

Содержание

ТОМ 1. ОСНОВЫ

Предисловие. Добро пожаловать!.....12

Глава 1. Введение в машинное обучение и глубокое обучение19

1.1. Зачем здесь эта глава	19
1.1.1. Извлечение значащей информации из данных	20
1.1.2. Экспертные системы	22
1.2. Изучение маркированных данных	23
1.2.1. Стратегия обучения	25
1.2.2. Стратегия компьютерного обучения	26
1.2.3. Обобщение	28
1.2.4. Более внимательный взгляд на обучение	30
1.3. Наблюдаемое обучение	31
1.3.1. Классификация	32
1.3.2. Регрессия	33
1.4. Ненаблюдаемое обучение	34
1.4.1. Кластеризация	35
1.4.2. Подавление шума	35
1.4.3. Уменьшение размерности	36
1.5. Генераторы	40
1.6. Усиленное обучение	41
1.7. Глубокое обучение	43
1.8. Что последует дальше	47
Справочные материалы	47
Задокументированные рисунки	48

Глава 2. Хаотичность и базовая статистика.....49

2.1. Зачем здесь эта глава	49
2.2. Случайные переменные	50
2.2.1. Случайные числа на практике	55
2.3. Некоторые общеупотребительные распределения	57
2.3.1. Равномерное распределение	57
2.3.2. Нормальное распределение	58
2.3.3. Распределение Бернулли	62
2.3.4. Мультиномиальное распределение	63
2.3.5. Ожидаемые величины	64
2.4. Зависимость	64
2.4.1. Независимые, одинаково распределенные переменные	65
2.5. Выбор и возврат	65
2.5.1. Выбор с возвращением	65
2.5.2. Выбор без возвращения	66
2.5.3. Осуществление выбора	67
2.6. Оценка ошибки обобщения	67
2.7. Многомерные пространства	71
2.8. Ковариация и корреляция	74
2.8.1. Ковариация	74

2.8.2. Корреляция	76
2.9. Квартет Энскомба	79
Справочные материалы	80
Глава 3. Вероятность.....	82
3.1. Почему здесь эта глава	82
3.2. Броски дротика	83
3.3. Простая вероятность.....	85
3.4. Условная вероятность	85
3.5. Совместные распределения	89
3.6. Маргинальная вероятность	92
3.7. Корректность измерений.....	93
3.7.1. Классификация выборок	93
3.7.2. Матрица ошибок	96
3.7.3. Интерпретация матрицы ошибок	98
3.7.4. Когда неправильная классификация хороша	100
3.7.5. Точность.....	102
3.7.6. Достоверность	103
3.7.7. Чувствительность	104
3.7.8. О достоверности и чувствительности	105
3.7.9. Другие критерии	108
3.7.10. Совместное использование достоверности и чувствительности	109
3.7.11. f1 счет	112
3.8. Применение матрицы ошибок	113
Справочные материалы	118
Глава 4. Правило Байеса.....	119
4.1. Почему здесь эта глава	119
4.2. Частотная и байесовская вероятности	120
4.2.1. Частотный подход.....	120
4.2.2. Байесовский подход	120
4.2.3. Обсуждение.....	121
4.3. Подбрасывание монеты.....	122
4.4. Это несмешенная монета?	122
4.4.1. Правило Байеса	131
4.4.2. Замечания по поводу правила Байеса.....	133
4.5. Поиски жизни.....	135
4.6. Повторные применения правила Байеса	139
4.6.1. Цикл постериор–приор.....	139
4.6.2. Пример: какую монету мы имеем?	141
4.7. Множественные гипотезы	146
Справочные материалы	153
Глава 5. Кривые и поверхности	154
5.1. Почему здесь эта глава	154
5.2. Введение	154
5.3. Производная.....	156
5.4. Градиент	164
Справочные материалы	169

Глава 6. Теория информации.....	170
6.1. Почему эта глава здесь	170
6.1.1. Информация: одно слово, два значения.....	170
6.2. Удивление и контекст	170
6.2.1. Удивление.....	171
6.2.2. Контекст	172
6.3. Бит как единица информации	173
6.4. Измерение информации	173
6.5. Размер события	175
6.6. Адаптивные коды.....	175
6.7. Энтропия.....	181
6.8. Кросс-энтропия.....	183
6.8.1. Два адаптивных кода.....	184
6.8.2. Смешение кодов	186
6.9. Расходимость.....	188
Справочные материалы	190
Глава 7. Классификация	192
7.1. Почему здесь эта глава.....	192
7.2. Двумерная классификация	193
7.2.1. 2D-бинарная классификация	193
7.3. 2D-классификация со многими классами	197
7.4. Бинарная категоризация со многими классами	199
7.4.1. Один против остальных.....	199
7.4.2. Один против одного.....	201
7.5. Кластеризация	204
7.6. Проклятие размерности.....	207
7.6.1. Особенности большой размерности.....	214
Справочные материалы	219
Глава 8. Обучение и тестирование	221
8.1. Почему здесь эта глава	221
8.2. Обучение.....	222
8.2.1. Тестирование характеристик.....	223
8.3. Тестовые данные	226
8.4. Проверочные данные.....	229
8.5. Кросс-валидация	232
8.5.1. k -кратная валидация	234
8.6. Использование результатов тестирования	237
Справочные материалы	237
Зaimствованные изображения	238
Глава 9. Избыточное обучение и недостаточное обучение	239
9.1. Почему здесь эта глава	239
9.2. Избыточность и недостаточность	240
9.2.1. Избыточность.....	240
9.2.2. Недостаточность	241
9.3. Избыточные данные	241
9.4. Преждевременная остановка	245
9.5. Регуляризация	247

9.6. Смещение и дисперсия.....	248
9.6.1. Аппроксимация базовых данных	249
9.6.2. Большое смещение, маленькая дисперсия.....	251
9.6.3. Маленькое смещение, большая дисперсия.....	252
9.6.4. Сравнение кривых.....	254
9.7. Аппроксимация по правилу Байеса.....	256
Справочные материалы	262
Глава 10. Нейроны	263
10.1. Почему здесь эта глава.....	263
10.2. Реальные нейроны	263
10.3. Искусственные нейроны.....	265
10.3.1. Персептрон.....	265
10.3.2. История персептрана	267
10.3.3. Современные искусственные нейроны	268
10.4. Заключение.....	272
Справочные материалы	273
Глава 11. Обучение и мышление.....	275
11.1. Почему здесь эта глава.....	275
11.2. Ступени обучения	275
11.2.1. Представление	276
11.2.2. Оценка	278
11.2.3. Оптимизация	279
11.3. Дедукция и индукция	280
11.4. Дедукция.....	281
11.4.1. Ошибки категориальных силлогизмов	285
11.5. Индукция	287
11.5.1. Термины индукции в машинном обучении	290
11.5.2. Ошибки индукции	290
11.6. Объединенные умозаключения	292
11.6.1. Шерлок Холмс, «мастер дедукции».....	293
11.7. Стимуляция адаптивного поведения.....	294
Справочные материалы	296
Глава 12. Подготовка данных	298
12.1. Почему здесь эта глава.....	298
12.2. Преобразование данных.....	298
12.3. Типы данных	301
12.3.1. Кодирование с использованием индивидуальных переменных	302
12.4. Базовая очистка данных	304
12.4.1. Очистка данных	304
12.4.2. Очистка данных на практике.....	305
12.5. Нормализация и стандартизация	305
12.5.1. Нормализация.....	306
12.5.2. Стандартизация	307
12.5.3. Запоминание преобразований	308
12.5.4. Типы преобразований	309
12.6. Выбор характеристик.....	310
12.7. Уменьшение размерности	311
12.7.1. Анализ главных компонентов (PCA).....	311

12.7.2. Стандартизация и РСА для изображений.....	317
12.8. Преобразования	324
12.9. Фрагментарная обработка.....	329
12.9.1. Обработка по выборкам	330
12.9.2. Обработка по характеристикам.....	331
12.9.3. Обработка по элементам.....	331
12.10. Преобразование с кросс-валидацией	332
Справочные материалы	336
Заемствованные изображения	337
Глава 13. Классификаторы	338
13.1. Почему здесь эта глава	338
13.2. Типы классификаторов.....	339
13.3. Алгоритм k -ближайших соседей (KNN)	340
13.4. Метод опорных векторов (SVM).....	346
13.5. Деревья решений	352
13.5.1. Построение деревьев.....	357
13.5.2. Разделение узлов	361
13.5.3. Контроль избыточного обучения	363
13.6. Наивный Байес	363
13.7. Обсуждение.....	368
Справочные материалы	370
Глава 14. Ансамбли	371
14.1. Почему здесь эта глава.....	371
14.2. Ансамбли	371
14.3. Голосование	372
14.4. Бутстреп-агрегация.....	373
14.5. Случайные леса	375
14.6. Экстрап-деревья	376
14.7. Бустинг	377
Справочные материалы	384
Глава 15. Библиотека Scikit-learn	385
15.1. Почему здесь эта глава.....	385
15.2. Введение	386
15.3. Конвенции Python.....	387
15.4. Оценщик	390
15.4.1. Создание объекта.....	391
15.4.2. Обучение с <code>fit()</code>	392
15.4.3. Предсказание с <code>predict()</code>	393
15.4.4. Функции <code>decision_function()</code> , <code>predict_proba()</code>	395
15.5. Кластеризация.....	395
15.6. Преобразования	398
15.6.1. Инверсные преобразования	402
15.7. Очистка данных.....	405
15.8. Ансамбли	407
15.9. Автоматизация	410
15.9.1. Кросс-валидация.....	410
15.9.2. Поиск гиперпараметров.....	413
15.9.3. Исчерпывающая решетка поиска.....	416

