Содержание

Об авторе 13
О рецензенте 13

Введение 15

О чем эта книга 15
Для кого предназначена книга 17
Программная часть 17
Принятые соглашения 18
Файлы примеров и файл цветной вклейки 19
Ждем ваших отзывов! 19

Глава 1. Введение в искусственный интеллект 21

Что такое искусственный интеллект 22
Зачем нужно изучать ИИ? 22
Области применения ИИ 25
Направления исследований ИИ 27
Оценка уровня искусственного интеллекта с помощью теста Тьюринга 30
Как научить машины мыслить подобно людям 32
Создание рациональных агентов 34
Универсальный решатель задач 35
Решение задач с помощью GPS 36
Создание интеллектуальных агентов 37
Типы моделей 38
Установка Python 3 39
Установка в Ubuntu 39
Установка в Mac OS X 39
Установка в Windows 40
Установка пакетов 40
Загрузка данных 41
Резюме 43

Глава 2. Классификация и регрессия посредством обучения с учителем 45

Обучение с учителем и без учителя 45
Что такое классификация 46
Предварительная обработка данных 47
Бинаризация 48
Исключение среднего 48

Масштабирование 49
Нормализация 50
Кодирование меток 51
Логистический классификатор 52
Наивный байесовский классификатор 57
Матрица неточностей 61
Машины опорных векторов 64
Классификация данных о доходах с помощью машин опорных векторов 65
Что такое регрессия 69
Создание регрессора одной переменной 70
Создание многомерного регрессора 73
Оценка стоимости недвижимости с использованием регрессора на основе машины опорных векторов 75
Резюме 77

Глава 3. Предсказательная аналитика на основе ансамблевого обучения 79

Что такое ансамблевое обучение 79
Построение моделей обучения посредством ансамблевого метода 80
Что такое деревья принятия решений 81
Создание классификатора на основе дерева принятия решений 81
Случайные и предельно случайные леса 85
Создание классификаторов на основе случайных и предельно случайных лесов 86
Оценка мер достоверности прогнозов 91
Обработка дисбаланса классов 95
Нахождение оптимальных обучающих параметров с помощью сеточного поиска 100
Вычисление относительной важности признаков 103
Прогнозирование интенсивности дорожного движения с помощью классификатора на основе предельно случайных лесов 106
Резюме 109

Глава 4. Распознавание образов с помощью обучения без учителя 111

Что такое обучение без учителя 111
Кластеризация данных с помощью метода k-средних 112
Оценка количества кластеров с использованием метода сдвига среднего 116
Оценка качества кластеризации с помощью силуэтных оценок 121
Что такое смешанные гауссовские модели 124
Создание классификатора на основе гауссовской смешанной модели 126

Нахождение подгрупп на фондовом рынке с использованием модели распространения сходства 130
Сегментирование рынка на основе моделей совершения покупок 133
Резюме 136

Глава 5. Создание рекомендательных систем 137

Создание обучающего конвейера 137
Извлечение ближайших соседей 140
Создание классификатора методом K ближайших соседей 143
Вычисление оценок сходства 150
Поиск пользователей с похожими предпочтениями методом коллаборативной фильтрации 155
Создание рекомендательной системы фильмов 158
Резюме 162

Глава 6. Логическое программирование 163

Что такое логическое программирование 163
Конструкции логического программирования 165
Решение задач с помощью логического программирования 166
Установка пакетов Python 167
Сопоставление математических выражений 167
Проверка простых чисел 169
Парсинг генеалогического дерева 170
Анализ географических данных 176
Создание решателя головоломок 180
Резюме 183

Глава 7. Методы эвристического поиска 185

Что такое эвристический поиск 185
Неинформированный и информированный виды поиска 186
Задачи с ограничениями 187
Методы локального поиска 187
Алгоритм имитации отжига 188
Конструирование строк с использованием жадного поиска 189
Решение задачи с ограничениями 193
Решение задачи о раскраске областей 197
Создание головоломки “8” 200
Создание решателя для прохождения лабиринта 205
Резюме 210

Глава 8. Генетические алгоритмы 211

Эволюционные и генетические алгоритмы 211
Фундаментальные понятия генетических алгоритмов 212
Генерация битовых образов с предопределенными параметрами 214
Визуализация хода эволюции 221
Решение задачи символической регрессии 230
Создание контроллера интеллектуального робота 235
Резюме 242

Глава 9. Создание игр с помощью искусственного интеллекта 245

Использование поисковых алгоритмов в играх 245
Комбинаторный поиск 246
Алгоритм MiniMax 247
Альфа-бета-отсечение 247
Алгоритм NegaMax 248
Установка библиотеки **`easyAI`** 249
Создание робота для игры Last Coin Standing (“Последняя монета”) 249
Создание робота для игры Tic-Tac-Toe (“крестики-нолики”) 253
Создание двух роботов, играющих между собой
в игру Connect Four (“Четыре в ряд”) 257
Создание двух роботов, играющих между собой
в игру Hexapawn (“Шесть пешек”) 261
Резюме 265

