (т. е. ρ – взаимно однозначное отображение) и разбивается классом бинарных гипотез, состоящим из однородных линейных разделителей в $\mathbb{R}^d$:

$$\mathcal{H} = \{\mathbf{x} \mapsto \mathrm{sign}(\langle \mathbf{w}, \mathbf{x}\rangle) : \mathbf{w} \in \mathbb{R}^d\}.$$

Поскольку $\mathrm{VCdim}(\mathcal{H}) = d$, отсюда следует, что $|C| = |\rho(C)| \le d$, что нам и нужно.

Чтобы установить справедливость нашего утверждения, достаточно показать, что $|\mathcal{H}_{\rho(C)}| = 2^{|C|}$. В самом деле, если дано подмножество $B \subset C$, то по определению разбиения существует $h_B \in \mathcal{H}_\Psi$, для которой

$$\forall x \in B h_B(x) = f_0(x) \text{ и } \forall x \in C \setminus B h_B(x) = f_1(x).$$

Пусть $\mathbf{w}_B \in \mathbb{R}^d$ – вектор, определяющий h_B. Для любого $x \in B$ имеем

$$\langle \mathbf{w}, \Psi(x, f_0(x))\rangle > \langle \mathbf{w}, \Psi(x, f_1(x))\rangle \Rightarrow \langle \mathbf{w}, \rho(x)\rangle > 0.$$

Аналогично для любого $x \in C \setminus B$ имеем

$$\langle \mathbf{w}, \rho(x)\rangle < 0.$$

Отсюда следует, что гипотеза $g_B \in \mathcal{H}$, определенная тем же вектором $\mathbf{w} \in \mathbb{R}^d$, помечает точки в $\rho(B)$ единицей, а точки в $\rho(C \setminus B)$ – нулем. Поскольку это справедливо для всех $B \subseteq C$, получаем, что $|C| = |\rho(C)|$ и $|\mathcal{H}_{\rho(C)}| = 2^{|C|}$, что и завершает доказательство. □

Граница, указанная в этой теореме, точна в том смысле, что существуют отображения Ψ, для которых $\mathrm{Ndim}(\mathcal{H}_\Psi) = \Omega(d)$. Например, это так для многовекторного построения (см. раздел 17.2 и библиографические сведения в конце этой главы). Таким образом, справедливо

Следствие 29.8. *Пусть $\mathcal{X} = \mathbb{R}^n$ и $\Psi : \mathcal{X} \times [k] \to \mathbb{R}^{nk}$ – зависящее от класса отображение в пространство признаков для многовекторного построения*

$$\Psi(\mathbf{x}, y) = [\underbrace{0,\ldots,0}_{\in\mathbb{R}^{(y-1)n}}, \underbrace{x_1,\ldots,x_n}_{\in\mathbb{R}^n}, \underbrace{0,\ldots,0}_{\in\mathbb{R}^{(k-y)n}}].$$

Пусть $\mathcal{H}_\Psi$ определен, как в (29.1). Тогда для размерности Натараджана $\mathcal{H}_\Psi$ имеют место оценки

$$(k-1)(n-1) \le \mathrm{Ndim}(\mathcal{H}_\Psi) \le kn.$$

29.4. О хороших и плохих правилах ERM

В этом разделе мы приведем пример класса гипотез, обладающего тем свойством, что не все правила ERM для него одинаково успешны. Более того, если допустить бесконечное число меток, то мы получим еще и пример класса, который допускает обучение каким-то правилом ERM, но другие правила ERM терпят на нем неудачу. Очевидно, отсюда также следует, что этот класс обучаем, но не обладает свойством равномерной сходимости. Для простоты мы рассмотрим только реализуемый случай.

Мы определим класс следующим образом. Пространство образцов $\mathcal{X}$ будет произвольным конечным или счетным множеством. Пусть $P_f(\mathcal{X})$ – множество всех конечных и коконечных подмножеств $\mathcal{X}$(то есть для любого $A \in P_f(\mathcal{X})$ либо A, либо $\mathcal{X} \setminus A$ должно быть конечным). Множеством меток будет не $[k]$, а $\mathcal{Y} = P_f(\mathcal{X}) \cup \{*\}$, где $*$ – некоторая специальная метка. Для любого $A \in P_f(\mathcal{X})$ определим $h_A : \mathcal{X} \to \mathcal{Y}$:

$$h_A(x) = \begin{cases} A, & \text{если } x \in A \\ *, & \text{если } x \notin A \end{cases}.$$

Наконец, в качестве класса гипотез возьмем

$$\mathcal{H} = \{h_A : A \in P_f(\mathcal{X})\}.$$

Пусть $\mathcal{A}$ – некоторый алгоритм ERM для $\mathcal{H}$. Предположим, что $\mathcal{A}$ применяется к выборке, помеченной гипотезой $h_A \in \mathcal{H}$. Гипотеза h_A – *единственная* в $\mathcal{H}$, которая могла бы вернуть метку A, поэтому если $\mathcal{A}$ видит метку A, то он «знает», что обучена гипотеза h_A, и, будучи алгоритмом ERM, обязан вернуть ее (заметим, что в этом случае ошибка возвращенной гипотезы равна 0). Следовательно, чтобы описать алгоритм ERM, мы должны только определить гипотезу, которую он возвращает, получив выборку вида

$$S = \{(x_1, *), \ldots, (x_m, *)\}.$$

Мы рассмотрим два алгоритма ERM. Первый, $\mathcal{A}_{good}$, определен как

$$\mathcal{A}_{good}(S) = h_\emptyset,$$

то есть он выводит гипотезу, которая предсказывает '*' для любого $x \in \mathcal{X}$. Второй алгоритм, $\mathcal{A}_{bad}$, определен как

$$\mathcal{A}_{bad}(S) = h_{\{x_1,\ldots,x_m\}^c}.$$

Следующее утверждение показывает, что выборочная сложность $\mathcal{A}_{bad}$ приблизительно в $|X|$ раз больше, чем выборочная сложность $\mathcal{A}_{good}$. Тем самым устанавливается существование разрыва между разными алгоритмами ERM. Если $\mathcal{X}$ бесконечно, то мы можем даже получить обучаемый класс, который не обучается никаким алгоритмом ERM.

Утверждение 29.9.

1. *Пусть $\epsilon, \delta > 0$, $\mathcal{D}$ – распределение на $\mathcal{X}$ и $h_A \in \mathcal{H}$. Пусть S – независимая и одинаково распределенная выборка, содержащая $m \geq 1/\epsilon \log(1/\delta)$ примеров, выбранных из*

$\mathcal{D}$ и помеченных h_A. Тогда с вероятностью не менее $1-\delta$ гипотеза, возвращенная алгоритмом $\mathcal{A}_{good}$, будет иметь ошибку не больше ϵ.

2. *Существует постоянная $a > 0$ такая, что для любого $0 < \epsilon < a$ существует распределение $\mathcal{D}$ на $\mathcal{X}$ и гипотеза $h_A \in \mathcal{H}$, для которых справедливо следующее утверждение: гипотеза, возвращенная алгоритмом $\mathcal{A}_{bad}$ после получения выборки размера $m \le (|\mathcal{X}|-1)/6\epsilon$, которая была выбрана из распределения $\mathcal{D}$ и помечена гипотезой h_A, будет иметь ошибку $\ge \epsilon$ с вероятностью $\ge e^{-1/6}$.*

Доказательство. Пусть $\mathcal{D}$ – распределение на $\mathcal{X}$ и предположим, что h_A дает правильную пометку. Для любой выборки $\mathcal{A}_{good}$ возвращает $h_\emptyset$ или h_A. Если он возвращает h_A, то его истинная ошибка равна нулю. Таким образом, он возвращает гипотезу с ошибкой $\ge \epsilon$, только если все m примеров, вошедших в выборку, взяты из $\mathcal{X} \setminus A$, и при этом ошибка $h_\emptyset$, $L_{\mathcal{D}}(h_\emptyset) = \mathrm{P}_{\mathcal{D}}[A]$, больше или равна ϵ. Предположим, что $m \ge 1/\epsilon \log(1/\delta)$; тогда вероятность последнего события не превышает $(1-\epsilon)^m \le \mathrm{e}^{-\epsilon m} \le \delta$. Тем самым пункт 1 доказан.

Теперь докажем пункт 2. Ограничимся случаем, когда $|\mathcal{X}| = d < \infty$. Доказательство для бесконечного $\mathcal{X}$ аналогично. Пусть $\mathcal{X} = \{x_0, \ldots, x_{d-1}\}$. Возьмем $a > 0$ настолько малым, что $1 - 2\epsilon \ge e^{-4}$ для всех $\epsilon < a$. Зафиксируем некоторое $\epsilon < a$. Определим распределение на $\mathcal{X}$, положив $\mathbb{P}[x_0] = 1 - 2\epsilon$ и $\mathbb{P}[x_i] = 2\epsilon/(d-1)$ для всех $1 \le i \le d-1$. Предположим, что правильной гипотезой является $h_\emptyset$, и пусть размер выборки равен m. Очевидно, что гипотеза, возвращенная алгоритмом $\mathcal{A}_{bad}$, ошибается на всех образцах из $\mathcal{X}$, не вошедших в выборку. В силу неравенства Чернова, если $m \le (d-1)/6\epsilon$, то с вероятностью $\ge e^{-1/6}$ выборка включает не более $(d-1)/2$ образцов из $\mathcal{X}$. Таким образом, ошибка возвращенной гипотезы $\ge \epsilon$. □

Из приведенного примера следует, что при многоклассовой классификации выборочная сложность разных алгоритмов ERM может различаться. Верно ли, что для *любого* класса гипотез существуют «хорошие» алгоритмы ERM? Следующее предположение утверждает, что ответ положительный.

Предположение 29.10. *В реализуемом случае выборочная сложность любого класса гипотез $\mathcal{H} \subset [k]^{\mathcal{X}}$ равна*

$$m_{\mathcal{H}}(\epsilon, \delta) = \tilde{O}\left(\frac{\mathrm{Ndim}(\mathcal{H})}{\epsilon}\right).$$

Подчеркнем, что нотация $\tilde{O}$ может скрывать только множители, полиномиально-логарифмически зависящие от ϵ, δ и $\mathrm{Ndim}(\mathcal{H})$, но не зависящие от k.

29.5. Библиографические сведения

Размерность Натараджана введена в работе Natarajan (1989). Там же доказана лемма Натараджана и обобщение фундаментальной теоремы. Обобщения и более точные варианты леммы Натараджана изучались в работе Haussler and Long (1995). В работе Ben-David, Cesa-Bianchi, Haussler and Long (1995) описано большое семейство размерностей – все они обобщают VC-размерность и могут использоваться для оценки выборочной сложности многоклассовой классификации.

Описанные здесь, а также другие примеры вычисления размерности Натараджана можно найти в работе Daniely et al. (2012). Примеры хороших и плохих алгоритмов ERM, а также предположение 29.10, взяты из работы Daniely et al. (2011).

29.6. Упражнения

29.1. Пусть d, $k > 0$. Покажите, что существует класс бинарных гипотез $\mathcal{H}_{\text{bin}}$VC-размерности d такой, что $\text{Ndim}(\mathcal{H}_{\text{bin}}^{\text{OvA,k}}) = d$.

29.2. Докажите лемму 29.6.

29.3. Докажите лемму Натараджана.

Указание. Зафиксируем некоторый образец $x_0 \in \mathcal{X}$. Для $i, j \in [k]$ обозначим $\mathcal{H}_{ij}$ множество всех функций $f : \mathcal{X} \setminus \{x_0\} \to [k]$, которые можно продолжить до функции из $\mathcal{H}$ любым из двух способов: определив $f(x_0) = i$ или $f(x_0) = j$. Покажите, что $|\mathcal{H}| \le |\mathcal{H}_{\mathcal{X}\setminus\{x_0\}}| + \sum_{i\neq j}|\mathcal{H}_{ij}|$ и воспользуйтесь индукцией.

29.4. Адаптируйте доказательство бинарной фундаментальной теоремы и леммы Натараджана для доказательства того, что существует универсальная постоянная $C > 0$ такая, что для любого класса гипотез, имеющего размерность Натараджана d, выборочная сложность $\mathcal{H}$ в агностическом случае ограничена сверху величиной

$$m_{\mathcal{H}}(\epsilon, \delta) \le C \frac{d \log\left(\frac{kd}{\epsilon}\right) + \log(1/\delta)}{\epsilon^2}.$$

29.5. Докажите, что для некоторой универсальной постоянной $C > 0$ и для любого класса гипотез, имеющего размерность Натараджана d, выборочная сложность $\mathcal{H}$ в агностическом случае ограничена снизу величиной

$$m_{\mathcal{H}}(\epsilon, \delta) \ge C \frac{d + \log(1/\delta)}{\epsilon^2}.$$

Указание. Выведите этот факт из бинарной фундаментальной теоремы.

29.6. Пусть $\mathcal{H}$ – класс бинарных гипотез, состоящий из (неоднородных) полупространств в $\mathbb{R}^d$. Цель этого упражнения – доказать, что $\text{Ndim}(\mathcal{H}_{\text{bin}}^{\text{OvA,k}}) \ge (d-1)\cdot(k-1)$.

1. Обозначим $\mathcal{H}_{\text{discrete}}$ класс всех функций $f : [k-1] \times [d-1] \to \{0, 1\}$, для которых существует i_0 такое, что для любого $j \in [d-1]$

$$\forall i < i_0 f(i, j) = 1, \text{ тогда как } \forall i > i_0 f(i, j) = 0.$$

   Покажите, что $\text{Ndim}(\mathcal{H}_{\text{discrete}}^{\text{OvA,k}}) = (d-1)\cdot(k-1)$.
2. Покажите, что $\mathcal{H}_{\text{discrete}}$ можно реализовать с помощью $\mathcal{H}$, т. е. существует отображение $\psi : [k-1] \times [d-1] \to \mathbb{R}^d$ такое, что

$$\mathcal{H}_{\text{discrete}} \subset \{h \circ \psi : h \in \mathcal{H}\}.$$

   Указание. Можно в качестве $\psi(i, j)$ взять вектор, j-я координата которого равна 1, последняя координата равна i, а все остальные равны 0.
3. Выведите отсюда, что $\text{Ndim}(\mathcal{H}^{\text{OvA,k}}) \ge (d-1)\cdot(k-1)$.

Глава 30

ГРАНИЦЫ СЖАТИЯ

В этой книге мы пытались охарактеризовать понятие обучаемости, применяя разные подходы. Сначала мы показали, что успешное обучение гарантируется наличием у класса гипотез свойства равномерной сходимости. Затем мы ввели понятие устойчивости и показали, что устойчивые алгоритмы заведомо являются хорошими обучаемыми. Но существуют и другие свойства, наличия которых может быть достаточно для обучения, и в этой и следующей главе мы рассмотрим два таких подхода: границы сжатия и PAC-байесовский подход.

В этой главе мы будем изучать границы сжатия. Грубо говоря, мы увидим, что если алгоритм обучения может выразить выходную гипотезу, используя лишь малое подмножество обучающего набора, то ошибка гипотезы на остальных примерах может служить оценкой ее истинной ошибки. Иными словами, алгоритм, умеющий «сжимать» свой выход, является хорошим обучаемым.