15.9.4. Случайная поисковая решетка	424
15.9.5. Конвейер.....	424
15.9.6. Граница решения.....	433
15.9.7. Конвейерные преобразования	434
15.10. Множества данных	435
15.11. Утилиты.....	437
15.12. Завершение.....	440
Справочные материалы	440
Глава 16. Нейронные сети прямого распространения.....	442
16.1. Почему здесь эта глава.....	442
16.2. Графики нейронных сетей.....	443
16.3. Синхронные и асинхронные потоки	445
16.3.1. Графы в практике.....	446
16.4. Инициализация весов	447
16.4.1. Инициализация	448
Справочные материалы	450
Глава 17. Функции активации	452
17.1. Почему здесь эта глава	452
17.2. Что делает функция активации	452
17.2.1. Формы функций активации	456
17.3. Основные функции активации.....	456
17.3.1. Линейные функции	456
17.3.2. Лестничная функция	457
17.4. Ступенчатые функции	458
17.5. Кусочно-линейные функции.....	460
17.6. Гладкие функции	463
17.7. Галерея функций активации	467
17.8. Софтмакс	468
Справочные материалы	470
Глава 18. Обратное распространение	471
18.1. Почему здесь эта глава.....	471
18.1.1. О тонкости настройки	472
18.2. Очень медленный способ обучения	473
18.2.1. Медленный способ обучения	475
18.2.2. Более быстрый способ обучения	477
18.3. Без функций активации.....	479
18.4. Выходы нейронов и ошибка сети.....	479
18.4.1. Пропорциональность изменения ошибок	480
18.5. Маленькая нейронная сеть	483
18.6. Шаг 1: дельты для выходных нейронов	487
18.7. Шаг 2: использование дельт для изменения весов.....	496
18.8. Шаг 3: дельты других нейронов	499
18.9. Обратное распространение в действии.....	504
18.10. Использование функций активации.....	508
18.11. Скорость обучения	514
18.11.1. Исследование скорости обучения.....	515
18.12. Обсуждение	523
18.12.1. Обратное распространение одной диаграммой.....	523

18.12.2. Что обратное распространение не делает.....	524
18.12.3. Что обратное распространение делает	524
18.12.4. Поддержка нейрона	525
18.12.5. Мини-пакеты	528
18.12.6. Параллельное обновление	528
18.12.7. Чем привлекательно обратное распространение	529
18.12.8. Обратное распространение не гарантировано.....	529
18.12.9. Немного истории	530
18.12.10. Погружение в математику.....	530
Справочные материалы	532
Глава 19. Оптимизаторы.....	534
19.1. Почему здесь эта глава.....	534
19.2. Геометрия ошибки	534
19.2.1. Минимумы, максимумы, плато и седла	534
19.2.2. Двумерная кривая ошибки	538
19.3. Настройка скорости обучения.....	541
19.3.1. Обновления постоянного размера	541
19.3.2. Изменение скорости обучения в процессе обучения.....	548
19.3.3. Расписание затуханий.....	551
19.4. Стратегии обновления	553
19.4.1. Пакетный градиентный спуск	554
19.4.2. Стохастический градиентный спуск	556
19.4.3. Мини-пакетный градиентный спуск	558
19.5. Варианты градиентного спуска	560
19.5.1. Метод импульса	561
19.5.2. Импульс Нестерова	567
19.5.3. Adagrad (Адаптивное градиентное обучение, Адаград)	571
19.5.4. Ададельта и Среднеквадратичное распространение (RMSprop)	572
19.5.5. Адам	574
19.6. Выбор оптимизатора	575
Справочные материалы	576

ТОМ 2. ПРАКТИКА

Глава 20. Глубокое обучение

Глава 21. Нейронные сети сверток

Глава 22. Рекуррентные нейронные сети

Глава 23. Keras, часть1.

Глава 24. Keras, часть 2

Глава 25. Автоматическое сжатие выборки

Глава 26. Стимулированное обучение

Глава 27. Порождающие состязательные сети

Глава 28. Применения для творчества

Глава 29. Множества данных

Глава 30. Глоссарий

Предисловие

Добро пожаловать!

*Несколько слов введения в эту книгу,
как получить файлы и рисунки,
и благодарности тем, кто помогал мне*

Что вы получите от этой книги

Привет!

Если вы интересуетесь глубоким обучением (Deep Learning, DL) и машинным обучением (Machine Learning, ML), то эта книга для вас.

Моя цель в этой книге – дать вам прочные навыки эффективного практического применения машинного обучения и глубокого обучения.

После прочтения этой книги вы будете уметь:

- разрабатывать и обучать собственные нейронные сети;
- использовать нейронные сети для понимания данных и создания новых данных;
- присваивать описательные категории текстам, изображениям и другим типам данных;
- предсказывать последующие значения последовательности данных;
- исследовать структуру ваших данных;
- обрабатывать ваши данные с максимальной эффективностью;
- использовать языки программирования и библиотеку DL по своему желанию;
- воспринимать новые знания и идеи и применять их на практике;
- получать удовольствие от обсуждения глубокого обучения с другими специалистами.

Мы используем серьезный, но дружелюбный подход, сопровождаемый большим количеством иллюстраций. Мы делаем это без каких-либо кодов и без всякой математики, за исключением умножения.

Если это звучит привлекательно для вас, добро пожаловать!

Для кого эта книга

Эта книга создана для тех, кто хочет использовать машинное обучение и глубокое обучение в своей работе. Это программисты, инженеры, ученые, руководители, музыканты, врачи и все, кто хочет работать с большими объемами данных, извлекая из них полезную информацию или формируя новые данные.

Многие инструменты машинного обучения и особенно глубокого обучения имеются в многочисленных библиотеках со свободным доступом, которые любой при желании может немедленно загрузить.

Но хотя эти инструменты легко доступны и легко устанавливаются, они все же требуют значительных технических знаний для правильного их применения. Совсем не трудно попросить компьютер выполнить что-то бессмысленное, и он радостно выполнит это, выдав на выходе бессмыслицу.

Такого рода вещи происходят все время. Хотя машинное обучение и глубокое обучение – мощный инструмент, он вовсе не слишком дружелюбен к пользователю.

Выбор правильных алгоритмов и затем применение их надлежащим образом требуют в дополнение последовательных, технически грамотных решений. Когда, как часто бывает, события развиваются не так, как планировалось, необходимы знания, чтобы понять, что происходит внутри системы, с тем чтобы исправить положение дел.

Существует множество подходов к освоению этих существенных знаний в зависимости от того, как вы хотите ими овладеть.

Некоторые любят жесткий детальный алгоритмический анализ, сопровождаемый обширной математикой. Если это тот способ, с помощью которого вы хотите овладеть знаниями, то существуют солидные труды, которые предлагают этот стиль представления знаний [Bishop06], [Goodfellow17]. Он требует серьезных усилий, но дает глубокое понимание того, как и почему этот механизм работает. Если вы выберете данный способ, то вам придется проделать значительную работу, чтобы применить теоретические знания на практике.

Другая крайность, когда человек просто хочет знать, как решить некую конкретную задачу. Серьезных книг, которые содержат библиотеки с рецептами машинного обучения, достаточно много [Chollet17], [Müller-Guido16], [Raschka15], [VanderPlus6]. Эти методы проще, чем математический подход, но вы можете ощущать недостаток структурной информации, объясняющей, как это работает. Без этой информации и соответствующей терминологии трудно разобраться, почему что-то, что, вы полагали, должно работать, не работает или почему что-то не работает так хорошо, как вы полагали. Это может вызвать желание почитать литературу, описывающую новые идеи и результаты, потому что обсуждения обычно предполагают общий уровень знаний о предмете, используемой библиотеке и языке.

В этой книге принят промежуточный подход. Я намерен дать вам инструменты для уверенного практического применения глубокого обучения. Я хочу, чтобы вы могли в вашей работе сделать разумный выбор и были способны следовать в общем потоке новейших идей, появляющихся почти каждый день.

Моя цель здесь – осветить фундаментальные основы достаточно глубоко, чтобы у вас была надежная база поддержки в вашей работе. Я хочу, чтобы вы имели достаточную базу не только для понимания материалов этой книги, но и материалов, которые, возможно, понадобятся вам для консультаций и изучения в процессе работы с глубоким обучением.

Это не книга о программировании. Программирование – важный аспект, но оно неизбежно вовлекает во все детали, которые необязательны для нашего основного предмета изучения. Примеры программирования ограничат нас одной библиотекой или одним языком программирования. И хотя детали необходимы для создания законченных систем, они могут отвлечь нас, когда будет необходимо сосредоточить внимание на основополагающих идеях. Вместо того чтобы вдаваться в дискуссии о циклах, индексах и структуре данных, мы будем все это

обсуждать здесь независимо от какого-либо языка или библиотеки. Если вы усвоите главную идею, то прочтение документации для любой библиотеки будет несложным делом.

Мы спустимся на землю в главах 15, 23 и 24, когда будем обсуждать научную библиотеку машинного обучения и библиотеку глубокого обучения Keras¹. Обе эти библиотеки базируются на языке Python. В этих главах мы погружаемся в детали библиотеки языка программирования Python и имеем в них много примеров с кодами.