Глава 10. Обработка естественного языка 267

Введение и установка пакетов 267
Токенизация текстовых данных 269
Преобразование слов в их базовые формы с помощью стемминга 270
Преобразование слов в их корневые формы с помощью
лемматизации 272
Разбиение текстовых данных на информационные блоки 274
Извлечение частотности слов с помощью модели Bag of Words 276
Создание прогнозатора категорий 280
Создание анализатора грамматических родов 283
Создание сентимент-анализатора 286
Тематическое моделирование с использованием
латентного размещения Дирихле 290
Резюме 294

Глава 11. Вероятностный подход к обработке последовательных данных 295
Что такое последовательные данные 295
Обработка временных рядов с помощью библиотеки Pandas 297
Извлечение срезов временных рядов данных 300
Выполнение операций над временными рядами 302
Извлечение статистик из временных рядов данных 305
Генерация данных с использованием скрытых марковских моделей 309
Идентификация буквенных последовательностей с помощью условных случайных полей 313
Анализ биржевого рынка 317
Резюме 321

Глава 12. Создание систем распознавания речи 323
Работа со звуковыми сигналами 323
Визуализация аудиосигналов 324
Преобразование аудиосигналов в частотные интервалы 326
Генерирование аудиосигналов 329
Синтезирование звуков для генерации музыки 331
Извлечение речевых признаков 333
Распознавание произносимых слов 337
Резюме 343

Глава 13. Обнаружение и отслеживание объектов 345
Установка библиотеки OpenCV 346
Вычисление разности между кадрами 346
Отслеживание объектов с помощью цветовых пространств 349
Отслеживание объектов путем вычитания фоновых изображений 353
Создание интерактивного трекера объектов с помощью алгоритма CAMShift 357
Отслеживание объектов с использованием оптических потоков 364
Обнаружение и отслеживание лиц 370
Использование каскадов Хаара для обнаружения лиц 371
Использование интегральных изображений для извлечения признаков 372
Отслеживание глаз и определение координат взора 375
Резюме 378

Глава 14. Искусственные нейронные сети 379
Введение в искусственные нейронные сети 379
Создание нейронной сети 380
Тренировка нейронной сети 380
Создание классификатора на основе перцептрона 381
Построение однослойной нейронной сети 384
Построение многослойной нейронной сети 388
Создание векторного квантизатора 392
Анализ последовательных данных с помощью рекуррентных нейронных сетей 395
Визуализация символов с использованием базы данных оптического распознавания символов 399
Создание системы оптического распознавания символов 401
Резюме 404

Глава 15. Обучение с подкреплением 407
Основы обучения с подкреплением 407
Обучение с подкреплением и обучение с учителем 408
Реальные примеры обучения с подкреплением 409
Строительные блоки обучения с подкреплением 410
Создание окружения 411
Создание агента обучения 416
Резюме 420

Глава 16. Глубокое обучение и сверточные нейронные сети 421
Что такое сверточные нейронные сети 421
Архитектура CNN 422
Типы слоев CNN 423
Создание линейного регрессора на основе перцептрона 424
Создание классификатора изображений на основе однослойной нейронной сети 431
Создание классификатора изображений на основе сверточной нейронной сети 433
Резюме 439

Предметный указатель 441

1

Введение в искусственный интеллект

В этой главе мы обсудим понятие искусственного интеллекта (ИИ) и способы его применения для решения реальных задач. Значительную часть своей повседневной жизни мы проводим, взаимодействуя с интеллектуальными системами. Такое взаимодействие происходит во время поиска информации в Интернете, биометрического распознавания лиц, отдавания голосовых команд. Все эти виды взаимодействий основаны на использовании систем искусственного интеллекта, которые становятся важным фактором современного стиля жизни. Подобные системы представляют собой сложные приложения, в которых для решения конкретных задач с помощью искусственного интеллекта привлекаются математические методы и программные алгоритмы. В этой книге вы ознакомитесь с фундаментальными принципами, лежащими в основе создания приложений подобного рода, и изучите примеры их практической реализации. Конечная цель — научить вас не бояться браться за новые и трудные задачи, поддающиеся решению с помощью искусственного интеллекта, с которыми вы можете столкнуться в процессе своей практической деятельности.

В этой главе вы ознакомитесь со следующими темами:

- что такое ИИ и почему следует его изучать;
- применения ИИ;
- разновидности ИИ;
- тест Тьюринга;
- рациональные агенты;

- универсальные решатели задач;
- создание интеллектуальных агентов;
- установка Python 3 в различных операционных системах;
- установка необходимых пакетов Python.