30.1. Границы сжатия

Чтобы обосновать интерес к этому вопросу, начнем с рассмотрения следующего протокола обучения. Сначала выберем последовательность k примеров и обозначим ее T. По этим примерам мы построим гипотезу, обозначаемую h_T. Теперь мы хотели бы оценить качество h_T, поэтому возьмем новую последовательность $m - k$ примеров (обозначим ее V) и вычислим ошибку h_T на V. Поскольку V и T независимы, мы сразу же получаем следующий результат, воспользовавшись неравенством Бернштейна (см. лемму B.10).

Лемма 30.1. *Предположим, что функция потерь принимает значения из диапазона [0, 1]. Тогда*

$$\mathbb{P}\left[L_{\mathcal{D}}(h_T) - L_V(h_T) \geq \sqrt{\frac{2L_V(h_T)\log(1/\delta)}{|V|}} + \frac{4\log(1/\delta)}{|V|}\right] \leq \delta.$$

Чтобы вывести эту оценку, нам понадобилась только независимость T и V. Поэтому протокол можно переопределить следующим образом. Сначала договоримся о последовательности k индексов $I = (i_1, \ldots, i_k) \in [m]^k$. Затем выберем последовательность m примеров $S = (z_1, \ldots, z_m)$. Теперь определим $T = S_I = (z_{i_1}, \ldots, z_{i_k})$, а к V отнесем все остальные примеры из S. Отметим, что этот протокол эквивалентен определенному ранее, поэтому лемма 30.1 по-прежнему справедлива.

Применяя лемму о границе объединения к случайно выбранной последовательности индексов, получаем такую теорему.

Теорема 30.2. *Пусть k – целое число, и B: $Z^k \to \mathcal{H}$ – отображение последовательностей k примеров в класс гипотез. Пусть $m \geq 2k$ – размер обучающего набора, и $A : Z^m \to \mathcal{H}$ – правило обучения, которое получает обучающую последовательность S размера m и возвращает гипотезу такую, что $A(S) = B(z_{i_1}, \ldots, z_{i_k})$ для некоторых индексов $(i_1, \ldots, i_k) \in [m]^k$. Обозначим $V = \{z_j : j \notin (i_1, \ldots, i_k)\}$ множество примеров, которые не были выбраны при определении $A(S)$. Тогда с вероятностью не менее $1 - \delta$ для случайной выборки S имеет место неравенство*

$$L_{\mathcal{D}}(A(S)) \leq L_V(A(S)) + \sqrt{L_V(A(S))\frac{4k\log(m/\delta)}{m}} + \frac{8k\log(m/\delta)}{m}.$$

Доказательство. Для любого $I \in [m]^k$ обозначим $h_I = B(z_{i_1}, \ldots, z_{i_k})$. Пусть $n = m - k$. Объединяя лемму 30.1 с леммой о границе объединения, получаем

$$\begin{aligned}
&\mathbb{P}\left[\exists I \in [m]^k \text{ такое, что } L_{\mathcal{D}}(h_I) - L_V(h_I) \geq \sqrt{\frac{2L_V(h_I)\log(1/\delta)}{n}} + \frac{4\log(1/\delta)}{n}\right] \\
&\leq \sum_{I\in[m]^k} \mathbb{P}\left[L_{\mathcal{D}}(h_I) - L_V(h_I) \geq \sqrt{\frac{2L_V(h_I)\log(1/\delta)}{n}} + \frac{4\log(1/\delta)}{n}\right] \\
&\leq m^k\delta.
\end{aligned}$$

Обозначим $\delta' = m^k\delta$. По предположению, $k \leq m/2$, т. е. $n = m - k \geq m/2$, поэтому из предыдущего неравенства следует, что с вероятностью не менее $1 - \delta$ имеет место

$$L_{\mathcal{D}}(A(S)) \leq L_V(A(S)) + \sqrt{L_V(A(S))\frac{4k\log(m/\delta')}{m}} + \frac{8k\log(m/\delta')}{m},$$

что и требовалось доказать. □

Отсюда сразу же вытекает

Следствие 30.3. *Предположим, что выполняются условия теоремы 30.2 и что дополнительно $L_V(A(S)) = 0$. Тогда с вероятностью не менее $1 - \delta$ имеет место неравенство*

$$L_{\mathcal{D}}(A(S)) \leq \frac{8k\log(m/\delta)}{m}.$$

Эти результаты приводят к следующему определению.

Определение 30.4 (схема сжатия). Пусть $\mathcal{H}$ – класс гипотез, состоящий из функций из $\mathcal{X}$ в $\mathcal{Y}$, и k – целое число. Говорят, что $\mathcal{H}$ имеет схему сжатия размера k, если выполняются следующие условия:

Для любого m существуют $A : Z^m \to [m]^k$ и $B : Z^k \to \mathcal{H}$ такие, что для всех $h \in \mathcal{H}$, если подать любой обучающий набор вида $(x_1, h(x_1)), \ldots, (x_m, h(x_m))$ на вход A, а затем подать набор $(x_{i_1}, h(x_{i_1})), \ldots, (x_{i_k}, h(x_{i_k}))$ на вход B, где $(i_1, \ldots, i_k)$ – выход A, то выход B, обозначаемый h', удовлетворяет условию $L_S(h') = 0$.

Это определение можно обобщить на нереализуемые последовательности.

Определение 30.5 (схема сжатия для нереализуемых последовательностей). Пусть $\mathcal{H}$ – класс гипотез, состоящий из функций из $\mathcal{X}$ в $\mathcal{Y}$, и k – целое число. Говорят, что $\mathcal{H}$ имеет схему сжатия размера k, если выполняются следующие условия.

Для любого m существуют $A : Z^m \to [m]^k$ и $B : Z^k \to \mathcal{H}$ такие, что для всех $h \in \mathcal{H}$, если подать любой обучающий набор вида $(x_1, y_1), \ldots, (x_m, y_m)$ на вход A, а затем подать набор $(x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k}))$ на вход B, где $(i_1, \ldots, i_k)$ – выход A, то выход B, обозначаемый h', удовлетворяет условию $L_S(h') \leq L_S(h)$.

Следующая лемма показывает, что существование схемы сжатия для реализуемого случая означает также существование схемы сжатия для нереализуемого случая.

Лемма 30.6. *Пусть $\mathcal{H}$ – класс гипотез для бинарной классификации, и предположим, что у него есть схема сжатия размера k в реализуемом случае. Тогда у него есть схема сжатия размера k и для нереализуемого случая.*

Доказательство. Рассмотрим следующую схему. Сначала найдем ERM-гипотезу и обозначим ее h. Затем отбросим все примеры, на которых h ошибается. Теперь применим реализуемую схему сжатия к удаленным примерам. Выход реализуемой схемы сжатия, обозначенный h', должен быть правилен на оставленных примерах. Поскольку h ошибается на удаленных примерах, ошибка h' не может быть больше ошибки, следовательно, h' – тоже ERM-гипотеза. □

30.2. Примеры

В последующих примерах мы представим схемы сжатия для нескольких классов гипотез для бинарной классификации. Ввиду леммы 30.6 мы сосредоточимся только на реализуемом случае. Стало быть, чтобы показать, что у некоторого класса гипотез есть схема сжатия, необходимо показать, что существуют A, B и k, для которых $L_S(h') = 0$.

30.2.1. Осепараллельные прямоугольники

Заметим, что это бесконечный несчетный класс. Мы покажем, что для него существует простая схема сжатия. Рассмотрим алгоритм A, работающий следующим образом. Для каждой размерности выберем два положительных примера с крайними значениями по этому измерению. Определим B как функцию, возвращающую минимальный объемлющий прямоугольник. Тогда для $k = 2d$ в реализуемом случае имеем $L_S(B(A(S))) = 0$.

30.2.2. Полупространства

Пусть $\mathcal{X} = \mathbb{R}^d$, рассмотрим класс однородных полупространств, $\{\mathbf{x} \mapsto \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle) : \mathbf{w} \in \mathbb{R}^d\}$.

Схема сжатия

Без ограничения общности можно считать, что все метки положительные (иначе заменим $\mathbf{x}_i$ на $y_i\mathbf{x}_i$). Схема сжатия будет устроена следующим образом. Сначала A находит вектор $\mathbf{w}$, принадлежащий выпуклой оболочке $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ и имеющий минимальную норму. Затем этот вектор представляется в виде выпуклой комбинации d точек из выборки (ниже будет показано, что это всегда возможно) Выходом A являются эти d точек. Алгоритм B получает эти d точек и в качестве $\mathbf{w}$ берет вектор с минимальной нормой, принадлежащий их выпуклой оболочке.

Докажем, что это действительно схема сжатия. Поскольку данные линейно разделимы, выпуклая оболочка $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ не содержит начала координат. Рассмотрим ближайшую к началу координат точку $\mathbf{w}$, принадлежащую этой выпуклой оболочке (такая точка единственна и является евклидовой проекцией начала координат на выпуклую оболочку). Мы утверждаем, что $\mathbf{w}$ разделяет данные[1]. Чтобы убедиться в этом, предположим противное – что $\langle \mathbf{w}, \mathbf{x}_i \rangle \le 0$ для некоторого i. Возьмем $\mathbf{w}' = (1 - \alpha)\mathbf{w} + \alpha\mathbf{x}_i$ для $\alpha = \|\mathbf{w}\|^2/(\|\mathbf{x}_i\|^2 + \|\mathbf{w}\|^2) \in (0, 1)$. Тогда $\mathbf{w}'$ также принадлежит выпуклой оболочке и

$$
\begin{aligned}
\|\mathbf{w}'\|^2 &= (1-\alpha)^2\|\mathbf{w}\|^2 + \alpha^2\|\mathbf{x}_i\|^2 + 2\alpha(1-\alpha)\langle \mathbf{w}, \mathbf{x}_i \rangle \\
&\le (1-\alpha)^2\|\mathbf{w}\|^2 + \alpha^2\|\mathbf{x}_i\|^2 \\
&= \frac{\|\mathbf{x}_i\|^4\|\mathbf{w}\|^2 + \|\mathbf{x}_i\|^2\|\mathbf{w}\|^4}{(\|\mathbf{w}\|^2 + \|\mathbf{x}_i\|^2)^2} \\
&= \frac{\|\mathbf{x}_i\|^2\|\mathbf{w}\|^2}{\|\mathbf{w}\|^2 + \|\mathbf{x}_i\|^2} \\
&= \|\mathbf{w}\|^2 \cdot \frac{1}{\|\mathbf{w}\|^2/\|\mathbf{x}_i\|^2 + 1} \\
&< \|\mathbf{w}\|^2.
\end{aligned}
$$

Мы пришли к противоречию.

Таким образом, мы показали, что $\mathbf{w}$ также является ERM-гипотезой. Наконец, поскольку $\mathbf{w}$ принадлежит выпуклой оболочке примеров, то по теореме Каратеодори $\mathbf{w}$ принадлежит также выпуклой оболочке подмножества, состоящего из $d + 1$ вершин многогранника. Кроме того, из минимальности нормы $\mathbf{w}$ следует, что $\mathbf{w}$ должна находиться на грани многогранника, а значит, может быть представлена в виде выпуклой комбинации d точек.

Остается показать, что $\mathbf{w}$ является также проекцией на многогранник, определенный d точками. Но это понятно. С одной стороны, меньший многогранник является подмножеством большего, поэтому проекция на меньший не может иметь меньшую норму. С другой стороны, $\mathbf{w}$ сама по себе является допустимым решением. Справедливость утверждения теперь следует из единственности проекции.

---

[1] Можно показать, что $\mathbf{w}$ определяет направление решения с максимальным зазором.

30.2.3. Разделение полиномов

Пусть $\mathcal{X} = \mathbb{R}^d$, рассмотрим класс $\mathbf{x} \mapsto \text{sign}(p(x))$, где p – полином степени r.

Заметим, что $p(x)$ можно записать в виде $\langle \mathbf{w}, \psi(\mathbf{x}) \rangle$, где элементами $\psi(x)$ являются все одночлены от $\mathbf{x}$ степени не выше r. Поэтому задача о построении схемы сжатия для $p(\mathbf{x})$ сводится к задаче о построении схемы сжатия для полупространств в $\mathbb{R}^{d'}$, где $d' = O(d^r)$.

30.2.4. Разделение с зазором

Предположим, что обучающий набор разделяется с зазором γ. Алгоритм перцептрона гарантирует, что после не более $1/\gamma^2$ обновлений будет достигнута сходимость к решению, которое не делает ни одной ошибки на всем обучающем наборе. Стало быть, мы имеем схему сжатия размера $k \leq 1/\gamma^2$.

ЛАНЬ®

30.3. Библиографические сведения

Схемы сжатия и их связь с обучением впервые изучались в работе Littlestone and Warmuth (1986). Как мы показали, если у класса есть схема сжатия, то он допускает обучение. Для проблем бинарной классификации из фундаментальной теоремы обучения следует, что VC-размерность такого класса конечна. Обратное утверждение – о том, что у любого класса конечной VC-размерности есть схема сжатия, – сформулировано Манфредом Вармутом, но до сих пор не доказано и не опровергнуто (см. также Floyd, 1989; Floyd & Warmuth, 1995; Ben-David & Litman, 1998; Livni & Simon, 2013).

ЛАНЬ®

PAC-БАЙЕСОВСКИЙ ПОДХОД

Принципы минимальной длины описания (MDL) и бритвы Оккама допускают потенциально очень большие классы гипотез, но определяют иерархию гипотез и предпочитают гипотезы, находящиеся на верхних ступенях иерархии. В этой главе мы опишем PAC-байесовский подход, который является обобщением этой идеи – априорное знание выражается путем определения априорного распределения на классе гипотез.

31.1. PAC-байесовские границы

Как и в парадигме MDL, мы определяем иерархию гипотез в нашем классе $\mathcal{H}$. Только теперь иерархия принимает вид априорного распределения на $\mathcal{H}$, т. е. мы назначаем вероятность (или плотность вероятности, если $\mathcal{H}$ – непрерывное множество) $P(h) \geq 0$ каждой гипотезе $h \in \mathcal{H}$ и называем $P(h)$ априорной оценкой h. Согласно принципу байесовского рассуждения, выходом алгоритма обучения необязательно является единственная гипотеза. Процесс обучения может определить апостериорное распределение вероятностей на $\mathcal{H}$, которое мы обозначим Q. В контексте обучения с учителем, когда $\mathcal{H}$ содержит функции из $\mathcal{X}$ в $\mathcal{Y}$, мы можем считать, что Q следующим образом определяет рандомизированное правило предсказания. Получая новый образец $\mathbf{x}$, мы выбираем случайную гипотезу $h \in \mathcal{H}$ из распределения Q и предсказываем $h(\mathbf{x})$. Определим потерю Q на примере z как

$$\ell(Q, z) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{h \sim Q}[\ell(h, z)].$$

В силу линейности математического ожидания, потерю обобщения и потерю обучения Q можно записать как

$$L_{\mathcal{D}}(Q) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{h \sim Q}[L_{\mathcal{D}}(h)] \quad \text{и} \quad L_S(Q) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{h \sim Q}[L_S(h)].$$

Следующая теорема утверждает, что разность между потерей обобщения и эмпирической потерей апостериорного распределения Q ограничена выражением, которой зависит от расхождения Кульбака–Лейблера между Q и априорным распределением P. Расхождение Кульбака–Лейблера – это естественная мера расстояния между двумя распределениями. Теорема говорит, что если мы хотим

минимизировать потерю обобщения Q, то должны совместно минимизировать эмпирическую потерю Q и расхождение Кульбака–Лейблера между Q и априорным распределением. Ниже мы покажем, что в некоторых случаях эта идея приводит к принципу минимизации регуляризированного риска.