Даже если вы не знаете данного языка, эти программы послужат вам примером технологий и программных структур, что поможет справляться с новыми проблемами. Коды в этих главах с программированием доступны в качестве файлов на языке Python. Они могут быть использованы с помощью программного окружения, базирующейся на браузере графической веб-оболочки Jupyter [Jupyter16] или с помощью классического, разработанного для Python окружения, такого как PyCharm [JetBrain17].

Большинство других глав также может быть поддержано выборочными файлами (notebooks) на Python. В них приводится код для каждой компьютерной графики в книге, часто используя методы, обсуждаемые в этой главе. Поскольку в книге отсутствует ориентация на Python и программирование (за исключением упомянутых выше глав), эти записи присутствуют в книге как бы «за сценой» и только слегка комментируются.

Машинное обучение, глубокое обучение и большие данные оказывают неожиданно быстрое и глубокое влияние на общество во всем мире. Что это означает для людей и культуры – сложный и важный предмет. Ряд интересных книг и статей, обсуждающих эту тему, часто приходит к тонкой смеси положительных и отрицательных выводов [Agüera y Arcas17] [Barrat15] [Domingos15] [Kaplan16].

Почти без математики

Многие, отнюдь неглупые люди не являются поклонниками сложных математических уравнений, и если это вы, то здесь вы дома.

В этой книге почти нет математики. Если вы справляетесь с умножением, то этого достаточно, поскольку это вся математика, которую мы используем.

Большинство обсуждаемых нами алгоритмов базируется на солидных теоретических источниках и является результатом аккуратного анализа и проработки. Важно знать это обстоятельство, когда вы модифицируете алгоритм для какой-то новой задачи или другой реализации. Но на практике почти все используют хорошо оптимизированные реализации с открытым кодом, написанные экспертами и доступные в бесплатных библиотеках.

Наша цель – понять принципы этих технологий, понять, как применять их надлежащим образом и как интерпретировать результаты. Ничто из этого не требует от нас вдаваться в математические подробности.

Если вы любите математику или хотите ознакомиться с теорией, следуйте ссылкам к каждой главе. Многие из этих материалов превосходны, интеллекту-

¹ См.: Джулли А., Пал С. Библиотека Keras – инструмент глубокого обучения. М.: ДМК Пресс, 2017. – Прим. перев.

ально полновесны и описывают детали, которые я сознательно опустил в этой книге. Но если математика не ваш конек, то и нет необходимости в математических деталях.

Много рисунков

Некоторые идеи более отчетливо проявляются в рисунках, чем в словах. И даже если слова выполняют свою работу, рисунки цементируют идеи. Поэтому эта книга обильно проиллюстрирована рисунками.

Все рисунки данной книги доступны для свободного считывания (см. далее).

Загрузки

Вы можете загрузить Jupiter/Python для этой книги, все рисунки и другие файлы, относящиеся к этой книге, совершенно свободно.

Все файлы (notebooks)

Все файлы оболочки Jupiter/Python в данной книге доступны на GitHub.

Файлы для главы 15 (scikit-learn – Python-модуль для машинного обучения) и глав 23 и 24 (Keras) содержат все коды, которые представлены в этих главах.

Другие файлы доступны в качестве «за сценой» для просмотра того, как получены рисунки. Они слегка документированы и служат скорее справкой, чем учебным пособием.

Файлы опубликованы по лицензии Массачусетского технологического института (MIT), что, по существу, означает, что вы свободны использовать их для любых целей. Никаких гарантий, что в них отсутствуют дефекты, что они будут надежно работать, что они не приведут к сбоям и т. п., не дается. Чувствуйте себя свободными извлекать коды и адаптировать их так, как вам удобно, хотя в лицензии говорится об использовании только в личных целях (это для файлов с пометкой LICENSE).

<https://github.com/blueberrymusic/DeepLearningBookCode-Volume1>

<https://github.com/blueberrymusic/DeepLearningBookCode-Volume2>

Все рисунки

Все рисунки в этой книге доступны на GitHub в формате PNG с высоким разрешением. Вы можете свободно использовать их в классах, обсуждениях, лекциях, докладах, статьях и даже других книгах.

Так же, как код, рисунки опубликованы по лицензии Массачусетского технологического института (MIT), и вы можете использовать их по своему усмотрению, сохраняя уведомление об авторских правах. Используя рисунки, вы не обязаны кредитовать меня как их создателя, но я буду признателен вам, если вы сделаете это.

Наименования файлов соответствуют номерам рисунков в книге, поэтому их нетрудно найти. Когда вы будете искать что-либо визуально, то будет полезно по-

смотреть страницы в уменьшенном формате. Каждая такая страница содержит 20 изображений:

<https://github.com/blueberrymusic/DeepLearningBookFigures--thumbnails>

Сами рисунки сгруппированы в двух томах:

<https://github.com/blueberrymusic/DeepLearningBookFigures-Volume1>

<https://github.com/blueberrymusic/DeepLearningBookFigures-Volume2>

Источники

Перечень литературы и ресурсов содержит другие файлы, такие как шаблоны для иконок глубокого обучения, которые используются нами в этой книге.

<https://github.com/blueberrymusic/DeepLearningBook-Resources>

Опечатки

Хотя мной приняты все возможные меры, в книге такого объема трудно избежать ошибок. Если вы обнаружите что-то, что покажется вам неверным, пожалуйста, дайте мне знать по адресу andrew@dlbasics.com.

Два тома

Книга оказалась очень большой, поэтому я сделал ее двухтомником с примерно одинаковым размером.

Поскольку двухтомник является, по существу, одной книгой, второй том начинается там, где заканчивается первый. Если вы в данный момент читаете второй том, то, следовательно, вы или уже прочитали первый, или чувствуете себя достаточно уверенно для понимания содержания второго тома.

Благодарности

Авторы любят говорить, что никто не пишет книги в одиночку. Мы говорим так, потому что так оно и есть.

Я чрезвычайно благодарен Эрику Брауну (Eric Braun), Эрику Хейнесу (Eric Haines), Стиву Друкеру (Steven Drucker) и Тому Рейке (Tom Reike) за последовательную и воодушевляющую помощь в этом проекте, помогавшую мне уверенно чувствовать себя на протяжении всего времени осуществления проекта. Спасибо за ваше дружелюбие и поддержку.

Большое спасибо моим рецензентам за щедрые и глубокие комментарии, значительно улучшившие книги: Адаму Финкельштейну (Adam Finkelstein), Алексу Колберну (Alex Colburn), Александру Келлеру (Alexander Keller), Алин Рокфорд (Alyn Rockwood), Анджело Песке (Angelo Pesce), Барбаре Монес, (Barbara Mones), Брайану Уайвиллу (Brian Wyvill), Крейгу Каплану (Craig Kaplan), Дагу Робле, (Doug Roble), Эрику Брауну (Eric Braun), Эрику Хейнесу (Eric Haines), Грегу Тэрку (Greg Turk), Джейффи Халтвиstu (Jeff Hultquist), Джессике Ходжинс, (Jessica Hodgins) Кри-

сти Мортон (Kristi Morton), Лезли Истед (Lesley Istead), Луис Авардо (Luis Avarado), Мэтт Фарр (Matt Pharr), Майку Тика (Mike Tyka), Морган МакГвире (Morgan McGuire), Паулю Бэдсли (Paul Beardsley), Паулю Страуссу (Paul Strauss), Петеру Ширли (Peter Shirley), Филиппу Слузалеку (Philipp Slusallek), Сербан Порумбеску (Serban Porumbescu), Стефанусу Ду Тойту (Stefanus Du Toit), Стивен Друкер (Steven Drucker), ВенхАО Ю (Wenhao Yu) и Закори Эриксон (Zackory Erickson).

Особая благодарность ответственным рецензентам Александру Келлеру (Alexander Keller), Эрику Хейнесу (Eric Haines), Джессике Ходжинс, (Jessica Hodgins) и Луис Авардо (Luis Avarado), которые прочитали всю книгу или большую часть рукописи и дали существенные предложения как по представлению содержания, так и по структуре книги.

Благодарю Морган МакГвире (Morgan McGuire) за применение технологии Markdeep, что позволило мне в большей степени сосредоточиться на том, что я должен сказать, чем на том, каким должен быть формат.

Спасибо Тодду Жиманскому (Todd Szymanski) за вдумчивые советы по оформлению и компоновке содержания и титула книги и выявление ошибок в их размещении.

Спасибо первым читателям, которые выявили опечатки и другие проблемы:

Кристиану Форгангу (Christian Forfang), Дэвиду Полу (David Pol), Эрику Хейнесу (Eric Haines), Гопи Мeenакшисундараму (Gopi Meenakshisundaram), Косте Смоленскому (Kostya Smolenskiy), Мурисио Вивес, (Mauricio Vives), Майку Вонгу (Mike Wong) и Мринал Мохит (Mrinal Mohit).

Все эти люди улучшили книгу, но окончательные решения были за мной. И проблемы, которые остались, являются моей ответственностью.

СПРАВОЧНЫЕ МАТЕРИАЛЫ

Этот раздел появляется в каждой главе. Он содержит справки по всем документам, которые относятся к содержанию данной главы. Здесь также могут присутствовать другие полезные материалы: статьи, веб-страницы, документация, блоги и иные источники.

Где только возможно, я предпочитал те источники, которые доступны онлайн, поэтому вы можете немедленно получить их, используя Сеть. Исключение составляют обычно книги, но иногда я привожу важные онлайн-справки, даже если за ними стоят требования оплаты.

[Agüera y Arcas17] *Blaise Agüera y Arcas, Margaret Mitchell, and Alexander Todorov. Physiognomy's New Clothes.* Medium, 2017. <https://medium.com/@blaisea/physiognomys-new-clothes-f2d4b59fdd6a>.