Что такое искусственный интеллект

Искусственный интеллект (ИИ) позволяет наделять машины возможностями, имитирующими интеллектуальное поведение человека и его способность рассуждать. Машины управляются программным обеспечением, поэтому ИИ имеет много общего с интеллектуальными программами, контролирующими поведение машин. Наука об ИИ разрабатывает теории и методологии, позволяющие машинам оценивать окружающую обстановку и реагировать на различные ситуации так, как на них реагировал бы человек.

Как показывает тщательный анализ направлений развития науки об ИИ, в своих попытках дать определение ИИ ученые применяли различные подходы. В современном мире ИИ задействуется во многих областях, принимая самые разнообразные формы. Мы хотим, чтобы машины могли ощущать и рассуждать, думать и действовать. Кроме того, мы хотим, чтобы поведение машин было рациональным.

Работы в области ИИ тесно связаны с изучением свойств человеческого мозга. Исследователи полагают, что понимание принципов работы мозга сделает создание ИИ вполне осуществимой задачей. Имитируя процессы, происходящие в человеческом мозге в процессе обучения, мышления и принятия решений, мы можем создать машину, способную делать то же самое. Такая машина послужит платформой для создания систем, способных к обучению.

Зачем нужно изучать ИИ?

Успехи в создании ИИ способны повлиять на все аспекты нашей жизни. Исследования в этой области направлены на изучение свойств и закономерностей поведения объектов. С помощью ИИ мы хотим создавать умные системы и стремимся понять, как заставить машины выполнять творческие функции. Реализуя интеллектуальные системы, мы приближаемся к пониманию того, каким образом одни интеллектуальные системы, подобные нашему мозгу, способны справляться с созданием других.

Рис. 1.1 дает некоторое представление о процессе обработки информации нашим мозгом.

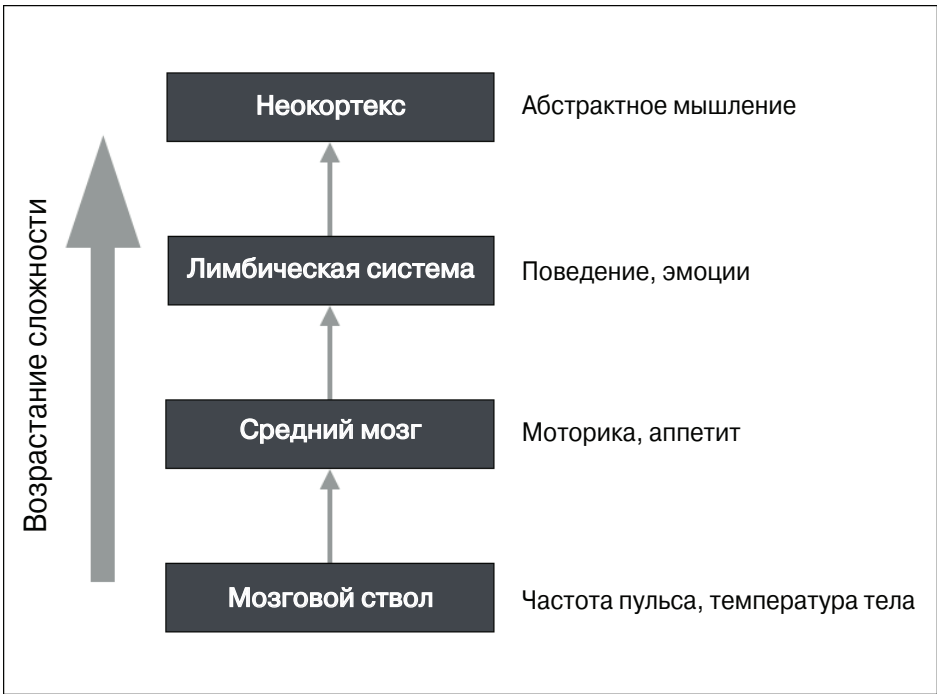

Рис. 1.1

По сравнению с другими областями науки, такими как математика или физика, которые существуют столетиями, наука об ИИ сравнительно молодая. За последние два десятилетия она продемонстрировала замечательные достижения, примерами которых могут служить беспилотные автомобили и интеллектуальные шагающие роботы. Уже достигнутые результаты делают довольно очевидным тот факт, что исследования в области ИИ способны коренным образом изменить нашу жизнь в ближайшие годы.