Теорема 31.1. *Пусть $\mathcal{D}$ – произвольное распределение на множестве примеров Z. Пусть $\mathcal{H}$ – класс гипотез, и $\ell : \mathcal{H} \times Z \to [0, 1]$ – функция потерь. Пусть P – априорное распределение на $\mathcal{H}$, и $\delta \in (0, 1)$. Тогда с вероятностью не менее $1 - \delta$ для независимого и одинаково распределенного обучающего набора $S = \{z_1, \ldots, z_m\}$, случайно выбранного из распределения $\mathcal{D}$, для всех распределений Q на $\mathcal{H}$ (даже таких, которые зависят от S) имеет место неравенство*

$$L_{\mathcal{D}}(Q) \le L_S(Q) + \sqrt{\frac{D(Q\,||\,P) + \ln m/\delta}{2(m-1)}},$$

где

$$D(Q\,||\,P) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{h\sim Q}[\ln(Q(h)/P(h))]$$

– расхождение Кульбака–Лейблера.

ЛАНЬ®

Доказательство. Для любой функции $f(S)$, согласно неравенству Маркова:

$$\mathop{\mathbb{P}}_{S}[f(S) \ge \epsilon] = \mathop{\mathbb{P}}_{S}[e^{f(S)} \ge e^{\epsilon}] \le \frac{\mathbb{E}_S[e^{f(S)}]}{e^{\epsilon}}. \tag{31.1}$$

Положим $\Delta(h) = L_{\mathcal{D}}(h) - L_S(h)$. Мы применим неравенство (31.1) к функции

$$f(S) = \sup_Q \left(2(m-1)\mathop{\mathbb{E}}_{h\sim Q}(\Delta(h))^2 - D(Q\,||\,P)\right).$$

Теперь займемся ограничением $\mathbb{E}_S[e^{f(S)}]$. Основная хитрость – ограничить сверху $f(S)$ выражением, которое не зависит от Q, но зависит от априорного распределения P. Для этого зафиксируем S и заметим, что из определения $D(Q||P)$ следует, что для всех Q

$$\begin{aligned} 2(m-1)\mathop{\mathbb{E}}_{h\sim Q}(\Delta(h))^2 - D(Q\,||\,P) &= \mathop{\mathbb{E}}_{h\sim Q}[\ln(e^{2(m-1)\Delta(h)^2} P(h)/Q(h))] \\ &\le \ln \mathop{\mathbb{E}}_{h\sim Q}[e^{2(m-1)\Delta(h)^2} P(h)/Q(h)] \\ &= \ln \mathop{\mathbb{E}}_{h\sim P}[e^{2(m-1)\Delta(h)^2}], \end{aligned} \tag{31.2}$$

где неравенство вытекает из неравенства Йенсена и выпуклости логарифмической функции. Поэтому

$$\mathop{\mathbb{E}}_{S}(e^{f(S)}) \le \mathop{\mathbb{E}}_{S}\mathop{\mathbb{E}}_{h\sim P}[e^{2(m-1)\Delta(h)^2}]. \tag{31.3}$$

Преимущество выражения в правой части проистекает из того факта, что мы можем изменить порядок взятия математических ожиданий (поскольку P – априорное распределение, не зависящее от S), что дает

$$\mathop{\mathbb{E}}_{S}(e^{f(S)}) \leq \mathop{\mathbb{E}}_{h\sim P}\mathop{\mathbb{E}}_{S}[e^{2(m-1)\Delta(h)^2}]. \tag{31.4}$$

Далее мы утверждаем, что для всех h имеет место $\mathbb{E}_S[e^{2(m-1)\Delta(h)^2}] \leq m$. Чтобы доказать это, вспомним неравенство Хёфдинга, согласно которому

$$\mathop{\mathbb{P}}_{S}[\Delta(h) \geq \epsilon] \leq e^{-2m\epsilon^2}.$$

Отсюда следует, что $\mathbb{E}_S[e^{2(m-1)\Delta(h)^2}] \leq m$ (см. упражнение 31.1). Объединяя это с неравенством (31.4) и подставляя в (31.1), получаем

$$\mathop{\mathbb{P}}_{S}[f(S) \geq \epsilon] \leq \frac{m}{e^{\epsilon}}. \tag{31.5}$$

Обозначим правую часть этого неравенства δ, так что $\epsilon = \ln(m/\delta)$. Теперь мы получили, что с вероятностью не менее $1 - \delta$ для любого распределения Q имеем

$$2(m-1)\mathop{\mathbb{E}}_{h\sim Q}(\Delta(h))^2 - D(Q\,||\,P) \leq \epsilon = \ln(m/\delta).$$

Перегруппируя члены и снова используя неравенство Йенсена (функция x^2 выпуклая), заключаем, что

$$\left(\mathop{\mathbb{E}}_{h\sim Q}\Delta(h)\right)^2 \leq \mathop{\mathbb{E}}_{h\sim Q}(\Delta(h)) \leq \frac{\ln(m/\delta) + D(Q\,||\,P)}{2(m-1)}. \tag{31.6}$$

□

Замечание 31.1 (регуляризация). PAC-байесовская граница приводит к такому правилу обучения:

При заданном априорном распределении P вернуть апостериорное распределение Q, которое минимизирует функцию

$$L_S(Q) + \sqrt{\frac{D(Q\,||\,P) + \ln m/\delta}{2(m-1)}}. \tag{31.6}$$

Это правило похоже на принцип *минимизации регуляризированного риска*. То есть мы совместно минимизируем эмпирическую потерю Q на выборке и расхождение Кульбака–Лейблера между Q и P.

31.2. Библиографические сведения

PAC-байесовские границы были введены в работе McAllester (1998). См. также McAllester, 1999; McAllester, 2003; Seeger, 2003; Langford & Shawe-Taylor, 2003; Langford, 2006.

ЛАНЬ®

31.3. Упражнения

31.1. Пусть X – случайная величина, удовлетворяющая условию $\mathbb{P}[X \geq \epsilon] \leq e^{-2m\epsilon^2}$. Докажите, что $\mathbb{E}[e^{2(m-1)X^2}] \leq m$.

31.2.

- Предположим, что $\mathcal{H}$ – конечный класс гипотез, пусть априорное распределение – равномерное распределение на $\mathcal{H}$, а апостериорное задано так: $Q(h_S) = 1$ для некоторой гипотезы h_S и $Q(h) = 0$ для всех остальных $h \in \mathcal{H}$. Покажите, что

$$L_{\mathcal{D}}(h_S) \leq L_S(h) + \sqrt{\frac{\ln(|\mathcal{H}|) + \ln\, m/\delta}{2(m-1)}}.$$

  Сравните с границами, которые мы вывели, используя равномерную сходимость.
- Выведите границу, аналогичную границе Оккама из главы 7, используя PAC-байесовскую границу.

ЛАНЬ®

ТЕХНИЧЕСКИЕ ЛЕММЫ

ЛАНЬ®

Лемма A.1. *Пусть* $a > 0$. *Тогда* $x \geq 2a\log(a) \Rightarrow x \geq a\log(x)$. *Отсюда следует, что необходимым условием неравенства* $x < a\log(x)$ *является справедливость неравенства* $x < 2a\log(a)$.

Доказательство. Сначала заметим, что для $a \in (0, \sqrt{e}]$ неравенство $x \geq a\log(x)$ выполняется безусловно, и, следовательно, утверждение тривиально. Далее будем считать, что $a > \sqrt{e}$. Рассмотрим функцию $f(x) = x - a\log(x)$. Ее производная равна $f'(x) = 1 - a/x$. Как видим, для $x > a$ производная положительна и функция возрастает. Кроме того,

$$\begin{aligned} f(2a\log(a)) &= 2a\log(a) - a\log(2a\log(a)) \\ &= 2a\log(a) - a\log(a) - a\log(2\log(a)) \\ &= a\log(a) - a\log(2\log(a)). \end{aligned}$$

Поскольку $a - 2\log(a) > 0$ для всех $a > 0$, лемма доказана. □

Лемма A.2. Пусть $a \geq 1$ и $b > 0$. Тогда $x \geq 4a\log(2a) + 2b \Rightarrow x \geq a\log(x) + b$.

Доказательство. Достаточно доказать, что если $x \geq 4a\log(2a) + 2b$, то выполняются оба неравенства $x \geq 2a\log(x)$ и $x \geq 2b$. Так как мы предполагаем, что $a \geq 1$, то, очевидно, $x \geq 2b$. Кроме того, поскольку $b > 0$, имеем $x \geq 4a\log(2a)$, откуда в силу леммы A.1 следует, что $x \geq 2a\log(x)$. Что и требовалось доказать. □

Лемма A.3. *Пусть* X *– случайная величина и* $x' \in \mathbb{R}$ *– скаляр. Предположим, что существует такое* $a > 0$, *что для всех* $t \geq 0$ *имеет место неравенство* $\mathbb{P}[|X - x'| > t] \leq 2e^{-t2/a2}$. *Тогда* $\mathbb{E}[|X - x'|] \leq 4a$.

Доказательство. Для любого $i = 0, 1, 2, \ldots$ обозначим $t_i = ai$. Поскольку последовательность t_i монотонно возрастает, $\mathbb{E}[|X - x'|]$ не превосходит $\sum_{i=1}^{\infty} t_i \mathbb{P}[|X - x'| > t_{i-1}]$. В сочетании с предположением леммы получаем, что $\mathbb{E}[|X - x'|] \leq 2a\sum_{i=1}^{\infty} ie^{-(i-1)2}$. Теперь справедливость леммы вытекает из неравенств

$$\sum_{i=1}^{\infty} ie^{-(i-1)^2} \leq \sum_{i=1}^{5} ie^{-(i-1)^2} + \int_5^{\infty} xe^{-(x-1)^2} dx < 1.8 + 10^{-7} < 2.$$ □

Лемма A.4. *Пусть* X *– случайная величина и* $x' \in \mathbb{R}$ *– скаляр. Предположим, что существуют такие* $a > 0$ *и* $b \geq e$, *что для всех* $t \geq 0$ *имеет место неравенство* $\mathbb{P}[|X - x'| > t] \leq 2be^{-t2/a2}$. *Тогда* $\mathbb{E}[|X - x'|] \leq a(2 + \sqrt{\log b})$.

Доказательство. Для любого $i = 0, 1, 2, \ldots$ обозначим $t_i = a(i + \sqrt{\log b})$. Последовательность t_i монотонно возрастает, поэтому

$$\mathbb{E}[|X - x'|] \le a\sqrt{\log(b)} + \sum_{i=1}^{\infty} t_i \mathbb{P}[|X - x'| > t_{i-1}].$$

В силу предположения леммы

$$\begin{aligned}
\sum_{i=1}^{\infty} t_i \mathbb{P}[|X - x'| > t_{i-1}] &\le 2ab \sum_{i=1}^{\infty} (i + \sqrt{\log(b)}) e^{-1(i-1+\sqrt{\log(b)})^2} \\
&\le 2ab \int_{1+\sqrt{\log(b)}}^{\infty} x e^{-(x-1)^2} dx \\
&= 2ab \int_{\sqrt{\log(b)}}^{\infty} (y+1) e^{-y^2} dy \\
&\le 4ab \int_{\sqrt{\log(b)}}^{\infty} y e^{-y^2} dy \\
&= 2ab \left[e^{-y^2} \right]_{\sqrt{\log(b)}}^{\infty} \\
&= 2ab / b = 2a.
\end{aligned}$$

В сочетании с предыдущими неравенствами это завершает доказательство. □

Лемма А.5. *Пусть m, d – два целых положительных числа таких, что $d \le m - 2$. Тогда*

$$\sum_{k=0}^{d} \binom{m}{k} \le \left(\frac{em}{d} \right)^d.$$

Доказательство. Мы проведем доказательство по индукции. Для $d = 1$ левая часть равна $1 + m$, а правая – em, поэтому утверждение леммы истинно. Предположим, что утверждение верно для некоторого d, и докажем его справедливость для $d + 1$. По предположению индукции

$$\begin{aligned}
\sum_{k=0}^{d+1} \binom{m}{k} &\le \left(\frac{em}{d} \right)^d + \binom{m}{d+1}. \\
&= \left(\frac{em}{d} \right)^d \left(1 + \left(\frac{d}{em} \right)^d \frac{m(m-1)(m-2)\cdots(m-d)}{(d+1)d!} \right) \\
&\le \left(\frac{em}{d} \right)^d \left(1 + \left(\frac{d}{e} \right)^d \frac{(m-d)}{(d+1)d!} \right).
\end{aligned}$$

Воспользовавшись формулой Стирлинга, имеем далее

$$\le \left(\frac{em}{d} \right)^d \left(1 + \left(\frac{d}{e} \right)^d \frac{(m-d)}{(d+1)\sqrt{2\pi d}(d/e)^d} \right)$$

$$
\begin{aligned}
&= \left(\frac{em}{d}\right)^d \left(1 + \frac{(m-d)}{\sqrt{2\pi d}(d+1)}\right) \\
&= \left(\frac{em}{d}\right)^d \cdot \frac{d+1+(m-d)/\sqrt{2\pi d}}{d+1} \\
&\le \left(\frac{em}{d}\right)^d \cdot \frac{d+1+(m-d)/2}{d+1} \\
&= \left(\frac{em}{d}\right)^d \cdot \frac{d/2+1+m/2}{d+1} \\
&\le \left(\frac{em}{d}\right)^d \cdot \frac{m}{d+1},
\end{aligned}
$$

причем в последнем неравенстве мы использовали предположение $d \le m - 2$. С другой стороны,

$$
\begin{aligned}
\left(\frac{em}{d+1}\right)^{d+1} &= \left(\frac{em}{d}\right)^d \frac{em}{d+1} \cdot \left(\frac{d}{d+1}\right)^d \\
&= \left(\frac{em}{d}\right)^d \cdot \frac{em}{d+1} \cdot \frac{1}{(1+1/d)^d} \\
&\ge \left(\frac{em}{d}\right)^d \cdot \frac{em}{d+1} \cdot \frac{1}{e} \\
&= \left(\frac{em}{d}\right)^d \cdot \frac{m}{d+1},
\end{aligned}
$$

поэтому шаг индукции доказан. □

Лемма А.6. Для всех $a \in \mathbb{R}$ имеет место неравенство

$$\frac{e^a + e^{-a}}{2} \le e^{a^2/2}.$$

Доказательство. Заметим, что

$$e^a = \sum_{n=0}^{\infty} \frac{a^n}{n!}.$$

Поэтому

$$\frac{e^a + e^{-a}}{2} = \sum_{n=0}^{\infty} \frac{a^{2n}}{(2n)!}$$

и

$$e^{a^2/2} = \sum_{n=0}^{\infty} \frac{a^{2n}}{2^n n!}.$$

Для завершения доказательства заметим, что $(2n)! \ge 2^n n!$ для любого $n \ge 0$. □

КОНЦЕНТРАЦИЯ МЕРЫ

Пусть $Z_1, \ldots, Z_m$ – последовательность независимых и одинаково распределенных случайных величин, и μ – их среднее. Усиленный закон больших чисел утверждает, что когда m стремится к бесконечности, эмпирическое среднее $(1/m)$ $\sum_{i=1}^{m} Z_i$ стремится к среднему значению μ с вероятностью 1. Неравенства концентрации меры количественно выражают отклонение эмпирического среднего от математического ожидания при конечном m.