[Barrat15] *James Barrat.* Our Final Invention: Artificial Intelligence and the End of the Human Era. St. Martin's Griffin, 2015.

[Bishop06] *Christopher M. Bishop.* Pattern Recognition and Machine Learning. Springer-Verlag, 2006. С. 149–152.

[Chollet17] *François Chollet.* Deep Learning with Python. Manning Publications, 2017.

[Domingos15] *Pedro Domingos.* The Master Algorithm. Basic Books, 2015.

[Goodfellow17] *Ian Goodfellow, Yoshua Bengio, Aaron Courville*. Deep Learning. MIT Press, 2017. <http://www.deeplearningbook.org/>¹.

[JetBrains17] *JetBrains*. Pycharm Community Edition IDE. 2017. <https://www.jetbrains.com/pycharm/>.

[Jupyter16] The Jupyter team. 2016. <http://jupyter.org/>.

[Kaplan16] *Jerry Kaplan*. Artificial Intelligence: What Everyone Needs to Know. Oxford University Press, 2016.

[Müller-Guido16] *Andreas C. Müller and Sarah Guido*. Introduction to Machine Learning with Python. O'Reilly Press, 2016².

[Raschka15] *Sebastian Raschka*. Python Machine Learning. Packt Publishing, 2015³.

[VanderPlas16] *Jake VanderPlas*. Python Data Science Handbook. O'Reilly Media, 2016.

¹ Гудфеллоу Я., Бенджио И., Курвиль А. Глубокое обучение. М.: ДМК Пресс, 2017. ISBN: 978-5-97060-618-6.

² Мюллер А., Гвидо С. Введение в машинное обучение с помощью Python. М.: Вильямс, 2017. ISBN: 978-5-9908910-8-1.

³ Рашика С. Python и машинное обучение. М.: ДМК Пресс, 2017. ISBN: 978-5-97060-409-0.

Глава 1

Введение в машинное обучение и глубокое обучение

*Краткий обзор идей, языка, технологий,
которые мы будем использовать в этой книге*

1.1. ЗАЧЕМ ЗДЕСЬ ЭТА ГЛАВА

Эта глава позволит нам поближе познакомиться с главными идеями и основной терминологией машинного обучения.

Термин **машинное обучение** охватывает большой круг технологий, которые преследуют одну цель: извлечь имеющую смысл информацию из данных.

Под данными здесь подразумевается все, что может быть записано и измерено (курс акций в удачные дни, или масса других планет, или рост людей, посещающих местную ярмарку, но это могут быть и звуки (слова, произносимые кем-нибудь в его мобильный телефон), картинки (фотографии цветов или кошек), слова (текст газетной статьи или романа), или все, что мы хотели бы изучить или исследовать).

«Имеющая смысл информация» – это все, что мы можем извлечь из данных и что будет полезным для нас в некотором смысле. Мы решаем, что имеет смысл для нас, и затем создаем алгоритм, чтобы найти столь много из наших данных, насколько это возможно.

Выражением «машинное обучение» описывают множество различных алгоритмов и технологий. Было бы неплохо дать строгое определение для этого термина, но им пользуется так много людей, в столь многих различных направлениях, что лучше рассматривать его как имя большого расширенного собрания алгоритмов и принципов, которые позволяют анализировать огромное количество опытных данных для извлечения из них смысловой информации.

Совсем недавно выражение **глубокое обучение** относилось к методам машинного обучения, которые использовали специализированные слои вычислений, следующие один за другим. Это создавало «глубокую» структуру, подобную стопке блинов. Поскольку термин «глубокое обучение» относится к природе создаваемых нами систем, а не какому-либо конкретному алгоритму, он действительно

соответствует такому особому стилю или методу компьютерного обучения. Этот метод в последние годы был значительно усовершенствован.

Рассмотрим некоторые примеры приложений, которые используют машинное обучение для извлечения значащей информации из данных.

1.1.1. Извлечение значащей информации из данных

Почтовому отделению требуется ежедневно сортировать большое количество писем и посылок с написанными от руки почтовыми индексами. Компьютер научили читать эти индексы и автоматически отправлять почту по адресу, как показано на рис. 1.1.

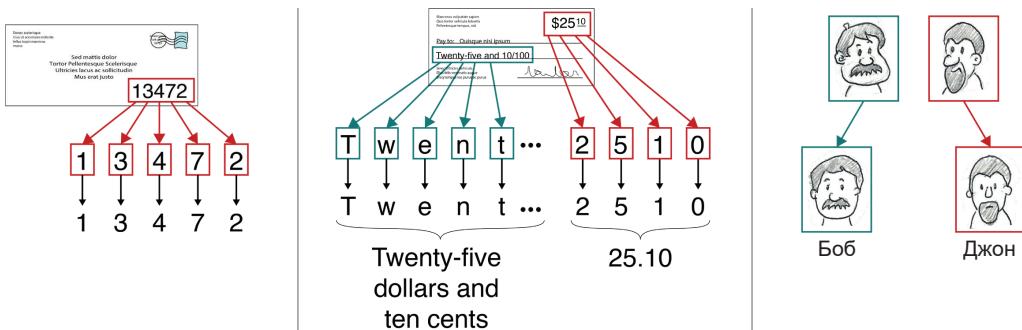


Рис. 1.1 ❖ Извлечение смысловой информации из множества данных. Слева: получение почтового индекса с конверта. В середине: считывание чисел и букв с чека. Справа: распознавание лиц по фотографии

Банки обрабатывают огромное количество рукописных чеков. Надежная проверка требует, чтобы числовая сумма, записанная вручную в соответствующее поле (например, «25.10»), соответствовала выписанной сумме в текстовой строке, например «Twenty-five dollars and ten cents» («Двадцать пять долларов и десять центов»). Компьютер может прочитать оба слова и подтвердить соответствие, как это показано в середине рис. 1.1.

Сайты социальных сетей хотят идентифицировать людей по фотографиям участников сети. Это означает не только обнаружение лиц на данной фотографии, но и определение того, где расположены эти лица, и затем сопоставление каждого лица с ранее наблюдавшимися лицами. Это становится еще более трудным при понимании того, что фактически каждая фотография человека уникальна: освещение, угол зрения, выражение, одежда и много других характеристик будет различаться от любого предыдущего снимка. Они хотели бы быть способными, взяв любое фото любого человека, корректно идентифицировать его, как показано на рис. 1.1.

Производители интеллектуальных цифровых помощников исследуют то, что говорят люди при использовании своих гаджетов, чтобы помощники могли разумно реагировать на речь. Сигнал от микрофона представляет собой последовательность цифр, которые описывают действующее на мембрану микрофона звуковое давление. Поставщики хотят проанализировать эти числа, чтобы понять звуки, вызвавшие их, слова, в которых эти звуки были частью, предложения, в ко-

торые были включены слова, и в конечном счете смысл этих предложений, как это показано в левой части рис. 1.2.

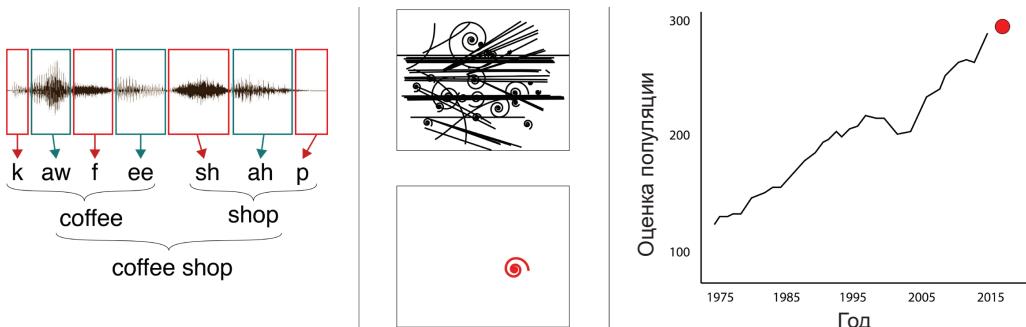


Рис. 1.2 ♦ Извлечение смысловой информации из данных. Слева: превращение записи в звуки, затем в слова и, наконец, в высказывание. В середине: нахождение характерной частицы в смазанном изображении с подобными частицами. Справа: предсказание популяции касаток, обитателей северных районов, у западных берегов Канады [Towers15]

Учеными создается огромное количество данных, поступающих от беспилотных летательных аппаратов, или как результат экспериментов по физике высоких энергий, или в качестве наблюдений за неземным пространством. Из этих переполненных потоков данных часто приходится выделять несколько отдельных событий, которые очень похожи на все остальные, но немного другие. Проанализировать все данные вручную было бы фактически невозможной задачей даже для огромного количества высококвалифицированных специалистов. Гораздо лучше автоматизировать этот процесс с помощью компьютеров, которые могут исчерпывающе разделить данные, никогда не упуская никаких подробностей, как, например, в середине рис. 1.2.

Консервисты отслеживают изменения с течением времени популяции видов животных, чтобы знать, что происходит с популяцией. Если она сокращается в течение длительного периода времени, возможно, требуется вмешательство. Если популяция стабильна или увеличивается, то нет причин для беспокойства. Прогнозирование последующих значений размеров популяции – это то, чему мы можем обучить компьютер, и он будет делать это весьма неплохо. Ежегодно регистрируемый размер популяции китов касаток у побережья Канады, а также его предсказанное возможное значение показаны справа на рис. 1.2 (адаптировано из [Towers15]).