Можно лишь удивляться тому, как человеческий мозг справляется с обработкой огромных объемов информации с минимальными усилиями. Мы распознаем объекты, понимаем другие языки, учимся новому и выполняем множество разнообразных сложных задач. Каким образом нашему мозгу удается это делать? Пытаясь делать то же самое с помощью машин, мы видим, что они остаются далеко позади! Например, рассуждая о таких вещах, как внеземная жизнь или путешествия во времени, мы даже не уверены в том, могут ли они существовать на самом деле. Хорошая новость относительно Святого Грааля ИИ заключается в том, что нам достоверно известно о его существовании. Этим Святым Граалем является наш мозг! Он служит ярчайшим примером интеллектуальной системы. Нам нужно лишь сымитировать его функциональность для создания систем, способных делать нечто похожее, а возможно, даже большее.

Отдельные этапы преобразования исходных данных в знания можно условно представить в виде следующей схемы (рис. 1.2).

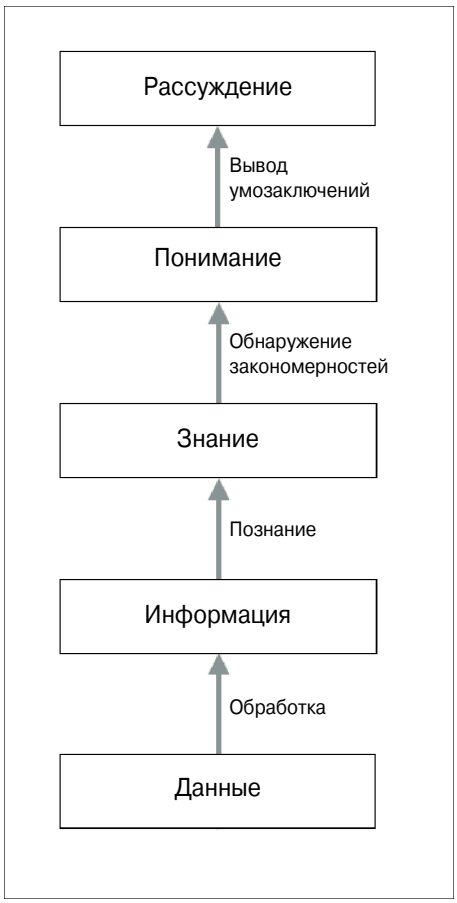

Рис. 1.2

Одна из основных причин нашего стремления к изучению ИИ — возможность автоматизации многих процессов. Мы живем в мире, в котором:

- приходится иметь дело с огромными объемами данных, с обработкой которых человеческий мозг просто не в состоянии справиться;
- данные поступают одновременно из множества источников;
- эти данные не структурированы и поступают хаотично;
- знания, полученные на основе этих данных, должны непрерывно обновляться, поскольку сами данные подвержены постоянным изменениям;
- восприятие данных и ответная реакция на них должны с высокой точностью осуществляться в режиме реального времени.

Несмотря на то что человеческий мозг проявляет замечательные способности в отношении анализа окружающей обстановки, его возможностей недостаточно для удовлетворения всех перечисленных выше условий. Следовательно, мы вынуждены изобретать и разрабатывать интеллектуальные системы, позволяющие преодолеть это ограничение. Нам нужны системы искусственного интеллекта, которые могли бы:

- эффективно обрабатывать большие объемы данных, хранение которых в настоящее время стало возможным благодаря облачным вычислениям;
- получать данные одновременно из нескольких источников без каких-либо задержек;
- индексировать и организовывать данные способами, обеспечивающими возможность их осмысления;
- обучаться на новых данных и постоянно обновлять получаемые знания, используя подходящие алгоритмы обучения;
- принимать решения и реагировать на изменяющиеся обстоятельства в режиме реального времени.

Методики ИИ активно используются для совершенствования существующих машин, чтобы они становились все умнее и могли быстрее и эффективнее выполнять возложенные на них функции.

Области применения ИИ

Теперь, когда вы уже знаете, как обрабатывается информация, мы можем перейти к рассмотрению применения ИИ в реальной жизни. ИИ проявляет себя в разных формах, поэтому очень важно понимать, чем именно он может быть полезен для той или иной сферы деятельности. ИИ интенсивно используется во многих областях, и круг его применений чрезвычайно быстро расширяется. Рассмотрим наиболее популярные из них.

- **Компьютерное зрение.** Разработаны системы, предназначенные для обработки таких визуальных данных, как изображения и видео. Такие системы анализируют содержание изображений и извлекают полезную информацию на основании предоставленных типовых образцов. Например, Google использует технологию *реверсивного (обратного) поиска изображений* для нахождения визуально подобных изображений в Интернете (рис. 1.3).
- **Обработка естественного языка.** Системы этого типа предназначены для распознавания текстов, написанных на естественных языках. Мы можем взаимодействовать с машиной, передавая ей команды в виде

текстовых предложений. Поисковые системы интенсивно используют эту технологию для доставки релевантных результатов поиска пользователям.