B.1. Неравенство Маркова

Начнем с неравенства Маркова. Пусть Z – неотрицательная случайная величина. Математическое ожидание Z можно записать в виде:

$$\mathbb{E}[Z] = \int_{x=0}^{\infty} \mathbb{P}[Z \geq x] dx. \tag{B.1}$$

Так как функция $\mathbb{P}[Z \geq x]$ монотонно не возрастает, получаем, что

$$\forall a \geq 0, \quad \mathbb{E}[Z] \geq \int_{x=0}^{a} \mathbb{P}[Z \geq x] dx \geq \int_{x=0}^{a} \mathbb{P}[Z \geq a] dx = a\mathbb{P}[Z \geq a]. \tag{B.2}$$

Перегруппировка членов приводит к неравенству Маркова

$$\forall a \geq 0, \quad \mathbb{P}[Z \geq a] \leq \frac{\mathbb{E}[Z]}{a}. \tag{B.3}$$

Для случайных величин, принимающих значения в диапазоне [0, 1], из неравенства Маркова можно вывести следующую лемму.

Лемма B.1. *Пусть Z – случайная величина, принимающая значения в диапазоне* [0, 1]. *Предположим, что* $\mathbb{E}[Z] = \mu$. *Тогда для любого* $a \in (0, 1)$

$$\mathbb{P}[Z > 1 - a] \geq \frac{\mu - (1 - a)}{a}.$$

Отсюда также следует, что для любого $a \in (0, 1)$,

$$\mathbb{P}[Z > a] \geq \frac{\mu - a)}{1 - a} \geq \mu - a.$$

Доказательство. Положим $Y = 1 - Z$. Тогда Y– неотрицательная случайная величина, для которой $\mathbb{E}[Y] = 1 - \mathbb{E}[Z] = 1 - \mu$. Применяя неравенство Маркова к Y, получаем

$$\mathbb{P}[Z \le 1-a] = \mathbb{P}[1-Z \ge a] = \mathbb{P}[Y \ge a] \le \frac{\mathbb{E}[Y]}{a} = \frac{1-\mu}{a}.$$

Следовательно,

$$\mathbb{P}[Z > 1-a] \ge 1 - \frac{1-\mu}{a} = \frac{a+\mu-1}{a}. \qquad \square$$

B.2. Неравенство Чебышева

Применяя неравенство Маркова к случайной величине $(Z - \mathbb{E}[Z])^2$, получаем неравенство Чебышева:

$$\forall a > 0, \quad \mathbb{P}[|Z - \mathbb{E}[Z]| \ge a] = \mathbb{P}[(Z - \mathbb{E}[Z])^2 \ge a^2] \le \frac{\mathrm{Var}[Z]}{a^2}, \tag{B.4}$$

где $\mathrm{Var}[Z] = \mathbb{E}[(Z - \mathbb{E}[Z])^2]$ – дисперсия Z.

Рассмотрим случайную величину $(1/m)\sum_{i=1}^{m} Z_i$. Поскольку $Z_1, \ldots, Z_m$ независимы и одинаково распределены, легко убедиться, что

$$\mathrm{Var}\left[\frac{1}{m}\sum_{i=1}^{m} Z_i\right] = \frac{\mathrm{Var}[Z_1]}{m}.$$

Применяя неравенство Чебышева, получаем следующую лемму.

Лемма B.2. *Пусть $Z_1, \ldots, Z_m$ – последовательность независимых и одинаково распределенных случайных величин и предположим, что $\mathbb{E}[Z_1] = \mu$, $\mathrm{Var}[Z_1] \le 1$. Тогда для любого $\delta \in (0, 1)$ с вероятностью не менее $1 - \delta$ имеем*

$$\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \le \sqrt{\frac{1}{\delta m}}.$$

Доказательство. Применяя неравенство Чебышева, получаем, что для всех $a > 0$

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| > a\right] \le \frac{\mathrm{Var}[Z_1]}{ma^2} \le \frac{1}{ma^2}.$$

Для завершения доказательства достаточно обозначить правую часть δ и решить неравенство относительно a. $\square$

Оценка расхождения между эмпирическим средним и математическим ожиданием полиномиально убывает с ростом m. Можно получить значительно более быстрое убывание. В следующих разделах мы выведем границы, убывающие экспоненциально.

B.3. Границы Чернова

Пусть $Z_1, \ldots, Z_m$ – независимые случайные величины Бернулли, т. е. для любого $i\mathbb{P}[Z_i = 1] = p_i$ и $\mathbb{P}[Z_i = 0] = 1 - p_i$. Положим $p = \sum_{i=1}^m p_i$ и $Z = \sum_{i=1}^m Z_i$. В силу монотонности экспоненциальной функции и неравенства Маркова для любого $t > 0$ имеем

$$\mathbb{P}[Z > (1+\delta)p] = \mathbb{P}[e^{tZ} > e^{t(1+\delta)p}] \le \frac{\mathbb{E}[e^{tZ}]}{e^{(1+\delta)tp}}. \tag{B.5}$$

Далее

$$\begin{aligned}
\mathbb{E}[e^{tZ}] &= \mathbb{E}[e^{t\sum_i Z_i}] = \mathbb{E}\left[\prod_i e^{tZ_i}\right] \\
&= \prod_i \mathbb{E}[e^{tZ_i}] && \text{в силу независимости} \\
&= \prod_i (p_i e^t + (1-p_i)e^0) \\
&= \prod_i (1 + p_i(e^t - 1)) \\
&\le \prod_i e^{p_i(e^t-1)} && \text{поскольку } 1 + x \le e^x \\
&= e^{\sum_i p_i(e^t-1)} \\
&= e^{(e^t-1)p}.
\end{aligned}$$

Полагая $t = \log(1 + \delta)$, из этого неравенства в сочетании с (B.5) получаем следующую лемму.

Лемма B.3. *Пусть $Z_1, \ldots, Z_m$ – независимые случайные величины Бернулли, т. е. для любого i $\mathbb{P}[Z_i = 1] = p_i$ и $\mathbb{P}[Z_i = 0] = 1 - p_i$. Обозначим $p = \sum_{i=1}^m p_i$ и $Z = \sum_{i=1}^m Z_i$. Тогда для любого $\delta > 0$*

$$\mathbb{P}[Z > (1 + \delta)p] \le e^{-h(\delta)p},$$

где

$$h(\delta) = (1 + \delta)\log(1 + \delta) - \delta.$$

Воспользовавшись неравенством $h(a) \ge a^2/(2 + 2a/3)$, получаем:

Лемма B.4. *В обозначениях леммы B.3 имеет место также неравенство*

$$\mathbb{P}[Z > (1+\delta)p] \le e^{-p\frac{\delta^2}{2+2\delta/3}}.$$

Для получения оценки снизу применимы аналогичные вычисления:

$$\mathbb{P}[Z < (1-\delta)p] = \mathbb{P}[-Z > -(1-\delta)p] = \mathbb{P}[e^{-tZ} > e^{-t(1-\delta)p}] \le \frac{\mathbb{E}[e^{-tZ}]}{e^{-(1-\delta)tp}} \tag{B.6}$$

и

$$\begin{aligned}
\mathbb{E}[e^{-tZ}] &= \mathbb{E}[e^{-t\sum_i Z_i}] = \mathbb{E}\left[\prod_i e^{-tZ_i}\right] \\
&= \prod_i \mathbb{E}[e^{-tZ_i}] && \text{в силу независимости} \\
&= \prod_i (1 + p_i(e^{-t} - 1)) \\
&\le \prod_i e^{p_i(e^{-t}-1)} && \text{поскольку } 1 + x \le e^x \\
&= e^{(e^{-t}-1)p}.
\end{aligned}$$

Полагая $t = -\log(1 - \delta)$, получаем

$$\mathbb{P}[Z < (1-\delta)p] \le \frac{e^{-\delta p}}{e^{(1-\delta)\log(1-\delta)p}} = e^{-ph(-\delta)}.$$

Легко проверить, что $h(-\delta) \ge h(\delta)$ и потому справедлива

Лемма B.5. *В обозначениях леммы B.3 имеет место также неравенство*

$$\mathbb{P}[Z < (1-\delta)p] \le e^{-ph(-\delta)} \le e^{-ph(\delta)} \le e^{-p\frac{\delta^2}{2+2\delta/3}}.$$

B.4. Неравенство Хёфдинга

Лемма B.6 (неравенство Хёфдинга). *Пусть $Z_1, \ldots, Z_m$ – последовательность независимых и одинаково распределенных случайных величин. Обозначим $\bar{Z} = (1/m)\sum_{i=1}^m Z_i$. Предположим, что $\mathbb{E}[\bar{Z}] = \mu$ и $\mathrm{P}[a \le Z_i \le b] = 1$ для любого i. Тогда для любого $\epsilon > 0$*

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^m Z_i - \mu\right| > \epsilon\right] \le 2\exp(-2m\epsilon^2/(b-a)^2).$$

Доказательство. Обозначим $X_i = Z_i - \mathbb{E}[Z_i]$ и $\bar{X} = (1/m)\sum_{i=1}^m X_i$. В силу монотонности экспоненциальной функции и неравенства Маркова имеем для любых $\lambda > 0$ и $\epsilon > 0$

$$\mathbb{P}[\bar{X} \ge e] = \mathbb{P}[e^{\lambda\bar{X}} \ge e^{\lambda e}] \le e^{-\lambda e}\mathbb{E}[e^{\lambda\bar{X}}].$$

По предположению о независимости имеем также

$$\mathbb{E}[e^{\lambda\bar{X}}] = \mathbb{E}\left[\prod_i e^{\lambda X_i/m}\right] = \prod_i \mathbb{E}[e^{\lambda X_i/m}].$$

По лемме Хёфдинга (лемма B.7 ниже), для любого i имеет место неравенство

$$\mathbb{E}[e^{\lambda X_i/m}] \le e^{\frac{\lambda^2(b-a)^2}{8m^2}}.$$

Следовательно,

$$\mathbb{P}[\bar{X} \geq \epsilon] \leq e^{-\lambda\epsilon} \prod_i e^{\frac{\lambda^2(b-a)^2}{8m^2}} = e^{-\lambda\epsilon + \frac{\lambda^2(b-a)^2}{8m^2}}.$$

Полагая $\lambda = 4m/(b-a)^2$, получаем

$$\mathbb{P}[\bar{X} \geq \epsilon] \leq e^{-\frac{2m\epsilon^2}{(b-a)^2}}.$$

ЛАНЬ®

Применяя точно такие же рассуждения к величине $-\bar{X}$, получаем, что $\mathbb{P}[\bar{X} \leq \epsilon] \leq e^{-\frac{2m\epsilon^2}{(b-a)^2}}$. Для завершения доказательства осталось применить в обоих случаях лемму о границе объединения. □

Лемма B.7 (лемма Хёфдинга). *Пусть X – случайная величина, принимающая значения из отрезка $[a, b]$ и такая, что $\mathbb{E}[X] = 0$. Тогда для любого $\lambda > 0$*

$$\mathbb{E}[e^{\lambda X}] \leq e^{\frac{\lambda^2(b-a)^2}{8}}.$$

Доказательство. Поскольку $f(x) = e^{\lambda x}$ – выпуклая функция, для любых $\alpha \in (0, 1)$ и $x \in [a, b]$ справедливо неравенство

$$f(x) \leq \alpha f(a) + (1 - \alpha) f(b).$$

Полагая $\alpha = (b - x)/(b - a) \in [0, 1]$, получаем

$$e^{\lambda X} \leq \frac{b-x}{b-a} e^{\lambda a} + \frac{x-a}{b-a} e^{\lambda b}.$$

Возьмем математическое ожидание обеих частей

$$\mathbb{E}[e^{\lambda X}] \leq \frac{b-\mathbb{E}[X]}{b-a} e^{\lambda a} + \frac{\mathbb{E}[X]-a}{b-a} e^{\lambda b} = \frac{b}{b-a} e^{\lambda a} + \frac{a}{b-a} e^{\lambda b},$$

где мы воспользовались тем фактом, что $\mathbb{E}[X] = 0$. Обозначим $h = \lambda(b - a)$, $p = -a(b - a)$ и $L(h) = -hp + \log(1 - p + pe^h)$. Тогда выражение в правой части можно записать в виде $e^{L(h)}$. Теперь для завершения доказательства достаточно показать, что $L(h) \leq h^2/8$. Но это следует из разложения в ряд Тейлора и того, что $L(0) = L'(0) = 0$ и $L''(h) \leq 1/4$ для всех h. □

B.5. Неравенства Беннета и Бернштейна

Неравенства Беннета и Бернштейна похожи на границы Чернова, но справедливы для любой последовательности независимых случайных величин. Мы приведем их без доказательства, желающие могут найти его, например, в книге Cesa-Bianchi and Lugosi (2006).

ЛАНЬ®

Лемма B.8 (неравенство Беннета). *Пусть $Z_1, \ldots, Z_m$ – независимые случайные величины с нулевым средним и предположим, что $Z_i \leq 1$ с вероятностью 1. Пусть*

$$\sigma^2 \geq \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}[Z_i^2].$$

Тогда для любого $\epsilon > 0$,

$$\mathbb{P}\left[\sum_{i=1}^{m} Z_i > \epsilon\right] \leq e^{-m\sigma^2 h\left(\frac{\epsilon}{m\sigma^2}\right)},$$

где

$$h(a) = (1 + a)\log(1 + a) - a.$$

Воспользовавшись неравенством $h(a) \geq a^2/(2 + 2a/3)$, можно доказать следующую лемму.

Лемма B.9 (неравенство Бернштейна). *Пусть* $Z_1, \ldots, Z_m$ *– независимые и одинаково распределенные случайные величины с нулевым средним. Если для всех* i $\mathbb{P}(|Z_i| < M) = 1$, *то для любого* $t > 0$

$$\mathbb{P}\left[\sum_{i=1}^{m} Z_i > t\right] \leq \exp\left(-\frac{t^2/2}{\sum \mathbb{E}Z_j^2 + Mt/3}\right).$$

B.5.1. Применение

Неравенством Бернштейна можно воспользоваться для интерполяции между скоростью $1/\epsilon$, выведенной для PAC-обучения в реализуемом случае (глава 2), и скоростью $1/\epsilon^2$, выведенной для нереализуемого случая (глава 4).