Эти шесть примеров иллюстрируют приложения, которые многим из нас уже знакомы, но таких приложений гораздо больше. Возможность с их помощью быстро извлекать значимую информацию способствует тому, что алгоритмы машинного обучения находят свой путь во все расширяющемся диапазоне областей применения.

Общее место здесь – это объем работы и кропотливое рассмотрение деталей. У нас могут быть миллионы данных для исследования, и мы должны попытаться извлечь какую-то значимую информацию из них. Люди устают, скучают и отвлекаются, но компьютеры будут просто пахать неуклонно и надежно.

1.1.2. Экспертные системы

Первоначальный, ставший популярным подход к поиску смысловой информации, скрытой внутри данных, связан с созданием экспертных систем. Суть идеи заключалась в изучении того, что знают специалисты-эксперты, что они делают, как они это делают, а затем автоматизации этого. По сути, создается компьютерная система, основанная на знаниях экспертов, которая способна имитировать их действия.

Это часто означает разработку **основанной на правилах системы**, в которой сосредоточено большое количество правил, следуя которым, компьютер имитирует специалиста-эксперта. Например, если мы пытаемся распознать цифры почтового индекса, мы можем создать правило, в котором говорится, что семерка имеет форму преимущественно горизонтальной линии вверху изображения, затем преимущественно диагональную линию, которая начинается от правого края горизонтальной линии и продолжается влево и вниз, как это показано на рис. 1.3.

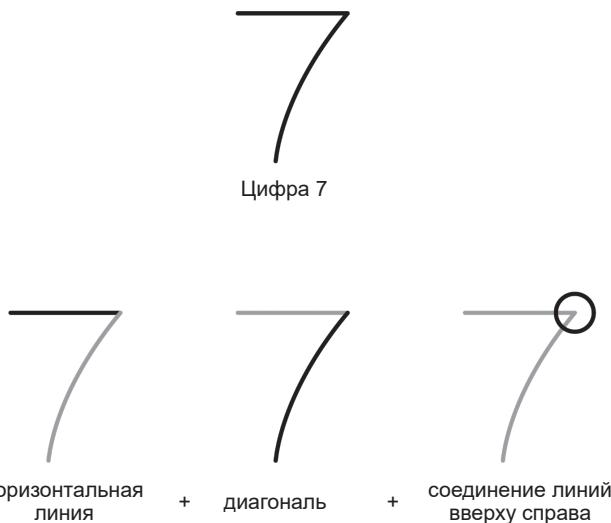


Рис. 1.3 ❖ Составление множества правил для распознавания написанной от руки цифры 7. Вверху: типичная цифра 7, которую мы хотим распознать. Внизу: три правила, создающих цифру 7. Изображение классифицируется как цифра 7, если все три правила соблюдены

Мы можем составить подобные правила для любой цифры. Это могло бы нас удовлетворить, если бы мы не имели цифры 7, изображенной на рис. 1.4. Некоторые люди ставят горизонтальную черту в середине цифры 7. Поэтому мы добавляем еще одно правило для такого специального случая.

Этот процесс ручного составления правил для распознавания иногда называют **техникой описания признаков** (термин также применяется при использовании компьютера для поиска этих признаков [VanderPlas16]).

Данный термин характеризует наше желание выявить (спроектировать) все признаки (или качества), которые объединяются и используются специалистом-экспертом. Вообще-то, это трудная работа. Как мы видели, нетрудно найти одно

правило и даже много таковых. Но представьте попытку нахождения множества правил, по которым рентгенолог определяет, является ли смазанное пятно на рентгеновском снимке доброкачественной опухолью или нет, или правила, которые позволяют авиационному диспетчеру справляться с напряженным графиком полетов, или по которым водитель безопасно ведет машину в сложных погодных условиях.

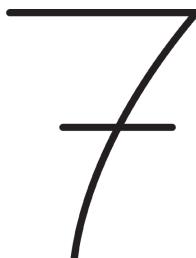


Рис. 1.4 ♦ Эта цифра 7 тоже может существовать, но не будет распознана с правилами на рис. 1.3 из-за дополнительной линии

Экспертные системы, основанные на правилах, способны справляться с некоторыми видами работ, но трудности составления качественного множества правил и необходимость обеспечения надежности их работы при наличии большого разнообразия данных определили невозможность их восприятия как общего решения проблемы. Согласование каждого шага в сложном процессе чрезвычайно затруднительно, а при необходимости также учитывать человеческий фактор при суждениях об опыте и интуиции это становится практически невозможным, за исключением самых простых сценариев.

Привлекательность систем машинного обучения состоит в том, что (на концептуальном уровне) они изучают релевантные характеристики множества данных *автоматически*. Поэтому нам не надо говорить им, как отличить цифру 2 от цифры 7, потому что система разбирается с этим сама. Но, чтобы сделать это хорошо, системе зачастую требуется большое количество данных. Огромное количество данных.

Все это является серьезной причиной того, что машинное обучение пользуется популярностью и реализуется на практике в последние годы. Поток необработанных данных, который обеспечивает интернет, позволяет извлечь множество полезной информации из множества данных. Онлайновые компании способны делать полезным каждое взаимодействие с каждым потребителем, собирая много данных, которые они затем могут обобщить и использовать как входную информацию для алгоритмов машинного обучения, получая тем самым все больше информации о своих потребителях и их пристрастиях.

1.2. ИЗУЧЕНИЕ МАРКИРОВАННЫХ ДАННЫХ

Существует много алгоритмов машинного обучения, и мы будем рассматривать многие из них в этой книге. Многие концептуально прямолинейны (хотя их математика и программирование могут быть сложными). Например, предположим,

что мы хотим найти наилучшую прямую линию, проходящую через группу данных точек, как на рис. 1.5.

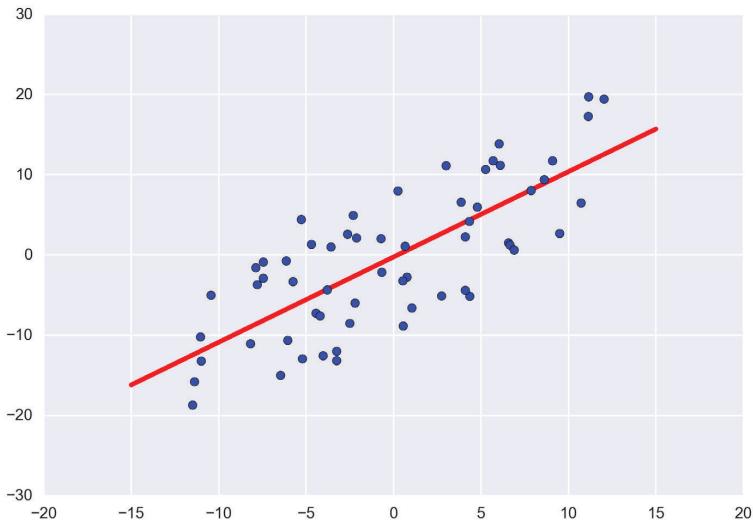


Рис. 1.5 ♦ Имея заданную группу точек (синего цвета), мы можем очевидным образом представить алгоритм, который вычислит наилучшую прямую линию (красного цвета), проходящую через эту группу точек

Концептуально мы можем представить себе алгоритм, который найдет прямую линию, пересекающую только небольшое число точек. Он будет работать в соответствии с некой формулой, чтобы подсчитать то количество точек, заданное число которых будет представлено на входе. Это знакомый алгоритм, который использует тщательно продуманный анализ, чтобы найти лучший способ решить проблему, и затем реализуется в программе, которая выполняет этот анализ. Это стратегия, используемая многими алгоритмами машинного обучения.

Напротив, стратегия, используемая многими алгоритмами глубокого обучения, менее знакома. Она предполагает медленное обучение на примерах, перебирая их снова и снова. Каждый раз, когда программа обнаруживает новые данные для обучения, она улучшает собственные параметры, найдя, в конце концов, набор значений, который обеспечит вычисления с желаемым нам результатом. В процессе выполнения алгоритма его работа значительно более непредсказуема, чем та, при которой ищется прямая линия. Идея здесь в том, что мы не знаем, как очевидным образом получить правильный ответ, и поэтому строим систему, которая сама делает это. Наш анализ и программирование создают алгоритм, который может выработать свои собственные ответы, вместо того чтобы реализовывать известный процесс, ведущий прямо к ответу.

Это действительно звучит немного дико, но это так. Программы, которые могут находить собственные ответы подобным путем, лежат в основе недавнего большого успеха алгоритмов глубокого обучения.

В нескольких последующих разделах мы рассмотрим более подробно эту технику, чтобы лучше понять ее, поскольку она, возможно, менее знакома, чем тради-

ционные алгоритмы машинного обучения. В пределе нам хотелось бы использовать эту технику для таких задач, как получение в качестве ответа системы имен всех, кто изображен на представленной системе фотографии.

Это серьезная задача, поэтому давайте начнем с чего-то значительно более простого и посмотрим на некоторые факторы обучения.

1.2.1. Стратегия обучения

Давайте рассмотрим один ужасный способ обучения детей новому предмету, обобщенный на рис. 1.6. Это совсем не так, как учат большинство детей на самом деле, но один из способов обучения компьютеров.