Рис. 1.3

- **Распознавание речи.** Эти системы способны воспринимать звуковую информацию и понимать произносимые слова. Например, наши смартфоны оборудованы интеллектуальными персональными помощниками, которые понимают голосовые команды и реагируют на них предоставлением соответствующей информации или выполнением запрошенных действий.
- **Экспертные системы.** В этих системах методики ИИ используются для принятия решений или предоставления соответствующих рекомендаций. Как правило, в таких областях, как финансы, медицина, маркетинг и др., для этой цели используют базы знаний. На рис. 1.4 иллюстрируется, что собой представляет экспертная система и как она взаимодействует с пользователем.
- **Игры.** ИИ широко применяется в индустрии игр. Он используется для проектирования интеллектуальных агентов, способных состязаться

в мастерстве игры с человеком. В качестве примера можно привести AlphaGo — компьютерную программу, которая умеет играть в стратегическую игру Go. ИИ также используется для проектирования игр другого типа, в которых от компьютера ожидается интеллектуальное поведение.

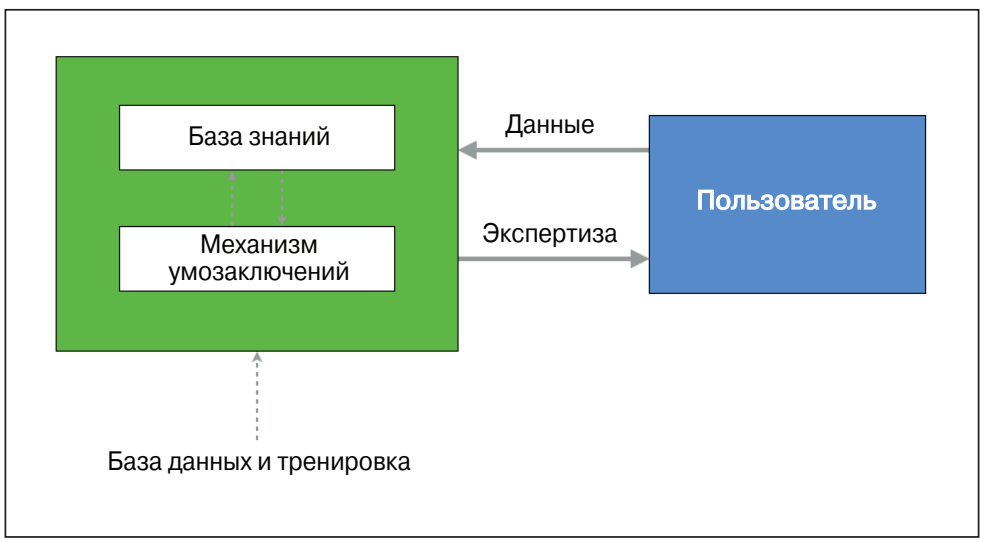

Рис. 1.4

- **Робототехника.** Робототехнические системы в действительности объединяют в себе многие концепции ИИ. Эти системы способны выполнять множество самых разнообразных задач. В зависимости от ситуации, роботы могут оборудоваться датчиками и приводными элементами, обеспечивающими выполнение всевозможных действий. Датчики могут распознавать предметы, находящиеся в поле их зрения, измерять их температуру, реагировать на выделяемое ими тепло или совершаемые ими движения и т.п. Вмонтированные в электронные платы процессоры выполняют необходимые расчеты в режиме реального времени. Кроме того, роботы могут адаптировать свое поведение к изменению внешних условий.

Направления исследований ИИ

Очень важно понимать суть различных направлений исследования ИИ, поскольку это позволит вам выбрать наиболее подходящую платформу для решения стоящей перед вами задачи. Ниже перечислены темы, которые доминируют в этой области.

- **Машинное обучение и распознавание образов.** Возможно, это наиболее популярное направление разработок в области ИИ. Мы проектируем и разрабатываем программы, способные учиться на предоставляемых им данных. На основании моделей обучения мы можем составлять прогнозы, касающиеся неизвестных данных. В этом отношении одним из основных ограничений является то обстоятельство, что эффективность подобных программ ограничивается мощностью данных. Принцип функционирования типичной системы машинного обучения иллюстрируется на рис. 1.5.