Лемма B.10. *Пусть* $\ell : \mathcal{H} \times Z \to [0, 1]$ *– функция потерь. Пусть* $\mathcal{D}$ *– произвольное распределение на* Z. *Зафиксируем некоторое* h. *Тогда для любого* $\delta \in (0, 1)$ *имеют место неравенства*

$$1.\ \mathop{\mathbb{P}}_{S\sim\mathcal{D}^m}\left[L_S(h) \geq L_{\mathcal{D}}(h) + \sqrt{\frac{2L_{\mathcal{D}}(h)\log(1/\delta)}{3m}} + \frac{2\log(1/\delta)}{m}\right] \leq \delta.$$

$$2.\ \mathop{\mathbb{P}}_{S\sim\mathcal{D}^m}\left[L_{\mathcal{D}}(h) \geq L_S(h) + \sqrt{\frac{2L_S(h)\log(1/\delta)}{m}} + \frac{4\log(1/\delta)}{m}\right] \leq \delta.$$

Доказательство. Определим случайные величины $\alpha_1, \ldots, \alpha_m$ такие, что $\alpha_i = \ell(h, z_i) - L_{\mathcal{D}}(h)$. Заметим, что $\mathbb{E}[\alpha_i] = 0$ и что

$$\begin{aligned}\mathbb{E}[\alpha_i^2] &= \mathbb{E}[\ell(h, z_i)^2] - 2L_{\mathcal{D}}(h)\mathbb{E}[\ell(h, z_i)] + L_{\mathcal{D}}(h)^2 \\ &= \mathbb{E}[\ell(h, z_i)^2] - L_{\mathcal{D}}(h)^2 \\ &\leq \mathbb{E}[\ell(h, z_i)^2] \\ &\leq \mathbb{E}[\ell(h, z_i)] = L_{\mathcal{D}}(h),\end{aligned}$$

где в последнем неравенстве мы воспользовались тем, что $\ell(h, z_i) \in [0, 1]$ и, следовательно, $\ell(h, z_i)^2 \leq \ell(h, z_i)$. Применяя неравенство Бернштейна к α_i, получаем

$$\mathbb{P}\left[\sum_{i=1}^{m}\alpha_i > t\right] \le \exp\left(-\frac{t^2/2}{\sum \mathbb{E}\alpha_j^2 + t/3}\right)$$
$$\le \exp\left(-\frac{t^2/2}{mL_{\mathcal{D}}(h) + t/3}\right) \stackrel{\text{def}}{=} \delta.$$

Решение относительно t дает

$$\frac{t^2/2}{mL_{\mathcal{D}}(h) + t/3} = \log(1/\delta)$$
$$\Rightarrow t^2/2 - \frac{\log(1/\delta)}{3}t - \log(1/\delta)mL_{\mathcal{D}}(h) = 0$$
$$\Rightarrow t = \frac{\log(1/\delta)}{3} + \sqrt{\frac{\log^2(1/\delta)}{3^2} + 2\log(1/\delta)mL_{\mathcal{D}}(h)}$$
$$\le 2\frac{\log(1/\delta)}{3} + \sqrt{2\log(1/\delta)mL_{\mathcal{D}}(h)}.$$

Поскольку $(1/m)\sum_i \alpha_i = L_S(h) - L_{\mathcal{D}}(h)$, то с вероятностью не менее $1 - \delta$

$$L_S(h) - L_{\mathcal{D}}(h) \le 2\frac{\log(1/\delta)}{3m} + \sqrt{\frac{2\log(1/\delta)L_{\mathcal{D}}(h)}{m}},$$

что доказывает первое неравенство. Второе неравенство доказывается аналогично. □

B.6. Неравенство Слада

Пусть X – биномиальная случайная величина с параметрами (m, p), т. е. $X = \sum_{i=1}^{m} Z_i$, где каждая Z_i равна 1 с вероятностью p и 0 с вероятностью $1 - p$. Предположим, что $p = (1 - \epsilon)/2$. Неравенство Слада (Slud, 1977) утверждает, что вероятность $\mathbb{P}[X \ge m/2]$ ограничена снизу вероятностью того, что нормальная случайная величина будет больше или равна $\sqrt{m\epsilon^2/(1-\epsilon^2)}$. Следующая лемма вытекает из стандартной оценки границ хвостов нормального распределения.

Лемма B.11. *Пусть X – биномиальная случайная величина с параметрами (m, p), и предположим, что $p = (1 - \epsilon)/2$. Тогда*

$$\mathbb{P}[X \ge m/2] \ge \frac{1}{2}\left(1 - \sqrt{1 - \exp(-m\epsilon^2/(1-\epsilon^2))}\right).$$

B.7. Концентрация случайных величин χ^2

Пусть $X_1, \ldots, X_k$ – k независимых нормально распределенных случайных величин, т. е. для всех i $X_i \sim N(0, 1)$. Распределение случайной величины X_i^2 называется χ^2 (х и квадрат), а распределение случайной величины $Z = X_1^2 + \ldots + X_k^2$ называется χ_k^2

(хи-квадрат с k степенями свободы). Очевидно, что $\mathbb{E}[X_i^2] = 1$ и $\mathbb{E}[Z] = k$. Следующая лемма утверждает, что X_k^2 сконцентрирована вокруг своего среднего.

Лемма B.12. *Пусть $Z \sim \chi_k^2$. Тогда для любого $\epsilon > 0$ имеет место неравенство*

$$\mathbb{P}[Z \le (1-\epsilon)k] \le e^{-\epsilon^2 k/6},$$

и для любого $\epsilon \in (0, 3)$ имеет место неравенство

$$\mathbb{P}[Z \ge (1+\epsilon)k] \le e^{-\epsilon^2 k/6}.$$

Наконец, для любого $\epsilon \in (0, 3)$

$$\mathbb{P}[(1-\epsilon)k \le Z \le (1+\epsilon)k] \ge 1 - 2e^{-\epsilon^2 k/6}.$$

Доказательство. Запишем $Z = \sum_{i=1}^k X_i^2$, где $X_i \sim N(0, 1)$. Для доказательства обеих границ воспользуемся методом Чернова. В случае первого неравенства сначала ограничим $\mathbb{E}[e^{-\lambda X_1^2}]$, где $\lambda > 0$ будет определено позже. Так как $e^{-a} \le 1 - a + a^2/2$ для любого $a \ge 0$, то

$$\mathbb{E}[e^{-\lambda X_1^2}] \le 1 - \lambda \mathbb{E}[X_1^2] + \frac{\lambda^2}{2}\mathbb{E}[X_1^4].$$

Пользуясь хорошо известными тождествами $\mathbb{E}[X_1^2] = 1$ и $\mathbb{E}[X_1^4] = 3$ и тем фактом, что $1 - a \le e^{-a}$, получаем

$$\mathbb{E}[e^{-\lambda X_1^2}] \le 1 - \lambda + \frac{3}{2}\lambda^2 \le e^{-\lambda + \frac{3}{2}\lambda^2}.$$

Теперь, применяя метод ограничения Чернова, получаем

$$\begin{aligned}
\mathbb{P}[-Z \ge -(1-\epsilon)k] &= \mathbb{P}[e^{-\lambda Z} \ge e^{-(-\epsilon)k\lambda}] \\
&\le e^{(1-\epsilon)k\lambda}\mathbb{E}[e^{-\lambda Z}] \\
&= e^{(1-\epsilon)k\lambda}\left(\mathbb{E}[e^{-\lambda X_1^2}]\right)^k \\
&\le e^{(1-\epsilon)k\lambda} e^{-\lambda k + \frac{3}{2}\lambda^2 k} \\
&= e^{-\epsilon k\lambda + \frac{3}{2}k\lambda^2}.
\end{aligned}$$

Взяв $\lambda = \epsilon/3$, мы докажем первое из неравенств в лемме.

Чтобы доказать второе неравенство, воспользуемся известным выражением для функции, порождающей моменты случайной величины с распределением χ_k^2:

$$\forall \lambda < \frac{1}{2}, \quad \mathbb{E}[e^{\lambda Z^2}] = (1-2\lambda)^{-k/2}. \tag{B.7}$$

Из этого выражения мы, пользуясь методом Чернова, получаем

$$\begin{aligned}\mathbb{P}[Z \geq (1+\epsilon)k)] &= \mathbb{P}e^{\lambda Z} \geq e^{(1+\epsilon)k\lambda} \\ &\leq e^{-(1+\epsilon)k\lambda}\mathbb{E}e^{\lambda Z} \\ &= e^{-(1+\epsilon)k\lambda}(1-2\lambda)^{-k/2} \\ &\leq e^{-(1+\epsilon)k\lambda}e^{k\lambda} = e^{-k\lambda},\end{aligned}$$

где последнее неравенство справедливо, потому что $1 - a \leq e^{-a}$. Полагая $\lambda = \epsilon/6$ (по нашему предположению, эта величина попадает в интервал $(0, 1/2)$), мы получаем второе неравенство из леммы.

Наконец, последнее неравенство следует из первых двух и леммы о границе объединения. □

ЛАНЬ®

ЛАНЬ®

ЛИНЕЙНАЯ АЛГЕБРА

C.1. Основные определения

В этом приложении мы будем говорить только о линейной алгебре конечномерного евклидова пространства. Под векторами мы будем понимать векторы-столбцы.

Пусть даны два d-мерных вектора $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$. Их скалярным произведением называется выражение

$$\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=1}^{d} u_i v_i .$$

Евклидовой нормой (или нормой ℓ_2) называется величина $||\mathbf{u}|| = \sqrt{\langle \mathbf{u}, \mathbf{u} \rangle}$. Мы будем также использовать норму ℓ_1, $||\mathbf{u}||_1 = \sum_{i=1}^{d} |u_i|$ и норму ℓ_∞, $||\mathbf{u}||_\infty = \max_i |u_i|$.

Подпространством $\mathbb{R}^d$ называется подмножество $\mathbb{R}^d$, замкнутое относительно операций сложения и умножения на скаляр. Линейной оболочкой векторов $\mathbf{u}_1, \ldots, \mathbf{u}_k$ называется подпространство, состоящее из всех векторов вида

$$\sum_{i=1}^{k} \alpha_i \mathbf{u}_i .$$

Множество векторов $U = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ называется линейно независимым, если для любого i вектор $\mathbf{u}_i$ не принадлежит линейной оболочке векторов $\mathbf{u}_1, \ldots, \mathbf{u}_{i-1}, \mathbf{u}_{i+1}, \ldots, \mathbf{u}_k$. Говорят, что множество U порождает подпространство V или что V натянуто на U, если V является линейной оболочкой векторов из U. Говорят, что множество векторов U является *базисом* V, если оно одновременно порождает V и является линейно независимым. Размерностью V называется размер базиса V (и можно доказать, что все базисы V имеют одинаковый размер). Говорят, что U – ортогональное множество, если для всех $i \neq j$ $\langle \mathbf{u}_i, \mathbf{u}_j \rangle = 0$. Говорят, что U – ортонормированное множество, если оно ортогонально и для любого i $||\mathbf{u}_i|| = 1$.

Если дана матрица $A \in \mathbb{R}^{n,d}$, то ее образом называется линейная оболочка столбцов, а ядром – подпространство всех векторов, для которых $A\mathbf{u} = \mathbf{0}$. Рангом A называется размерность ее образа.

Результатом транспонирования матрицы A (обозначается $A^\top$) называется матрица, в позиции (i, j) которой находится элемент матрицы A в позиции (j, i). Говорят, что матрица A симметрична, если $A^\top = A$.

C.2. Собственные значения и собственные векторы

Пусть $A \in \mathbb{R}^{d,d}$ – матрица. Ненулевой вектор $\mathbf{u}$ называется собственным вектором A, соответствующим собственному значению λ, если

$$A\mathbf{u} = \lambda\mathbf{u}.$$

Теорема C.1 (спектральное разложение). *Если $A \in \mathbb{R}^{d,d}$ – симметричная матрица ранга k, то существует ортонормированный базис $\mathbb{R}^d$, $\mathbf{u}_1, \ldots, \mathbf{u}_d$, такой, что каждый вектор $\mathbf{u}_i$ является собственным вектором A. Более того, A можно записать в виде $A = \sum_{i=1}^{d} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top$, где каждое λ_i – собственное значение, соответствующее собственному вектору $\mathbf{u}_i$. Тоже самое можно записать в виде $A = UDU^\top$, где столбцами U являются векторы $\mathbf{u}_1, \ldots, \mathbf{u}_d$, а D – диагональная матрица, в которой $D_{i,i} = \lambda_i$ и $D_{i,j} = 0$ для $i \neq j$. Наконец, количество ненулевых λ_i равно рангу матрицы, собственные векторы, соответствующие ненулевым собственным значениям, порождают образ матрицы, а собственные векторы, соответствующие нулевым собственным значениям, порождают ядро матрицы.*

C.3. Положительно определенные матрицы

Симметричная матрица $A \in \mathbb{R}^{d,d}$ называется положительно определенной, если все ее собственные значения положительны. Матрица A называется положительно полуопределенной, если все ее собственные значения неотрицательны.

Теорема C.2. *Пусть $A \in \mathbb{R}^{d,d}$ – симметричная матрица. Тогда следующие определения положительной полуопределенности A эквивалентны:*

- *все собственные значения A неотрицательны*;
- *для любого вектора $\mathbf{u}\langle \mathbf{u}, A\mathbf{u}\rangle \geq 0$;*
- *существует такая матрица B, что $A = BB^\top$.*

C.4. Сингулярное разложение

Пусть $A \in \mathbb{R}^{m,n}$ – матрица ранга r. Если $m \neq n$, то спектральное разложение, описанное в теореме C.1 неприменимо. Мы опишем другое разложение A, которое называется сингулярным разложением, или коротко SVD (Singular Value Decomposition).

Единичные векторы $\mathbf{v} \in \mathbb{R}^n$ и $\mathbf{u} \in \mathbb{R}^m$ называются правым и левым *сингулярными векторами* A, соответствующими сингулярному значению $\sigma > 0$, если

$$A\mathbf{v} = \sigma\mathbf{u} \text{ и } A^\top\mathbf{u} = \sigma\mathbf{v}.$$

Сначала мы покажем, что если можно найти r ортонормированных сингулярных векторов с положительными сингулярными значениями, то можно представить A в виде $A = UDV^\top$, где столбцами U и V являются правые и левые сингулярные векторы соответственно, а D – диагональная матрица размера $r \times r$, на диагонали которой расположены сингулярные значения.

Лемма С.3. *Пусть $A \in \mathbb{R}^{m,n}$ – матрица ранга r. Предположим, что $\mathbf{v}_1, \ldots, \mathbf{v}_r$ – ортонормированное множество правых сингулярных векторов A, $\mathbf{u}_1, \ldots, \mathbf{u}_r$ – ортонормированное множество соответственных левых сингулярных векторов A, а $\sigma_1, \ldots, \sigma_r$ – соответствующие сингулярные значения. Тогда*

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top.$$

Отсюда следует, что если U – матрица, столбцами которой являются $\mathbf{u}_i$, V – матрица, столбцами которой являются $\mathbf{v}_i$, а D – диагональная матрица с $D_{i,i} = \sigma_i$, *то*

$$A = UDV^\top.$$

Доказательство. Любой правый сингулярный вектор A должен принадлежать образу A (в противном случае сингулярное значение было бы равно 0). Поэтому $\mathbf{v}_1, \ldots, \mathbf{v}_r$ образуют ортонормированный базис образа A. Дополним его до ортонормированного базиса $\mathbb{R}^n$, добавив векторы $\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n$. Определим $B = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$. Достаточно доказать, что для всех i имеет место $A\mathbf{v}_i = B\mathbf{v}_i$. Очевидно, что если $i > r$, то $A\mathbf{v}_i = 0$ и $B\mathbf{v}_i = 0$. Для $i \le r$ имеем

$$B\mathbf{v}_i = \sum_{j=1}^{r} \sigma_j \mathbf{u}_j \mathbf{v}_j^\top \mathbf{v}_i = \sigma_i \mathbf{u}_i = A\mathbf{v}_i,$$

где последнее равенство вытекает из определения. □

Следующая лемма устанавливает связь между сингулярными значениями A и собственными значениями $A^\top A$ и $AA^\top$.