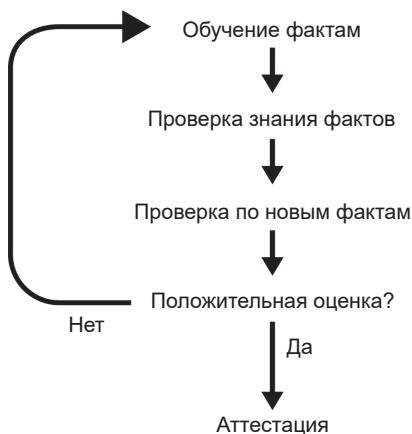


Рис. 1.6 ♦ Действительно ужасный способ пытаться так учить человека. Сначала составляется перечень фактов. Затем каждый студент проверяется на знание этих фактов и других фактов, которые отсутствовали в перечне, но которые, мы полагаем, могут быть извлечены из предъявленного перечня, если перечень был достаточно хорошо усвоен. Если студент получает хорошую оценку по этим тестам (особенно по второму), он аттестуется. В противном случае он возвращается к повторному изучению того же перечня фактов

В этом сценарии (надеюсь, воображаемом) преподаватель стоит перед классом и диктует перечень фактов, который учащиеся, предполагается, запоминают. Каждую пятницу в полдень они сдают два теста. Первый тест проверяет запоминание ими фактов из перечня. Второй тест проводится сразу после первого, при котором задаются вопросы, о которых учащийся заранее не знал, с тем чтобы проверить общее понимание им материала. Конечно, маловероятно, что кто-то будет «понимать» что-либо, если ему будет дан только перечень фактов, что является одной из причин, по которой такой подход ужасен.

Если учащийся получает положительную оценку по второму тесту, то он аттестуется.

Если данный учащийся не сдает второй тест, он на следующей неделе повторяет весь процесс: преподаватель повторяет факты из того же перечня, затем вновь проверяет таких учащихся по первому перечню на запоминание и потом по второму тесту на понимание или способность обобщать. Снова и снова каж-

дый учащийся повторяет этот процесс, пока не пройдет второй тест, чтобы получить аттестацию.

Это ужасный способ обучения для детей, но он оказывается отличным способом обучать компьютеры.

В этой книге мы увидим много других подходов к обучению компьютеров, но ограничимся пока этим и посмотрим на него более внимательно.

Мы увидим, что не в пример многим людям каждый раз, когда мы будем предъявлять компьютеру одну и ту же информацию, он будет учиться немного больше.

1.2.2. Стратегия компьютерного обучения

Мы начнем с накопления фактов, которым собираемся обучить. Мы будем набирать так много данных, сколько только сможем. Каждую группу наблюдаемых данных (скажем, погоды в данный момент) называют **выборкой**, а имена измерений, которые осуществлялись (температура, скорость ветра, влажность и т. д.), называют **признаками**, или **характеристиками** [Bishop06]. Каждое имя измерения или признак ассоциируется с величиной, обычно запоминаемой как число. Чтобы подготовить наши данные для компьютера, мы предъявляем каждую выборку (то есть группу данных с величинами всех признаков) эксперту, который изучает эти признаки и осуществляет **маркировку** этой выборки. Например, если наша выборка является фотографией, маркировкой может быть имя человека или тип животного, показанные на ней, или, скажем, присутствует или нет движение транспорта на фотографии или транспорт на ней стоит в пробке.

Давайте используем в нашем примере изменения погоды в горах. Мнение эксперта, применяя шкалу от 0 до 100, говорит о его уверенности, что погода благоприятствует прогулке. Идея показана на рис. 1.7.

Мы будем обычно использовать эти маркированные выборки, но сейчас на время отложим их.

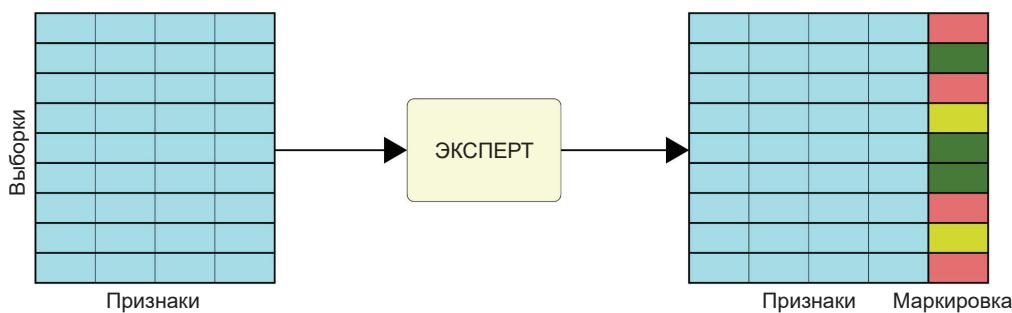


Рис. 1.7 ♦ Чтобы маркировать множество данных, мы начинаем с перечня выборок или элементов данных. Каждая выборка имеет перечень признаков, которые описывают ее. Мы даем это множество данных эксперту, который исследует признаки каждой выборки одной за другой и маркирует каждую выборку

Когда мы получим маркированные выборки, мы вводим их в компьютер и говорим ему, что надо найти способ уточнить правильность маркировки для каждого входа. Мы *не говорим* ему, как это делать. Вместо этого мы даем алгоритм

с большим количеством параметров (возможно, даже миллион), которые он может уточнять.

Различные способы обучения будут использовать различные алгоритмы, и многое в этой книге будет посвящено их рассмотрению и тому, как их успешно использовать. Когда мы выбрали алгоритм, мы можем загрузить данные на его вход, чтобы получить результат на выходе. Этот результат будет являться компьютерным **предсказанием** того, что компьютер думает об экспертной маркировке для этой выборки.

Если предсказание компьютера согласуется с экспертной маркировкой, мы не будем ничего менять. Но когда компьютер ошибается, мы попросим его модифицировать внутренние, используемые им параметры алгоритма, так чтобы при представлении ему этих данных вновь он лучше предсказал правильные ответы.

В основе такого процесса лежит метод проб и ошибок. Компьютер делает все возможное, чтобы дать нам правильный ответ, и если снова ошибается, то существует процедура, которой он может следовать и которая помогает ему изменять и улучшать ответ.

Мы только проверяем предсказания компьютера относительно маркировки эксперта, после того как оно будет получено. Если оно не согласуется, подсчитываем **ошибку**, называемую также **ценой**, или **потерями**. Это величина, которая говорит алгоритму, насколько он хорош. Система использует текущие значения своих внутренних параметров, экспертное предсказание (которое мы знаем) и свое собственное неверное предсказание, чтобы уточнить параметры алгоритма таким образом, чтобы предсказать более правильную маркировку при новом предъявлении ему той же выборки. Позже мы посмотрим более подробно, как осуществляются такие шаги. На рис. 1.8 проиллюстрирована эта идея.

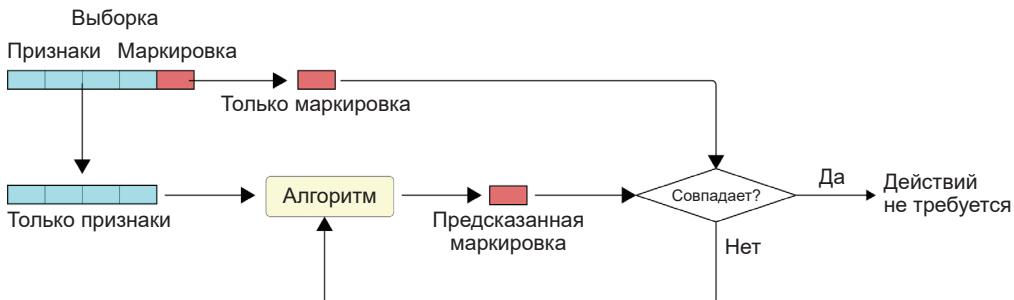


Рис. 1.8 ♦ Один шаг тренинга, или обучения. Мы разделяем признаки и их маркировку. По признакам алгоритм предсказывает маркировку. Мы сравниваем предсказание с реальной маркировкой. Если предсказанная маркировка согласуется с той, которую мы хотим, то ничего не предпринимаем. В противном случае мы говорим алгоритму модифицироваться или обновиться таким образом, чтобы в меньшей степени повторить эту ошибку вновь

Мы говорим, что **тренируем** систему, чтобы **обучить**, как **предсказывать** маркированные данные посредством анализа выборок в результате **последовательных тренировок и обновления** алгоритма при неправильном предсказании.

Мы посмотрим на другие возможные алгоритмы и шаги обновления в деталях в последующих главах. Сейчас важно понять, что каждый алгоритм обучает по-

средством изменения своих внутренних параметров, чтобы сделать предсказание. Он может значительно изменить их после неверного предсказания, но есть опасность, что он изменит их так сильно, что сделает предсказание еще хуже. Он может менять их в небольших пределах, но будет работать при этом медленнее, чем мог бы. Правильное соотношение между этими крайностями – это то, что мы должны найти с помощью проб и ошибок для каждого типа алгоритма и каждого множества данных в результате тренинга. Мы назовем количество обновлений **скоростью обучения**. Маленькая скорость обучения медленная, но осторожная, в то время как большая скорость обучения хотя и ускоряет процесс, но может привести к ошибкам.

В качестве аналогии предположим, что мы находимся в пустыне и используем металлоискатель для поиска закопанного ящика с провиантом. Мы видим датчиком металлоискателя над поверхностью, получаем отклик в каком-то направлении и продвигаемся в этом направлении. Если мы будем осторожны, мы будем двигаться малыми шагами, так, чтобы не пройти мимо ящика и не потерять сигнал. Если мы будем излишне активны, то будем двигаться большими шагами, чтобы найти ящик как можно быстрее. И только если мы, начав двигаться большими шагами, будем делать их все меньше и меньше при приближении к ящику, мы сможем найти его. Таким же образом обычно регулируется скорость обучения: последовательность изменений большая вначале, затем в процессе обучения уменьшается.