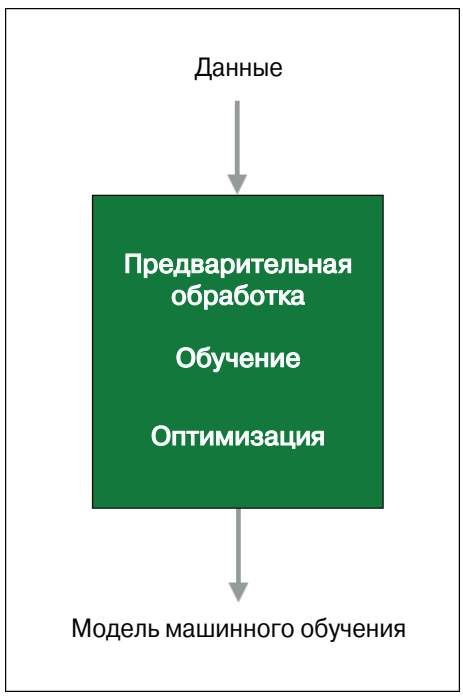

Рис. 1.5

Получая результаты наблюдения, система тренируется, сравнивая их с теми результатами, которые уже наблюдались в предоставленных ей примерах с известным ответом. Например, в случае системы распознавания лиц программа будет пытаться найти соответствие образцам глаз, носа, губ, бровей и т.п. с целью идентификации личности на основе соответствующих изображений, хранящихся в базе данных пользователей.

- ***Логический ИИ.*** Для выполнения программ в системах логического ИИ используются методы математической логики. Программа логического ИИ в основном представляет собой набор утверждений в

логической форме, которые выражают факты и правила, относящиеся к конкретной предметной области. Подобные подходы интенсивно используются для установления соответствия шаблонам, парсинга текста, семантического анализа, а также при решении ряда других задач.

- **Поиск.** В программах ИИ широко используются методы поиска. Такие программы исследуют большое количество всевозможных вариантов и выбирают наиболее оптимальный из них. Например, такой подход часто применяется во многих стратегических играх, в частности в шахматах, а также при управлении компьютерными сетями, распределении ресурсов, планировании и т.п.
- **Представление знаний.** Чтобы факты, относящиеся к окружающему нас миру, имели для системы смысл, они должны предоставляться ей в той или иной форме. Для этой цели часто используют языки математической логики. При удачно выбранной форме представления знаний система способна функционировать более интеллектуальным образом. Близкой к этому областью исследований является онтология, которая имеет дело с формализацией описания существующих видов объектов. Информационные онтологии дают формальные определения свойств объектов и отношений между ними, существующих в конкретной предметной области. Обычно это делается с привлечением определенной таксономии или некоторой иерархической структуры. Диаграмма, приведенная на рис. 1.6, поясняет различие между информацией и знаниями.

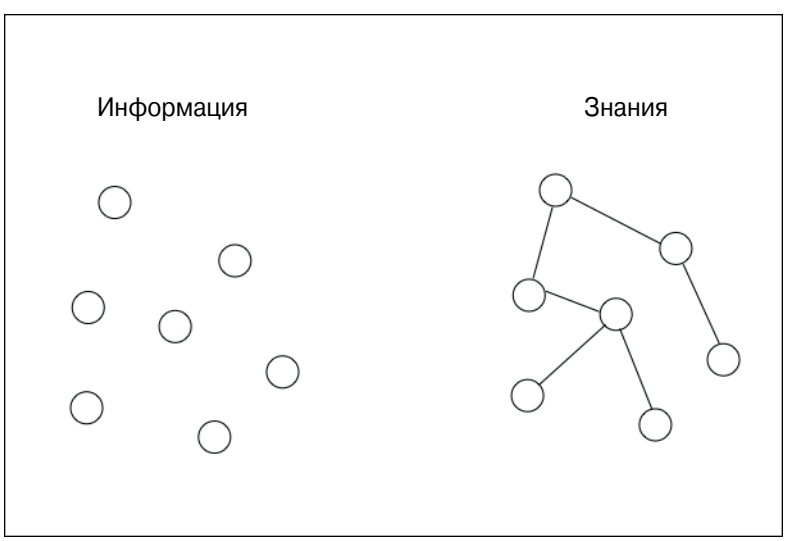

Рис. 1.6

- **Планирование.** Это направление связано с оптимальным планированием, обеспечивающим максимально возможную отдачу при минимальных затратах. В подобных программах исходными данными служат факты, относящиеся к конкретной ситуации, и формулировка цели. Таким программам также должны быть известны факты о внешнем мире, чтобы они ориентировались в том, какими правилами следует руководствоваться. На основании предоставленной информации программа генерирует наиболее оптимальный план, обеспечивающий достижение поставленной цели.
- **Эвристика.** Эвристика — это методология, использовать которую наиболее целесообразно в тех случаях, когда решение задачи должно быть найдено в кратчайшие сроки, но при этом не требуется, чтобы полученное решение было оптимальным. Здесь речь идет скорее о выдвижении правдоподобной гипотезы относительно подхода, который должен быть предпринят для решения задачи. В исследованиях ИИ часто встречаются ситуации, когда мы не можем проверить каждую из имеющихся возможностей, чтобы затем выбрать наилучший вариант. Именно в таких случаях и приходится прибегать к эвристическим методам для достижения своих целей. Эти методы широко используются в таких областях ИИ, как робототехника, поисковые системы и т.п.
- **Генетическое программирование.** Это автоматический подбор программы для решения определенной задачи путем объединения возможностей программ и выбора их комбинированного варианта, наиболее пригодного для данной задачи. Код таких программ пишется в виде набора генов с использованием алгоритма, который обеспечивает получение программы, способной найти действительно эффективное решение поставленной задачи.