Лемма С.4. *Векторы $\mathbf{v}$, $\mathbf{u}$ являются правым и левым сингулярными векторами A с сингулярным значением σ тогда и только тогда, когда $\mathbf{v}$ – собственный вектор $A^\top A$ с собственным значением σ^2, а $\mathbf{u} = \sigma^{-1} A\mathbf{v}$ – собственный вектор $AA^\top$ с собственным значением σ^2.*

Доказательство. Предположим, что σ – сингулярное значение A, а $\mathbf{v} \in \mathbb{R}^n$ – соответствующий ему правый сингулярный вектор. Тогда

$$A^\top A\mathbf{v} = \sigma A^\top \mathbf{u} = \sigma^2 \mathbf{v}.$$

Аналогично

$$AA^\top \mathbf{u} = \sigma A\mathbf{v} = \sigma^2 \mathbf{u}.$$

Обратно, если $\lambda \ne 0$ – собственное значение $A^\top A$, соответствующее собственному вектору $\mathbf{v}$, то $\lambda > 0$, потому что $A^\top A$ – положительно полуопределенная матрица. Положим $\sigma = \sqrt{\lambda}$, $\mathbf{u} = \sigma^{-1} A\mathbf{v}$. Тогда

$$\sigma\mathbf{u} = \sqrt{\lambda}\,\frac{A\mathbf{u}}{\sqrt{\lambda}} = A\mathbf{v}$$

и

$$A^{\mathsf{T}}\mathbf{u} = \frac{1}{\sigma}A^{\mathsf{T}}A\mathbf{u} = \frac{\lambda}{\sigma}\mathbf{v} = \sigma\mathbf{v}. \qquad \square$$

Наконец, покажем, что если A имеет ранг r, то у нее есть r ортонормированных сингулярных векторов.

Лемма С.5. *Пусть $A \in \mathbb{R}^{m,\,n}$ – матрица ранга r. Определим следующие векторы*:

$$\mathbf{v}_1 = \operatorname*{argmax}_{\mathbf{v}\in\mathbb{R}^n:\|\mathbf{v}\|=1} \|A\mathbf{v}\|$$

$$\mathbf{v}_2 = \operatorname*{argmax}_{\substack{\mathbf{v}\in\mathbb{R}^n:\|\mathbf{v}\|=1\\ \langle\mathbf{v},\mathbf{v}_1\rangle=0}} \|A\mathbf{v}\|$$

$$\vdots$$

$$\mathbf{v}_r = \operatorname*{argmax}_{\substack{\mathbf{v}\in\mathbb{R}^n:\|\mathbf{v}\|=1\\ \forall i<r,\ \langle\mathbf{v},\mathbf{v}_1\rangle=0}} \|A\mathbf{v}\|.$$

Тогда $\mathbf{v}_1, \ldots, \mathbf{v}_r$ – ортонормированное множество правых сингулярных векторов A.

Доказательство. Сначала заметим, что поскольку ранг A равен r, то образ A является подпространством размерности r, и потому легко проверить, что для всех $i = 1, \ldots, r \|A\mathbf{v}_i\| > 0$. Пусть $W \in \mathbb{R}^{n,n}$ – ортогональная матрица, полученная спектральным разложением $A^{\mathsf{T}}A$, т. е. $A^{\mathsf{T}}A = WDW^{\mathsf{T}}$, где D – диагональная матрица с $D_{1,1} \geq D_{2,2} \geq \ldots \geq 0$. Мы покажем, что $\mathbf{v}_1, \ldots, \mathbf{v}_r$ – собственные векторы $A^{\mathsf{T}}A$, соответствующие ненулевым собственным значениям, и, следовательно, из леммы С.4 следует, что они являются также правыми сингулярными векторами A. Доказательство проведем по индукции. В качестве основания индукции заметим, что любой единичный вектор $\mathbf{v}$ можно записать в виде $\mathbf{v} = W\mathbf{x}$, где $\mathbf{x} = W^{\mathsf{T}}\mathbf{v}$ и при этом $\|\mathbf{x}\| = 1$. Следовательно,

$$\|A\mathbf{v}\|^2 = \|AW\mathbf{x}\|^2 = \|WDW^{\mathsf{T}}W\mathbf{x}\|^2 = \|WD\mathbf{x}\|^2 = \|D\mathbf{x}\|^2 = \sum_{i=1}^{n} D_{i,i}^2 x_i^2.$$

Поэтому

$$\max_{\mathbf{v}:\|\mathbf{v}\|=1} \|A\mathbf{v}\|^2 = \max_{\mathbf{x}:\|\mathbf{x}\|=1} \sum_{i=1}^{n} D_{i,i}^2 x_i^2.$$

Максимум правой части достигается, когда $\mathbf{x} = (1, 0, \ldots, 0)$, откуда следует, что $\mathbf{v}_1$ – первый собственный вектор $A^{\mathsf{T}}A$. Поскольку $\|A\mathbf{v}_1\| > 0$, то $D_{1,1} > 0$, что нам и требуется. В качестве шага индукции предположим, что утверждение верно для некоторого $1 \leq t \leq r - 1$. Тогда любой вектор $\mathbf{v}$, ортогональный $\mathbf{v}_1, \ldots, \mathbf{v}_t$, можно записать в виде $\mathbf{v} = W\mathbf{x}$, где первые t элементов $\mathbf{x}$ равны нулю. Отсюда следует, что

$$\max_{\mathbf{v}:||\mathbf{v}||=1,\ \forall i\le t,\ \mathbf{v}^{\top}\mathbf{v}_i=0} \|A\mathbf{v}\|^2 = \max_{\mathbf{x}:||\mathbf{x}||=1} \sum_{i=t+1}^{n} D_{i,i}^2 x_i^2.$$

Максимум правой части достигается, когда все элементы вектора равны 0, кроме $x_{t+1} = 1$. Отсюда следует, что $\mathbf{v}_{t+1}$ – $(t+1)$-й столбец W. Наконец, поскольку $||A\mathbf{v}_{t+1}|| > 0$, то $D_{t+1,t+1} > 0$, что требуется. На этом доказательство завершается. □

Следствие С.6 (теорема о сингулярном разложении). *Пусть $A \in \mathbb{R}^{m,n}$ – матрица ранга r. Тогда $A = UDV^{\top}$, где D – матрица размера $r \times r$, содержащая ненулевые сингулярные значения A, а столбцы U и V – ортонормированные левые и правые сингулярные векторы A. Более того, для всех i $D_{i,i}^2$ – собственное значение $A^{\top}A$, i-й столбец V – соответствующий ему собственный вектор $A^{\top}A$, а i-й столбец U – соответствующий собственный вектор $AA^{\top}$.*

ЛАНЬ®

ЛАНЬ®

ЛИТЕРАТУРА

Abernethy, J., Bartlett, P. L., Rakhlin, A. & Tewari, A. (2008), «Optimal strategies and minimax lower bounds for online convex games», in *Proceedings of the nineteenthannual conference on computational learning theory*.

Ackerman, M. & Ben-David, S. (2008), «Measures of clustering quality: A working set of axioms for clustering», in *Proceedings of Neural Information Processing Systems* (NIPS), pp. 121–128.

Agarwal, S. & Roth, D. (2005), «Learnability of bipartite ranking functions», in *Proceedings of the 18th annual conference on learning theory*, pp. 16–31.

Agmon, S. (1954), «The relaxation method for linear inequalities», *Canadian Journal of Mathematics* **6** (3), 382–392.

Aizerman, M. A., Braverman, E. M. & Rozonoer, L. I. (1964), «Theoretical foundations of the potential function method in pattern recognition learning», *Automation and Remote Control* **25**, 821–837.

Allwein, E. L., Schapire, R. & Singer, Y. (2000), «Reducing multiclass to binary: A unifying approach for margin classifiers», *Journal of Machine Learning Research* **1**, 113–141.

Alon, N., Ben-David, S., Cesa-Bianchi, N. & Haussler, D. (1997), «Scale-sensitive dimensions, uniform convergence, and learnability», *Journal of the ACM* **44** (4), 615–631.

Anthony, M. & Bartlet, P. (1999), *Neural Network Learning: Theoretical Foundations*, Cambridge University Press.

Baraniuk, R., Davenport, M., DeVore, R. & Wakin, M. (2008), «A simple proof of the restricted isometry property for random matrices», *Constructive Approximation* **28** (3), 253–263.

Barber, D. (2012), *Bayesian reasoning and machine learning*, Cambridge University Press.

Bartlett, P., Bousquet, O. & Mendelson, S. (2005), «Local rademacher complexities», *Annals of Statistics* **33** (4), 1497–1537.

Bartlett, P. L. & Ben-David, S. (2002), «Hardness results for neural network approximation problems», *Theor. Comput. Sci.* **284** (1), 53–66.

Bartlett, P. L., Long, P. M. & Williamson, R. C. (1994), «Fat-shattering and the learnability of real-valued functions», in *Proceedings of the seventh annual conference oncomputational learning theory*, (ACM), pp. 299–310.

Bartlett, P. L. & Mendelson, S. (2001), «Rademacher and Gaussian complexities: Risk-bounds and structural results», in *14th Annual Conference on Computational Learning Theory* (COLT) *2001*, Vol. 2111, Springer, Berlin, pp. 224–240.

Bartlett, P. L. & Mendelson, S. (2002), «Rademacher and Gaussian complexities: Risk-bounds and structural results», *Journal of Machine Learning Research* **3**, 463–482.

Ben-David, S., Cesa-Bianchi, N., Haussler, D. & Long, P. (1995), «Characterizations of learnability for classes of {0, …, *n*}-valued functions», *Journal of Computer and System Sciences* **50**, 74–86.

Ben-David, S., Eiron, N. & Long, P. (2003), «On the difficulty of approximately maximizing agreements», *Journal of Computer and System Sciences* **66**(3), 496–514.

Ben-David, S. & Litman, A. (1998), «Combinatorial variability of Vapnik-Chervonenkis classes with applications to sample compression schemes», *Discrete Applied Mathematics* **86** (1), 3–25.

Ben-David, S., Pal, D., & Shalev-Shwartz, S. (2009), «Agnostic online learning», in Conference on Learning Theory (COLT).

Ben-David, S. & Simon, H. (2001), «Efficient learning of linear perceptrons», *Advancesin Neural Information Processing Systems*, pp. 189–195.

Bengio, Y. (2009), «Learning deep architectures for AI», *Foundations and Trends inMachine Learning* **2** (1), 1–127.

Bengio, Y. & LeCun, Y. (2007), «Scaling learning algorithms towards AI», *Large-Scale Kernel Machines* **34**.

Bertsekas, D. (1999), *Nonlinear programming*, Athena Scientific.

Beygelzimer, A., Langford, J. & Ravikumar, P. (2007), «Multiclass classification with filter trees», *Preprint*, *June*.

Birkhoff, G. (1946), «Three observations on linear algebra», *Revi. Univ. Nac. Tucuman*, *ser. A* **5**, 147–151.

Bishop, C. M. (2006), *Pattern recognition and machine learning*, Vol. 1, Springer: New York.

Blum, L., Shub, M. & Smale, S. (1989), «On a theory of computation and complexity over the real numbers: Np-completeness, recursive functions and universal machines», *Am. Math. Soc.* **21**(1), 1–46.

Blumer, A., Ehrenfeucht, A., Haussler, D. & Warmuth, M. K. (1987), «Occam′s razor», *Information Processing Letters* **24** (6), 377–380.

Blumer, A., Ehrenfeucht, A., Haussler, D. & Warmuth, M. K. (1989), «Learnability and the Vapnik-Chervonenkis dimension», *Journal of the Association for Computing Machinery* **36** (4), 929–965.

Borwein, J. & Lewis, A. (2006), *Convex analysis and nonlinear optimization*, Springer.

Boser, B. E., Guyon, I. M. & Vapnik, V. N. (1992), «A training algorithm for optimal margin classifiers», in COLT, pp. 144–152.

Bottou, L. & Bousquet, O. (2008), «The tradeoffs of large scale learning», in NIPS, pp. 161–168.

Boucheron, S., Bousquet, O. & Lugosi, G. (2005), «Theory of classification: A survey ofrecent advances», *ESAIM: Probability and Statistics* **9**, 323–375.

Bousquet, O. (2002), Concentration Inequalities and Empirical Processes Theory Applied to the Analysis of Learning Algorithms, PhD thesis, Ecole Polytechnique.

Bousquet, O. & Elisseeff, A. (2002), «Stability and generalization», *Journal of Machine-Learning Research* **2**, 499–526.

Boyd, S. & Vandenberghe, L. (2004), *Convex optimization*, Cambridge University Press.

Breiman, L. (1996), Bias, variance, and arcing classifiers, Technical Report 460, Statistics Department, University of California at Berkeley.

Breiman, L. (2001), «Random forests», *Machine Learning* **45** (1), 5–32.

Breiman, L., Friedman, J. H., Olshen, R. A. & Stone, C. J. (1984), *Classification andregression trees*, Wadsworth & Brooks.

Candès, E. (2008), «The restricted isometry property and its implications for compressed sensing», *Comptes Rendus Mathematique* **346** (9), 589–592.

Candes, E. J. (2006), «Compressive sampling», in *Proc. of the int. congress of math.*, Madrid, Spain.

Candes, E. & Tao, T. (2005), «Decoding by linear programming», *IEEE Trans. OnInformation Theory* **51**, 4203–4215.

Cesa-Bianchi, N. & Lugosi, G. (2006), *Prediction, learning, and games*, Cambridge University Press.

Chang, H. S., Weiss, Y. & Freeman, W. T. (2009), «Informative sensing», *arXiv preprint arXiv:0901.4275*.

Chapelle, O., Le, Q. & Smola, A. (2007), «Large margin optimization of ranking measures», in *NIPS workshop: Machine learning for Web search* (Machine Learning).

Collins, M. (2000), «Discriminative reranking for natural language parsing», in Machine Learning.

Collins, M. (2002), «Discriminative training methods for hidden Markov models: Theoryand experiments with perceptron algorithms», in *Conference on Empirical Methods in Natural Language Processing*.

Collobert, R. & Weston, J. (2008), «A unified architecture for natural language processing: deep neural networks with multitask learning», in International Conference on Machine Learning (ICML).

Cortes, C. & Vapnik, V. (1995), «Support-vector networks», *Machine Learning* **20** (3), 273–297.

Cover, T. (1965), «Behavior of sequential predictors of binary sequences», *Trans. 4th Prague conf. information theory statistical decision functions, random processes*, pp. 263–272.

Cover, T. & Hart, P. (1967), «Nearest neighbor pattern classification», *Information Theory, IEEE Transactions on* **13** (1), 21–27.