Есть занятный способ получения компьютером отличной оценки без всякого обучения – ничего не делать, а просто запомнить входы. Чтобы получить идеальный результат, все, что нужно алгоритму, – это просто запомнить маркировку эксперта для каждой выборки, а затем выдать нам эту маркировку. Другими словами, нет необходимости обучать как-то вычислить маркировку для данной выборки. Просто надо посмотреть правильный ответ в таблице. В нашем воображаемом сценарии обучения учащихся это эквивалентно запоминанию учащимся ответов на все вопросы в тестах.

Иногда это хорошая стратегия, и мы увидим далее, что существуют полезные алгоритмы, следующие именно такой стратегии. Но если наша цель – научить компьютер узнать что-либо о данных, что позволило бы обобщить это для получения новых данных, такой способ, как правило, неприемлем. Проблема в том, что мы уже имеем маркировки, запомненные системой, так что наша работа по ее тренировке не даст нам ничего нового, поскольку компьютер ничего не узнает из этих данных, а просто возьмет ответы, заглянув в таблицу, и у него не будет идеи, как выработать предсказание для новых данных, которые ему были еще неизвестны, и запомнить их. Задача в том, чтобы получить систему, способную прогнозировать маркировки для новых данных, которые ей не были известны раньше, чтобы мы могли затем уверенно использовать их в системе, на вход которой новые данные для предсказания будут поступать постоянно.

Если алгоритм хорошо работает с обучающим множеством, но плохо работает с новыми данными, мы будем говорить, что он плохо **обобщает**.

1.2.3. Обобщение

Теперь вернемся к маркировке данных, о которой мы не стали рассуждать в предыдущем разделе.

Мы дадим оценку того, как хорошо система может обобщать то, чему она научилась, показывая ей выборки, которых она еще не видела. Это тестовое множество скажет нам, как хорошо система работает с новыми данными.

Давайте посмотрим на **классификатор** (или **категорайзер**). Такая система присваивает маркировку каждой выборке, которая описывает, к какой из нескольких категорий, или классов, эта выборка принадлежит. Если выборка на входе представляет собой песню, то маркировка может быть жанром (например, рок или классика). Если это фотография животного, то маркировка может говорить, какое это животное (например, тигр или слон). В нашем примере мы можем маркировать каждый день как подходящий для прогулки на три категории: плохой, хороший, превосходный.

Мы попросим компьютер предсказать маркировку для каждой выборки из тестового множества (выборок, которые ему не были представлены ранее) и затем сравним предсказания компьютера и маркировку эксперта, как это показано на рис. 1.9.

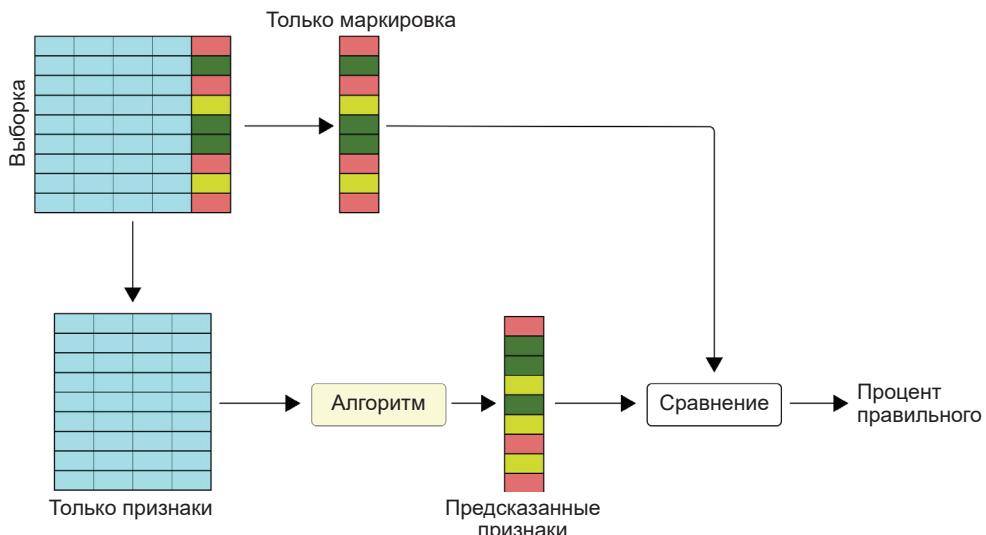


Рис. 1.9 ♦ Полный процесс оценки классификатора (называемого также категоризатором)

На рис. 1.9 мы разделили тестовые данные на признаки и маркировку. Алгоритм присваивает или предсказывает маркировку для каждого множества признаков, затем мы сравниваем предсказание с реальной маркировкой, чтобы оценить правильность. Если она достаточно хороша, мы можем внедрять систему. Если результат не слишком хороший, мы можем вернуться и продолжить тренинг. Заметьте, что, в отличие от процесса тренинга, в процессе на рис. 1.9 отсутствуют обратная связь и обучение. Пока мы не вернемся к улучшающему тренингу, алгоритм не изменит своих параметров независимо от качества предсказания.

Если предсказания компьютера с этой новой выборкой (точнее, новой для компьютера) недостаточно хорошо согласуются с маркировкой эксперта, то мы возвращаемся к процессу, изображенному на рис. 1.8. С каждым шагом мы

предъявляем компьютеру каждую выборку обучающего множества снова, давая ему возможность опять изучить ее таким же образом, как и раньше. Подчеркнем, что это те же выборки, то есть мы просим компьютер изучить снова и снова те же самые данные. Обычно их сначала **перемешивают** так, что выборки поступают каждый раз в различной последовательности, но алгоритм при этом новой информации не получает. Затем мы просим алгоритм снова предсказать маркировку для тестового множества. Если она и на этот раз недостаточно хорошая, то мы проводим обучение с оригинальным выборочным множеством снова и затем снова проводим тестирование. Мы делаем это снова и снова, повторяя процесс и предъявляя компьютеру снова и снова те же данные, просто позволяя ему каждый раз обучаться дольше и дольше. Как было уже сказано, это ужасный путь обучения учащихся, но компьютер от этого не заскучает и не возмутится, а только каждый раз будет изучать возможность стать немного лучше, извлекая новую порцию информации из данных.

1.2.4. Более внимательный взгляд на обучение

Обычно мы верим, что в наших данных присутствует взаимосвязь. В конце концов, если данные просто чисто случайны, мы не будем пытаться извлечь из них информацию. Процесс в предыдущем разделе заключается в том, что, заставляя компьютер снова и снова обращаться к обучающей выборке, мы надеемся, что, обучаясь каждый раз понемногу от каждой выборки, алгоритм в конечном итоге найдет эту взаимосвязь между признаками в каждой выборке и осуществит маркировку, соответствующую маркировке эксперта. Затем он использует найденную взаимосвязь в новых данных при тестировании. Если он получит в большинстве своем правильные ответы, то можно говорить о высокой точности, или малой **ошибке обобщения**.

Но если компьютер постоянно не улучшает свое предсказание маркировки при предъявлении тестовой выборки, мы останавливаем обучение, поскольку прогресс в обучении отсутствует. Обычно в таком случае мы модифицируем алгоритм в надежде, что получим более удовлетворительный результат, и затем снова начинаем процесс обучения. Но это только наша надежда. Нет гарантии, что это будет для каждого множества хороший обучающий алгоритм, как и отсутствует гарантия, что мы вообще найдем таковой. Хорошей новостью является то, что мы без какой-либо математической гарантии часто можем на практике найти решения, которые иногда работают даже лучше, чем эксперты.

Одной из причин, по которой алгоритм может потерпеть неудачу при обучении, является отсутствие у него достаточных компьютерных ресурсов, чтобы найти взаимосвязь между выборками и маркировками. Мы иногда думаем о создании своего рода компьютерной **модели**, лежащей в основе данных. Например, если температура ежедневно растет в первые три часа, когда мы измеряем ее, компьютер может создать «модель», сообщающую, что по утрам температура растет. Это версия данных. Таким же образом маленькая пластиковая версия спортивного автомобиля или самолета является «моделью» этих средств транспорта. Классификатор, который мы рассматривали раньше, является одним из примеров модели.

В компьютере модель формируется структурой программного обеспечения и величинами параметров, которые она использует. Большая программа и большие множества параметров могут привести к моделям, которые способны узнать

больше из данных. Мы говорим, что они имеют большую **емкость**, или **представительскую мощность**. Этот термин говорит нам о том, насколько глубоко и широко возможно обучение данным алгоритмом. Большая емкость позволяет в большей степени раскрыть смысловую информацию из имеющихся у нас данных.

В качестве аналогии представим, что мы работаем с автомобильным дилером и нам предложили написать аннотацию на продаваемые нами автомобили. Предположим, что отдел продаж прислал нам перечень «слов одобрения», описывающих продаваемые нами автомобили. Через некоторое время после выполнения этой работы мы, возможно, поймем, как наилучшим образом представить каждый автомобиль этой модели.

Предположим, этот дилер затем купил наш первый мотоцикл. Слова, предложенные нам, или наша модель, не имеют достаточной емкости, чтобы представить это транспортное средство в дополнение ко всем другим транспортным средствам, которые мы уже описали. Нам просто недостаточно имеющегося словаря, чтобы отнести 2-колесное транспортное средство к 4-колесному. Если мы можем использовать более мощную модель с большей емкостью (то есть более обширный словарь для описания транспортных средств), мы сможем сделать такую работу лучше.