Оценка уровня искусственного интеллекта с помощью теста Тьюринга

Легендарный математик и компьютерный ученый Алан Тьюринг предложил тест, названный в его честь *тестом Тьюринга*, который позволяет определить уровень "интеллектуальности" машины. С помощью этого теста можно проверить, способен ли компьютер имитировать поведение мыслящих существ. Тьюринг определил как *разумное* такое поведение компьютера, которое в процессе "беседы" с ним невозможно отличить от поведения человека. Этого было бы достаточно для того, чтобы убедить интервьюера в том, что ответы на поставленные им вопросы даются человеком.

Чтобы проверить, способны ли на подобное машины, Тьюринг предложил следующую организацию теста: адресованные машине вопросы должны задаваться человеком посредством текстового интерфейса. В качестве еще одного ограничения он указал, что человек, задающий вопросы, не должен знать, кто именно выступает в качестве его собеседника — человек или машина, т.е. предполагается, что собеседником может быть как машина, так и человек. Чтобы реализовать описанную схему теста, человек должен взаимодействовать с обоими объектами посредством текстового интерфейса. Эти два объекта называются *респондентами*. Одним из них должен быть человек, другим — машина.

Машина-респондент проходит тест, если интервьюеру не удается определить, от кого исходили ответы — от машины или от человека. Схема организации теста Тьюринга представлена на рис. 1.7.

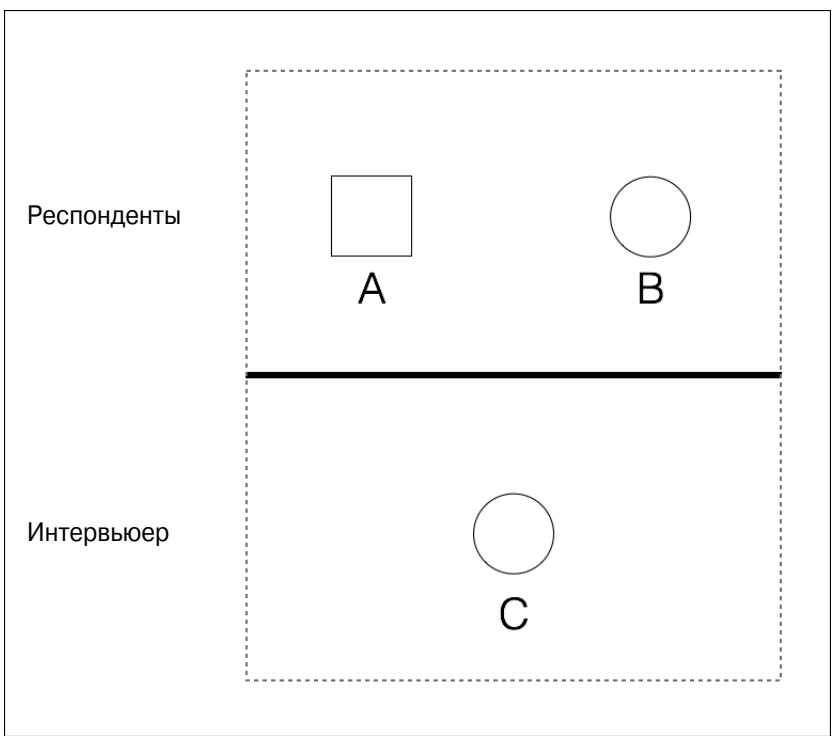

Рис. 1.7

Как нетрудно догадаться, для машины-респондента это задание является довольно трудным. В процессе беседы происходит очень многое. Как минимум машина должна хорошо справляться со следующими задачами.

- **Обработка естественного языка.** Это необходимо машинам для того, чтобы они могли общаться с интервьюером. Машина должна уметь выполнять синтаксический анализ предложений, определять контекст и адекватно отвечать на вопросы.
- **Представление знаний.** Машина нуждается в сохранении информации, предоставленной ей до проведения диалога. Она также должна отслеживать информацию, полученную в процессе диалога, чтобы использовать ее повторно, если в этом возникнет необходимость.
- **Вывод умозаключений.** Очень важно, чтобы машина понимала, каким образом следует интерпретировать сохраненную информацию. Обычно у людей это происходит автоматически, что позволяет им выводить умозаключения в режиме реального времени.
- **Машинное обучение.** Это необходимо для того, чтобы машина могла приспосабливаться к новым условиям в режиме реального времени. Машина должна анализировать и обнаруживать закономерности, чтобы выводить соответствующие умозаключения.