Crammer, K. & Singer, Y. (2001), «On the algorithmic implementation of multiclass kernel-based vector machines», *Journal of Machine Learning Research* **2**, 265–292.

Cristianini, N. & Shawe-Taylor, J. (2000), *An introduction to support vector machines*, Cambridge University Press.

Daniely, A., Sabato, S., Ben-David, S. & Shalev-Shwartz, S. (2011), «Multiclass learnability and the erm principle», in COLT.

Daniely, A., Sabato, S. & Shwartz, S. S. (2012), «Multiclass learning approaches: A theoretical comparison with implications», in NIPS.

Davis, G., Mallat, S. & Avellaneda, M. (1997), «Greedy adaptive approximation», *Journal of Constructive Approximation* **13**, 57–98.

Devroye, L. & Györfi, L. (1985), *Nonparametric density estimation: The L B1 S view*, Wiley.

Devroye, L., Györfi, L. & Lugosi,G. (1996), *A probabilistic theory of pattern recognition*, Springer.

Dietterich, T. G. & Bakiri, G. (1995), «Solving multiclass learning problems via error-correcting output codes», *Journal of Artificial Intelligence Research* **2**, 263–286.

Donoho, D. L. (2006), «Compressed sensing», *Information Theory, IEEE Transactions* **52** (4), 1289–1306.

Dudley, R., Gine, E. & Zinn, J. (1991), «Uniform and universal Glivenko-Cantelli classes», *Journal of Theoretical Probability* **4** (3), 485–510.

Dudley, R. M. (1987), «Universal Donsker classes and metric entropy», *Annals of Probability* **15** (4), 1306–1326.

Fisher, R. A. (1922), «On the mathematical foundations of theoretical statistics», *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* **222**, 309–368.

Floyd, S. (1989), «Space-bounded learning and the Vapnik-Chervonenkis dimension», in COLT, pp. 349–364.

Floyd, S. & Warmuth, M. (1995), «Sample compression, learnability, and the Vapnik-Chervonenkis dimension», *Machine Learning* **21** (3), 269–304.

Frank, M. & Wolfe, P. (1956), «An algorithm for quadratic programming», *Naval Res. Logist. Quart.* **3**, 95–110.

Freund, Y. & Schapire, R. (1995), «A decision-theoretic generalization of on-line learning and an application to boosting», in European Conference on Computational Learning Theory (EuroCOLT), Springer-Verlag, pp. 23–37.

Freund, Y. & Schapire, R. E. (1999), «Large margin classification using the perceptron algorithm», *Machine Learning* **37** (3), 277–296.

Garcia, J. & Koelling, R. (1996), «Relation of cue to consequence in avoidance learning», *Foundations of animal behavior: classic papers with commentaries* **4**, 374.

Gentile, C. (2003), «The robustness of the p-norm algorithms», *Machine Learning* **53** (3), 265–299.

Georghiades, A., Belhumeur, P. & Kriegman, D. (2001), «From few to many: Illumination cone models for face recognition under variable lighting and pose», *IEEE Trans. Pattern Anal. Mach. Intelligence* **23** (6), 643–660.

Gordon, G. (1999), «Regret bounds for prediction problems», in Conference on Learning Theory (COLT).

Gottlieb, L.-A., Kontorovich, L. & Krauthgamer, R. (2010), «Efficient classification formetric data», in *23rd conference on learning theory*, pp. 433–440.

Guyon, I. & Elisseeff, A. (2003), «An introduction to variable and feature selection», *Journal of Machine Learning Research*, *Special Issue on Variable and Feature Selection* **3**, 1157–1182.

Hadamard, J. (1902), «Sur les problèmes aux dérivées partielles et leur signification physique», *Princeton University Bulletin* **13**, 49–52.

Hastie, T., Tibshirani, R. & Friedman, J. (2001), *The elements of statistical learning*, Springer.

Haussler, D. (1992), «Decision theoretic generalizations of the PAC model for neuralnet and other learning applications», *Information and Computation* **100** (1), 78–150.

Haussler, D. & Long, P. M. (1995), «A generalization of Sauer's lemma», *Journal of Combinatorial Theory, Series A* **71** (2), 219–240.

Hazan, E., Agarwal, A. & Kale, S. (2007), «Logarithmic regret algorithms for online convex optimization», *Machine Learning* **69** (2–3), 169–192.

Hinton, G. E., Osindero, S. & Teh, Y.-W. (2006), «A fast learning algorithm for deep belief nets», *Neural Computation* **18** (7), 1527–1554.

Hiriart-Urruty, J.-B. & Lemaréchal, C. (1993), *Convex analysis and minimization algorithms*, Springer.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003), «A practical guide to support vector classification».

Hyafil, L. & Rivest, R. L. (1976), «Constructing optimal binary decision trees is NP complete», *Information Processing Letters* **5** (1), 15–17.

Joachims, T. (2005), «A support vector method for multivariate performance measures», in *Proceedings of the international conference on machine learning* (ICML).

Kakade, S., Sridharan, K. & Tewari, A. (2008), «On the complexity of linear prediction: Risk bounds, margin bounds, and regularization», in NIPS.

Karp, R. M. (1972), *Reducibility among combinatorial problems*, Springer.

Kearns, M. & Mansour, Y. (1996), «On the boosting ability of top-down decision tree learning algorithms», in ACM Symposium on the Theory of Computing (STOC).

Kearns, M. & Ron, D. (1999), «Algorithmic stability and sanity-check bounds for leave-one-out cross-validation», *Neural Computation* **11** (6), 1427–1453.

Kearns, M. & Valiant, L. G. (1988), «Learning Boolean formulae or finite automata is as hard as factoring, Technical Report TR-14-88, Harvard University, Aiken Computation Laboratory.

Kearns, M. & Vazirani, U. (1994), *An Introduction to Computational Learning Theory*, MIT Press.

Kearns, M. J., Schapire, R. E. & Sellie, L. M. (1994), «Toward efficient agnostic learning», *Machine Learning* **17**, 115–141.

Kleinberg, J. (2003), «An impossibility theorem for clustering», NIPS, pp. 463–470.

Klivans, A. R. & Sherstov, A. A. (2006), Cryptographic hardness for learning intersections of halfspaces, in FOCS.

Koller, D. & Friedman, N. (2009), *Probabilistic graphical models: Principles and techniques*, MIT Press.

Koltchinskii, V. & Panchenko, D. (2000), «Rademacher processes and bounding the risk of function learning», in *High Dimensional Probability II*, Springer, pp. 443–457.

Kuhn, H. W. (1955), «The hungarian method for the assignment problem», *Naval Research Logistics Quarterly* **2** (1–2), 83–97.

Kutin, S. & Niyogi, P. (2002), «Almost-everywhere algorithmic stability and generalization error», in *Proceedings of the 18th conference in uncertainty in artificial intelligence*, pp. 275–282.

Lafferty, J., McCallum, A. & Pereira, F. (2001), «Conditional random fields: Probabilistic models for segmenting and labeling sequence data», in *International conference onmachine learning*, pp. 282–289.

Langford, J. (2006), «Tutorial on practical prediction theory for classification», *Journal of machine learning research* **6** (1), 273.

Langford, J. & Shawe-Taylor, J. (2003), «PAC-Bayes & margins», in NIPS, pp. 423–430.

Le, Q. V., Ranzato,M.-A.,Monga, R., Devin,M., Corrado, G., Chen, K., Dean, J. & Ng, A. Y. (2012), «Building high-level features using large scale unsupervised learning», in ICML.

Le Cun, L. (2004), «Large scale online learning», in *Advances in neural information processing systems 16: Proceedings of the 2003 conference*, Vol. 16, MIT Press, p. 217.

Le Cun, Y. & Bengio, Y. (1995), «Convolutional networks for images, speech, and time-series», in *The handbook of brain theory and neural networks*, The MIT Press.

Lee, H., Grosse, R., Ranganath, R. & Ng, A. (2009), «Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations», in ICML.

Littlestone, N. (1988), «Learning quickly when irrelevant attributes abound: A newlinear-threshold algorithm», *Machine Learning* **2**, 285–318.

Littlestone, N. & Warmuth, M. (1986), Relating data compression and learnability. Unpublished manuscript.

Littlestone, N. & Warmuth, M. K. (1994), «The weighted majority algorithm», *Informationand Computation* **108**, 212–261.
Livni, R., Shalev-Shwartz, S. & Shamir, O. (2013), «A provably efficient algorithm for training deep networks», *arXiv preprint arXiv:1304.7045*.
Livni, R. & Simon, P. (2013), «Honest compressions and their application to compression schemes», in COLT.
MacKay, D. J. (2003), *Information theory, inference and learning algorithms*, Cambridge University Press.
Mallat, S. & Zhang, Z. (1993), «Matching pursuits with time-frequency dictionaries», *IEEE Transactions on Signal Processing* **41**, 3397–3415.
McAllester, D. A. (1998), «Some PAC-Bayesian theorems», in COLT.
McAllester, D. A. (1999), «PAC-Bayesian model averaging», in COLT, pp. 164–170.
McAllester, D. A. (2003), «Simplified PAC-Bayesian margin bounds», in COLT, pp. 203–215.
Minsky, M. & Papert, S. (1969), *Perceptrons: An introduction to computational geometry*, The MIT Press.
Mukherjee, S., Niyogi, P., Poggio, T. & Rifkin, R. (2006), «Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization», *Advances in Computational Mathematics* **25** (1–3), 161–193.
Murata, N. (1998), «A statistical study of on-line learning», *Online Learning and Neural Networks*, *Cambridge University Press*.
Murphy, K. P. (2012), *Machine learning: a probabilistic perspective*, The MIT Press.
Natarajan, B. (1995), «Sparse approximate solutions to linear systems», *SIAM J. Computing* **25** (2), 227–234.
Natarajan, B. K. (1989), «On learning sets and functions», *Mach. Learn.* **4**, 67–97.
Nemirovski, A., Juditsky, A., Lan, G. & Shapiro, A. (2009), «Robust stochastic approximation approach to stochastic programming», *SIAM Journal on Optimization* **19** (4), 1574–1609.
Nemirovski, A. & Yudin, D. (1978), *Problem complexity and method efficiency in optimization*, Nauka, Moscow.
Nesterov, Y. (2005), Primal-dual subgradient methods for convex problems, Technicalreport, Center for Operations Research and Econometrics (CORE), Catholic University of Louvain (UCL).
Nesterov, Y. & Nesterov, I. (2004), *Introductory lectures on convex optimization: A basic course*, Vol. 87, Springer, Netherlands.
Novikoff, A. B. J. (1962), «On convergence proofs on perceptrons», in *Proceedings of the symposium on the mathematical theory of automata*, Vol. XII, pp. 615–622.
Parberry, I. (1994), *Circuit complexity and neural networks*, The MIT Press.
Pearson, K. (1901), «On lines and planes of closest fit to systems of points in space», *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **2** (11), 559–572.
Phillips, D. L. (1962), «A technique for the numerical solution of certain integral equations of the first kind», *Journal of the ACM* **9** (1), 84–97.
Pisier, G. (1980–1981), «Remarques sur un résultat non publié de B. Maurey».
Pitt, L. & Valiant, L. (1988), «Computational limitations on learning from examples», *Journal of the Association for Computing Machinery* **35** (4), 965–984.

Poon, H. & Domingos, P. (2011), «Sum-product networks: A new deep architecture», in Conference on Uncertainty in Artificial Intelligence (UAI).

Quinlan, J. R. (1986), «Induction of decision trees», *Machine Learning* **1**, 81–106.

Quinlan, J. R. (1993), *C4.5: Programs for machine learning*, Morgan Kaufmann.

Rabiner, L. & Juang, B. (1986), «An introduction to hidden markov models», *IEEE ASSP Magazine* **3** (1), 4–16.

Rakhlin, A., Shamir, O. & Sridharan, K. (2012), «Making gradient descent optimal for strongly convex stochastic optimization», in ICML.

Rakhlin, A., Sridharan, K. & Tewari, A. (2010), «Online learning: Random averages, combinatorial parameters, and learnability», in NIPS.

Rakhlin, S., Mukherjee, S. & Poggio, T. (2005), «Stability results in learning theory», *Analysis and Applications* **3** (4), 397–419.

Ranzato, M., Huang, F., Boureau, Y. & Lecun, Y. (2007), «Unsupervised learning of invariant feature hierarchies with applications to object recognition», in Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE, pp. 1–8.

Rissanen, J. (1978), «Modeling by shortest data description», *Automatica* **14**, 465–471.

Rissanen, J. (1983), «A universal prior for integers and estimation by minimum description length», *The Annals of Statistics* **11** (2), 416–431.

Robbins, H. & Monro, S. (1951), «A stochastic approximation method», *The Annals of Mathematical Statistics*, pp. 400–407.

Rogers, W. & Wagner, T. (1978), «A finite sample distribution-free performance bound for local discrimination rules», *The Annals of Statistics* **6** (3), 506–514.

Rokach, L. (2007), *Data mining with decision trees: Theory and applications*, Vol. 69, World Scientific.

Rosenblatt, F. (1958), «The perceptron: A probabilistic model for information storageand organization in the brain», *Psychological Review* **65**, 386–407. (Reprinted in *Neurocomputing*, MIT Press, 1988).

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986), «Learning internal representations by error propagation», in D. E. Rumelhart & J. L. McClelland, eds, *Parallel distributed processing – explorations in the microstructure of cognition*, MIT Press, chapter 8, pp. 318–362.

Sankaran, J. K. (1993), «A note on resolving infeasibility in linear programs by constraint relaxation», *Operations Research Letters* **13** (1), 19–20.

Sauer, N. (1972), «On the density of families of sets», *Journal of Combinatorial Theory Series A* **13**, 145–147.

Schapire, R. (1990), «The strength of weak learnability», *Machine Learning* **5** (2),197–227.

Schapire, R. E. & Freund, Y. (2012), *Boosting: Foundations and algorithms*, MIT Press.

Schölkopf, B. & Smola, A. J. (2002), *Learning with kernels: Support vector machines, regularization, optimization and beyond*, MIT Press.

Seeger, M. (2003), «Pac-bayesian generalisation error bounds for gaussian process classification», *The Journal of Machine Learning Research* **3**, 233–269.

Shakhnarovich, G., Darrell, T. & Indyk, P. (2006), *Nearest-neighbor methods in learningand vision: Theory and practice*, MIT Press.

Shalev-Shwartz, S. (2007), Online Learning: Theory, Algorithms, and Applications, PhD thesis, The Hebrew University.

Shalev-Shwartz, S. (2011), «Online learning and online convex optimization», *Foundations and Trendsin Machine Learning* **4** (2), 107–194.

Shalev-Shwartz, S., Shamir, O., Srebro, N. & Sridharan, K. (2010), «Learnability, stability and uniform convergence», *The Journal of Machine Learning Research* **9999**, 2635–2670.

Shalev-Shwartz, S., Shamir, O. & Sridharan, K. (2010), «Learning kernel-based halfspaces with the zero-one loss», in COLT.

Shalev-Shwartz, S., Shamir, O., Sridharan, K. & Srebro, N. (2009), «Stochastic convex optimization», in COLT.