Большая модель означает и больший объем работ. И позже мы увидим в этой книге, что большая модель может часто дать лучший результат, чем маленькая, хотя потребует больше компьютерного времени и компьютерной памяти.

Величины, которые компьютер модифицирует с течением времени сам, называются его **параметрами**. Но наш обучающий алгоритм также управляетяется величинами, которые установили мы (такими, например, как скорость обучения, о которой говорилось ранее). Они именуются более неуклюжим словом **гиперпараметры**.

Разница между параметрами и гиперпараметрами состоит в том, что компьютер уточняет свои величины параметров в процессе обучения, в то время как гиперпараметры устанавливаются нами при написании и выполнении программ.

Если алгоритм достаточно хорошо обучен и удовлетворяет нас, хорошо справляясь с тестовым множеством, значит, он готов к **внедрению** или **реализации** во внешнем мире. Наши заказчики введут данные, и наша система выдаст правильную маркировку с предсказаниями. И таким образом изображения лиц скажут об именах, звуки – о словах, а измерения параметров среды превратятся в прогноз погоды.

Обзор литературных источников обширной и растущей области машинного обучения включает книги, целиком посвященные ей [Bishop06], [Goodfellow17].

В этой главе будет дан обзор основных категорий современных инструментов в данной области. Мы затем будем возвращаться к этим типам алгоритмов много-кратно на протяжении всей книги.

1.3. НАБЛЮДАЕМОЕ ОБУЧЕНИЕ

Когда наши выборки поступают с предварительно обозначенными маркировками, как в примерах, приведенных ранее, мы говорим, что мы осуществляем **наблюдаемое обучение**. Наблюдением являются наши маркировки. Они поступают для сравнения с предсказанной маркировкой, как это показано на рис. 1.8,

и алгоритм решает, является предсказанная маркировка правильной или нет. Существует два основных типа наблюдаемого обучения, называемых **классификацией и регрессией**. Классификация относится к наблюдению за данным набором категорий, с тем чтобы найти одну, которая наилучшим образом описывает специфический вход. Регрессия имеет дело со множеством измерений и предсказанием некой другой величины (часто той, которая поступает следующей, но может также и предшествовать началу множества или быть где-то в середине его).

Рассмотрим их по очереди.

1.3.1. Классификация

Предположим, у нас есть коллекция фотографий повседневных предметов, и мы хотим сортировать их по изображению: яблочный чиллер, саламандра, пианино и т. д. Мы говорим, что хотим классифицировать, или категоризировать, эти фотографии. Процесс называется классификацией, или категоризацией.

В этом методе мы начинаем тренинг, предъявляя компьютеру перечень всех маркировок (или классов, категорий), которым мы хотим его научить. Этот перечень – обычно просто комбинация всех маркировок для всех выборок в обучающем множестве, из которого изъяты дубликаты.

Затем мы обучаем систему с помощью множества фотографий и их маркировок, пока не определим, что система хорошо справляется с предсказанием правильной маркировки для каждой фотографии.

Теперь мы можем проверить систему на новых фотографиях, которые раньше ей не предъявлялись. Мы ожидаем, что она будет правильно маркировать изображения тех объектов, которые она видела во время обучения. Но если предмет не распознается или принадлежит к категории, которая не была частью обучающего множества, то система попытается выбрать лучшую категорию из тех, о которых она знает. На рис. 1.10 показана идея.



Колибри



Ковш



Штопор



Кольцевой зажим

Рис. 1.10 ❖ Когда мы осуществляем квалификацию, то обучаем классификатору комплектом изображений, каждое из которых ассоциируется с маркировкой. По окончании обучения мы можем предъявить новые изображения, и компьютер будет пытаться выбрать лучшую маркировку для каждого изображения. Классификатор не обучался на металлическую ложку или наушники, поэтому показал наилучшее приближение, которое он смог найти

На рис. 1.10 показаны четыре предъявленные обученному классификатору для идентификации изображения, которые он никогда не видел прежде [Simonyan14]. Замечательно, что компьютер идентифицировал штопор, даже притом, что он стилизован по форме под лодку. Система не была тренирована на изображения металлических ложек или наушников, но в обоих случаях нашла наилучший ва-

риант изображения. Чтобы корректно отобразить эти предметы, система должна была видеть во время обучения множество примеров подобных предметов.

1.3.2. Регрессия

Предположим, что мы имеем неполный состав измерений и хотим оценить недостающие значения. Скажем, мы подсчитывали количество посещений серии концертов на местной сцене. Выступавший ансамбль каждый вечер получал плату за выступление, составлявшую часть средств, вырученных от продажи билетов на этот концерт.

К сожалению, был потерян отчет о присутствии зрителей на одном из вечерних концертов, и, для того чтобы спланировать наш бюджет, мы хотели бы знать, какая посещаемость концерта ожидается завтра. Подсчеты, которые были сделаны, показаны в левой части рис. 1.11. Наши оценки отсутствующих значений даны в правой части рис. 1.11.

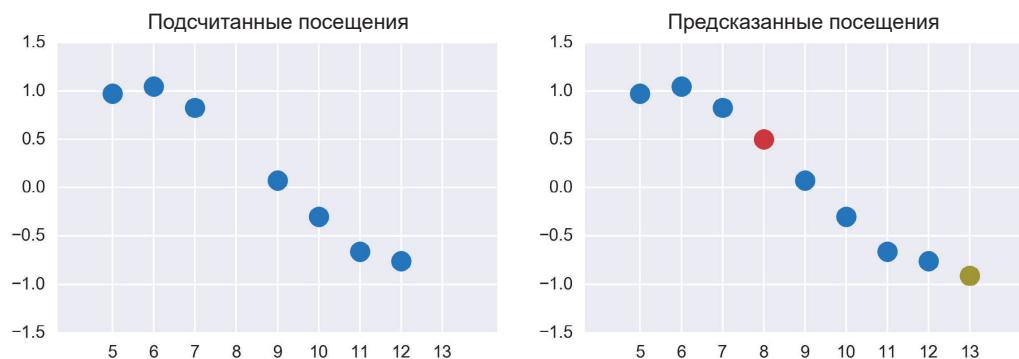


Рис. 1.11 ♦ При регрессии мы имеем множество входных и выходных значений. Здесь входами являются даты конвертов с 5 по 13 мая, а выходами – их посещения зрителями. Слева: измеренные данные с отсутствующими количествами посещений 8 мая. Справа: точки красного цвета являются оценкой отсутствующих данных 8 мая, а желтого цвета – предсказание посещения концерта зрителями 13 мая

Мы называем этот процесс восполнения и предсказания данных задачей **регрессии**. Это название, возможно, несколько не подходящее, поскольку «регрессия» означает возврат к более ранним условиям, а здесь, кажется, не происходит никакого возврата к прошлому.

Несоответствующее употребление этого слова появилось в статье, опубликованной в 1886 году, в которой ученый сообщал об исследовании роста детей [Galton86]. Он нашел, что, несмотря на то что некоторые имеют высокий рост, а другие – низкий, существует тенденция к тому, что рост людей со временем выравнивается. Он описал это явление как «регрессия к усреднению», что означало, что измерения имеют тенденцию двигаться от экстремальных значений к средним значениям.

Хотя это определение приписывают Галтону (Galton), очень похожее замечание высказано в обзоре Дарвина (Darwin) «О происхождении видов» (*On the Origin of Species*) [Darwin59]. Обозреватель Флеминг Джэнкин (Fleeming Jenkin) высказывал мысль, что вариации видов будут преданы забвению, поскольку существует

«общая тенденция возврата всего к стабильной посредственности» [Bryson04].

Сегодня слово «посредственность» имеет негативный оттенок, поэтому теперь чаще говорят «ретрессия к среднему». Слово «ретрессия» продолжает применяться для воплощения идеи использования статистических свойств данных для оценки отсутствующих или будущих значений.

Проблема «регрессии» – это просто одна из тех, при которых мы имеем некоторые значения, зависящие от входных значений (подобно тому, как посещения являются функцией дня месяца), и мы предсказываем новое значение для новых данных на входе.

Наиболее употребительным видом регрессии является **линейная регрессия**. Определение «линейная» относится к попытке согласовать наши входные данные с прямой линией, как это показано на рис. 1.12.

Прямая линия предпочтительна из-за ее простоты, но в этом примере она не слишком хорошо описывает данные. Наши данные поднимаются и опускаются относительно нее, и прямая линия не охватывает их. Это не самое плохое соглашение, но оно и не самое лучшее.

В качестве альтернативы мы можем использовать более сложную форму регрессии, применив более сложный тип кривой, как показано справа на рис. 1.12. Это может обеспечить лучшее согласование с данными, но требует большего компьютерного времени. Поскольку наши кривые стали более сложными, они обычно требуют и больше данных при работе с ними.



Рис. 1.12 ♦ Представление точек данных в математической формуле. Слева: линейная регрессия соглашается прямую линию (красного цвета) и данные (синего цвета). Линия не очень хорошо согласуется с данными, но ее достоинством является простота. Справа: более сложная регрессия соглашается кривой тех же данных. Она лучше согласуется с данными, но имеет более сложную форму и требует большей работы (и больше времени) при вычислениях

1.4. НЕНАБЛЮДАЕМОЕ ОБУЧЕНИЕ

Когда наши данные поступают на вход немаркованными, мы говорим, что любой обучающий алгоритм с данными на входе относится к категории **ненаблюдаемого обучения**. Это просто означает, что мы не «контролируем» процесс обучения посредством марковок. Система должна сама осуществить вычисления без нашей помощи.