Shalev-Shwartz, S. & Singer, Y. (2008), «On the equivalence of weak learnability and linear separability: New relaxations and efficient boosting algorithms», in *Proceedings of the nineteenth annual conference on computational learning theory*.

Shalev-Shwartz, S., Singer, Y. & Srebro, N. (2007), «Pegasos: Primal Estimated subGrAdient SOlver for SVM», in *International conference on machine learning*, pp. 807–814.

Shalev-Shwartz, S. & Srebro, N. (2008), «SVM optimization: Inverse dependence on training set size», in *International conference on machine learning* ICML, pp. 928–935.

Shalev-Shwartz, S., Zhang, T. & Srebro, N. (2010), «Trading accuracy for sparsity in optimization problems with sparsity constraints», *Siam Journal on Optimization* **20**, 2807–2832.

Shamir, O. & Zhang, T. (2013), «Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes», in ICML.

Shapiro, A., Dentcheva, D. & Ruszczyński, A. (2009), *Lectures on stochastic programming: modeling and theory*, Vol. 9, Society for Industrial and Applied Mathematics.

Shelah, S. (1972), «A combinatorial problem; stability and order formodels and theories in infinitary languages», *Pac. J. Math* **4**, 247–261.

Sipser, M. (2006), *Introduction to the Theory of Computation*, Thomson Course Technology.

Slud, E. V. (1977), «Distribution inequalities for the binomial law», *The Annals of Probability* **5** (3), 404–412.

Steinwart, I. & Christmann, A. (2008), *Support vector machines*, Springerverlag, New York.

Stone, C. (1977), «Consistent nonparametric regression», *The Annals of Statistics* **5** (4), 595–620.

Taskar, B., Guestrin, C. & Koller, D. (2003), «Max-margin markov networks», in NIPS.

Tibshirani, R. (1996), «Regression shrinkage and selection via the lasso», *J. Royal. Statist. Soc B.* **58** (1), 267–288.

Tikhonov, A. N. (1943), «On the stability of inverse problems», *Dolk. Akad. Nauk SSSR* **39** (5), 195–198.

Tishby, N., Pereira, F. & Bialek, W. (1999), «The information bottleneck method», in *The 37′th Allerton conference on communication, control, and computing*.

Tsochantaridis, I., Hofmann, T., Joachims, T. & Altun, Y. (2004), «Support vectormachine learning for interdependent and structured output spaces», in *Proceedings of the twenty-first international conference on machine learning*.

Valiant, L. G. (1984), «A theory of the learnable», *Communications of the ACM* **27** (11), 1134–1142.

Vapnik, V. (1992), «Principles of risk minimization for learning theory», in J. E. Moody, S. J. Hanson & R. P. Lippmann, eds., *Advances in Neural Information Processing Systems 4*, Morgan Kaufmann, pp. 831–838.

Vapnik, V. (1995), *The Nature of Statistical Learning Theory*, Springer.

Vapnik, V. N. (1982), *Estimation of Dependences Based on Empirical Data*, Springer-Verlag.

Vapnik, V. N. (1998), *Statistical Learning Theory*, Wiley.

Vapnik, V. N. & Chervonenkis, A. Y. (1971), «On the uniform convergence of relative frequencies of events to their probabilities», *Theory of Probability and Its Applications* **XVI** (2), 264–280.

Vapnik, V. N. & Chervonenkis, A. Y. (1974), *Theory of pattern recognition*, Nauka, Moscow (In Russian).

von Luxburg, U. (2007), «A tutorial on spectral clustering», *Statistics and Computing* **17** (4), 395–416.

von Neumann, J. (1928), «Zur theorie der gesellschaftsspiele (on the theory of parlor-games)», *Math. Ann.* **100**, 295–320.

von Neumann, J. (1953), «A certain zero-sum two-person game equivalent to the optimal assignment problem», *Contributions to the Theory of Games* **2**, 5–12.

Vovk, V. G. (1990), «Aggregating strategies», in COLT, pp. 371–383.

Warmuth, M., Glocer, K. & Vishwanathan, S. (2008), «Entropy regularized lpboost», in *Algorithmic Learning Theory* (ALT).

Warmuth, M., Liao, J. & Ratsch, G. (2006), «Totally corrective boosting algorithms thatmaximize the margin», in *Proceedings of the 23rd international conference on machinelearning*.

Weston, J. & Watkins, C. (1999), «Support vector machines for multi-class pattern recognition», in *Proceedings of the seventh european symposium on artificial neural-networks*.

Wolpert, D. H. & Macready, W. G. (1997), «No free lunch theorems for optimization», *Evolutionary Computation*, *IEEE Transactions on* **1** (1), 67–82.

Zhang, T. (2004), «Solving large scale linear prediction problems using stochastic gradient descent algorithms», in *Proceedings of the twenty-first international conference on machine learning*.

Zhao, P. & Yu, B. (2006), «On model selection consistency of Lasso», *Journal of Machine Learning Research* **7**, 2541–2567.

Zinkevich, M. (2003), «Online convex programming and generalized in finitesimal gradient ascent», in *International conference on machine learning*.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

Символ

ℓ_1 норма, 175, 320, 351, 373

A

AdaBoost, 124, 128, 351

C

C4.5, 246
CART, 246
Consistent алгоритм, 278

E

EM-алгоритм, 336
ERM, 31

F

F1-мера, 236

H

Halving алгоритм, 279

I

ID3, 244

K

k-медианы, 301
k-медоиды, 301
k-средних алгоритм, 300
k-средних алгоритм, 302
  мягкий, 340

L

LASSO, 353
Ldim, 279, 280

M

MDL. *См.* Минимальная длина описания

P

PAC-байесовский подход, 402
PAC-обучение, 39
  агностическое, 41, 45
PCA. *См.* Метод главных компонент

R

RLM. *См.* Минимизация регуляризированной потери

S

SRM. *См.* Структурная минимизация риска
SVD. *См.* Сингулярное разложение
SVM. *См.* Метод опорных векторов

T

TF-IDF, 223

V

VC-размерность, 64, 66

W

Weighted-Majority алгоритм, 285

А

Автокодировщики, 356
Априорное знание, 59

Б

Базовая гипотеза, 131
Байеса формула, 342
Байесовское рассуждение, 341
Беннета неравенство, 413
Бернштейна неравенство, 414
Беспрефиксный язык, 86
Булевы конъюнкции, 47, 75, 102

В

Вапника-Червоненкиса размерность. *См.* VC-размерность
Верность, 35, 39
Вероятно почти корректное обучение. *См.* PAC-обучение
Виолы-Джонса гипотеза, 134
Все пары, 221, 391
Выбор модели, 138, 141
Выборочная сложность, 40
Выигрыш, 245
Выпуклое-гладкое-ограниченное обучение, 159

Выпуклое-липшицево-ограниченное обучение, 158
Выпуклость, 149
  выпуклая функция, 150
  выпуклое множество, 149
  строго выпуклая функция, 167, 187
Вычислительная сложность, 96

Г

Гильбертово пространство, 208
Гипотеза, 30
Гладкость, 154, 169, 190
Гливенко-Кантелли классы, 54
Градиент, 151
Градиентный спуск, 177
Грама матрица, 211
Граница объединения, 35
Граница ошибок, 277
Границы сжатия, 397
Гребневая регрессия, 164
  ядерная, 217

Д

Дадли классы, 77
Дважды стохастическая матрица, 234
Двойственность, 203
  сильная, 203
  слабая, 203
Дендрограмма, 298, 299
Джини индекс, 246
Джонсона-Линденштрауса лемма, 318
Дискретизация, 53
Дискриминантный подход, 330

З

Зависящее от класса отображение в пространство признаков, 222
Зазор, 195
Зарезервированный набор, 140
Зауэра лемма, 70

И

Индуктивное смещение. *См*. Смещение
Интегральное изображение, 137
Информационное горлышко, 306
Информационный выигрыш, 246
Истинная ошибка, 31, 41

К

Кендалла тау, 231
Класс гипотез, 33
Классификатор, 30
Кластеризация, 296
  спектральная, 303
Компромисс между смещением и сложностью, 61
Контроль, 139, 140
  обучение-контроль-тестирование, 144
  перекрестная проверка, 143
Концентрация меры, 52, 409
Кривые обучения, 146

Л

Лемма о сжатии, 368
Линейная регрессия, 117
Линейное программирование, 113
Линейный дискриминантный анализ (ЛДА), 335
Линейный предиктор, 111
  однородный, 112
Липшицевость, 153, 168, 183
  субградиент, 181
Литлстоуна размерность. *См*. Ldim
Логистическая регрессия, 120
Локальный минимум, 151

М

Макдайрмида неравенство, 365
Максимальная связь, 299
Маркова неравенство, 409
Массарта лемма, 367
Матрица перестановок, 234
Метка, 29
Метод ближайших соседей, 250
  k-NN алгоритм, 251
Метод главных компонент, 313
Метод наименьших квадратов, 118
Метод опорных векторов, 194, 371
  Hard-SVM, 196
  Soft-SVM, 198
  граница обобщаемости, 200, 371
  двойственность, 203
  однородный, 197
  опорные векторы, 202
  ядерный трюк, 209
Мешок слов, 201
Минимальная длина описания, 85, 86, 243
Минимизация регуляризированной потери, 163, 191
Минимизация эмпирического риска. *См*. ERM
Многоклассовая классификация, 43, 219, 389
  SVM, 226

линейные предикторы, 222, 392
многовекторное построение, 223, 393
перцептрон, 241
сведение, 220, 392
СГС, 227
стоимостная, 224
Многомерные показатели качества, 235

Н

Надграфик, 150
Наивная байесовская классификация, 335
Натараджана размерность, 389
Недообучение, 61, 145
Независимость от представления, 45, 103
Независимость от распределения, 330
Нейронные сети, 260
прямого распространения, 261
СГС, 268
слоистые, 261
Неравенство оракула, 171
Неравномерное обучение, 80
Нормированная приведенная суммарная эффективность (NDGG), 231
Нормировка признаков, 353
Нормы, индуцирующие разреженность, 351

О

Область образцов, 29
Образец, 29
Обратное исключение, 351
Обратное распространение, 269
Обучающий набор, 30
Обучение признаков, 356
Обучение словаря, 356
Обучения без учителя, 297
Ограниченность, 158
Одиночная связи, 299
Один против всех, 220
Один против остальных, 220
Оккама бритва, 87
Онлайновая выпуклая оптимизация, 288
Онлайновое обучение, 276
Онлайновый градиентный спуск, 289
Оптимальный байесовский предиктор, 42, 48, 252
Ортогональное согласованное преследование, 348
Отбор признаков, 345, 346
Отношение правдоподобия, 336
Оценка апостериорного максимума, 343
Оценка максимального правдоподобия, 331
Ошибка
аппроксимации, 57
обобщения, 31
обучения, 31
оптимизации, 161
оценивания, 57

П

Параметрическое оценивание плотности распределения, 330
Переобучение, 31, 61, 145
Перцептрон, 114
многоклассовый, 241
онлайновый, 290
ядерный, 217
Пирсона коэффициент корреляции, 347
Полиномиальная регрессия, 119
Полнота, 236
Полупространство, 112
неразделимый случай, 113
однородное, 113, 197
разделимый случай, 112
Понижение размерности, 312
Порождающие модели, 330
Потеря, 31
Практически осуществимый, 96
Предиктор, 30
Предсказание структурированного выхода, 228
Преобразования признаков, 355
Проецирование, 185
Проклятие размерности, 255
Пространство версий, 278
Пространство признаков, 208
Прямой жадный отбор, 349

Р

Равномерная сходимость, 51
Радемахеровская сложность, 362
Разбиение, 65
Разложение ошибки, 60, 160
Ранжирование, 230
двудольное, 235
Распознавание лиц. *См.* Виолы-Джонса гипотеза
Реализуемость, 34
Регрессия, 43, 117, 164
Регуляризация, 163
Тихонова, 164, 166
Редукция решающего дерева, 246
Репрезентативная выборка, 50, 362
Решающие деревья, 242
Решающие пни, 126, 127
Риск, 31, 41, 44

С

Самоограниченность, 155
Свойство ограниченной изопериметрии (RIP), 320
Связи, 299
СГС. *См.* Стохастический градиентный спуск
Сжатое измерение сигнала, 319
Сильное обучение, 126
Сингулярное разложение, 419
Скрытые переменные, 336
Скрытые слои, 262
Слабое обучение, 124, 125
Слада неравенство, 415
Случайные леса, 247
Случайные проекции, 317
Смесь нормальных распределений, 336
Смещение, 33, 57, 60
Собственное обучение, 45
Спектральная кластеризация, 303
Специфичность, 236
Средняя связь, 299
Стандартный оптимальный алгоритм (СОА), 281
Стохастический градиентный спуск, 183
Структурная минимизация риска, 81, 139
Субградиент, 180
Субдифференциал, 180
Схема сжатия, 398
Сцепление, 376

Т

Теорема об отсутствии бесплатных завтраков, 57
Теорема о представителе, 210
Точность, 236
Трехчленная ДНФ, 103

У

Уверенность, 34, 39
Усиление, 124
Усиление уверенности, 136
Устойчивость, 166

Ф

Фильтры, 347
Функция активации, 261
Функция потерь, 44
  бинарная, 44, 160
  выпуклая, 156
  гладкая, 159
  квадратичная, 44
  кусочно-линейная, 160
  липшицева, 158
  логарифмическая, 333
  логистическая, 121
  обобщенная кусочно-линейная, 225
  по абсолютной величине, 118, 122, 159
  рамповая, 201
  суррогатная, 159, 291
Функция роста, 70

Х

Хёфдинга неравенство, 52

Ц

Целевое множество, 43

Ч

Частота терма, 223
Частотный подход, 341
Чебышева неравенство, 410
Чернова границы, 411
Числа покрытия, 375
Чувствительность, 236

Э

Эмпирическая ошибка, 31
Эмпирический риск, 31, 44
Энтропия, 333
  относительная, 333
Эффективно вычислимый, 96

Я

Ядерный PCA, 315
Ядра, 207
  гауссово, 212
  полиномиальное, 212
  радиально-базисное (RBF), 213
  ядерный трюк, 209

Книги издательства «ДМК Пресс» можно заказать
в торгово-издательском холдинге «Планета Альянс» наложенным платежом,
выслав открытку или письмо по почтовому адресу:
115487, г. Москва, 2-й Нагатинский пр-д, д. 6А.
При оформлении заказа следует указать адрес (полностью), по которому должны быть
высланы книги; фамилию, имя и отчество получателя.
Желательно также указать свой телефон и электронный адрес.
Эти книги вы можете заказать и в интернет-магазине: **www.alians-kniga.ru**.
Оптовые закупки: тел. **(499) 782-38-89**.
Электронный адрес: **books@alians-kniga.ru**.

Шай Шалев-Шварц, Шай Бен-Давид

Идеи машинного обучения

От теории к алгоритмам

Главный редактор *Мовчан Д. А.*
dmkpress@gmail.com
Перевод *Слинкин А. А.*
Корректор *Светлова О. Ю.*
Верстка *Чаннова А. А.*
Дизайн обложки *Мовчан А. Г.*

Формат 70×100 1/16 .
Гарнитура «PT Serif». Печать офсетная.
Усл. печ. л. 35,43. Тираж 200 экз.

Веб-сайт издательства: www.dmkpress.com