Вы, должно быть, задумались над тем, почему человек, участвующий в тесте, должен использовать текстовый интерфейс в процессе общения. Согласно Тьюрингу, физическая имитация человека является излишней для данного теста. В этом и заключается причина, по которой тест Тьюринга проводится без прямого физического контакта между человеком и машиной. Существует также полный тест Тьюринга, включающий эффекты зрения и движения. Чтобы пройти такой тест, машина должна видеть объекты, используя машинное зрение, и перемещаться, используя робототехнику.

Как научить машины мыслить подобно людям

На протяжении десятилетий мы пытаемся создать машину, которая могла бы мыслить, как человек. Для этого нам прежде всего нужно понять, как мыслят люди. Как мы должны действовать, чтобы понять природу человеческого мышления? Например, можно было бы вести записи, фиксирующие нашу реакцию на то, что происходит вокруг. Но этот способ очень быстро обнаружит свою бесперспективность из-за огромного количества необходимых записей. Другой подход основан на проведении экспериментов предопределенного формата. Мы разрабатываем серию вопросов, которые охватывают широкий спектр тем, имеющих непосредственное отношение к человеку, а затем анализируем, как люди на них отвечают.

Как только будет собран достаточно большой объем данных, мы сможем построить модель, имитирующую поведение человека. Далее эту модель можно будет использовать для создания программного обеспечения, способного думать, как люди. Конечно, легко сказать, да трудно сделать! Все, что нас интересует, — это выходная информация, предоставляемая программой в ответ на входную информацию. Если поведение программы согласуется с поведением человека, то мы можем сказать, что мышление человека описывается аналогичным механизмом.

Человеческое мышление можно условно разбить на несколько уровней, которые представлены на рис. 1.8, иллюстрирующем их иерархическую структуру.

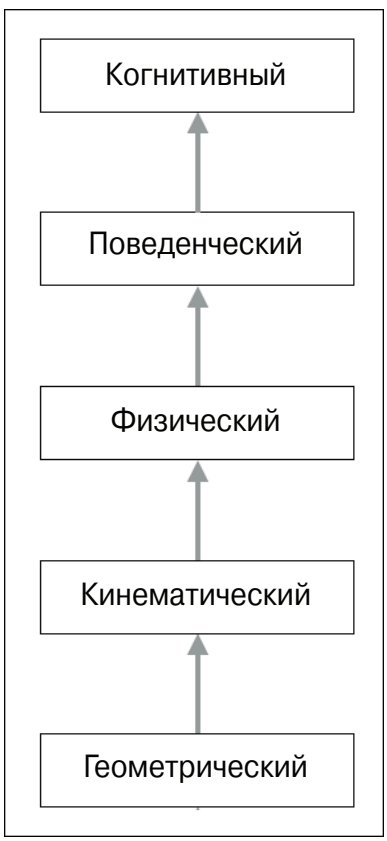

Рис. 1.8

Существует целый раздел компьютерной науки — *когнитивное моделирование*, которое занимается имитацией мыслительного процесса человека. Его целью является достижение понимания того, как люди решают задачи. Специалисты по когнитивному моделированию анализируют, какие

умственные процессы оказываются вовлеченными в поиск решения, и на основании этого строят программную модель. Впоследствии эта модель применяется для имитации поведения человека. Когнитивное моделирование задействуется во многих приложениях ИИ, таких как глубокое обучение, экспертные системы, обработка естественного языка, робототехника и др.

Создание рациональных агентов

Многие исследования в области ИИ нацелены на создание *рациональных агентов*. Что такое рациональный агент? Прежде чем обсудить это понятие, определим смысл термина "рациональность". *Рациональность* — это выполнение действий, которые соответствуют данным конкретным обстоятельствам. Действия должны выполняться таким образом, чтобы принести максимальную выгоду выполняющему их объекту. Говорят, что агент действует рационально, если он, придерживаясь установленного набора правил, выполняет действия, направленные на достижение поставленной цели. Он принимает решения и действует исключительно в соответствии с имеющейся информацией. Подобные системы широко применяются для проектирования роботов, предназначенных для выполнения различных заданий в незнакомой местности.

Но что такое *соответствующие действия*? Ответ зависит от того, какие задачи поставлены перед агентом. Предполагается, что агент действует интеллектуально, без вмешательства человека. Мы хотим наделить агента способностью приспосабливаться к новым ситуациям. Он должен адекватно оценивать окружающую обстановку и действовать так, чтобы добиться наилучших с его точки зрения результатов. Наилучшие результаты диктуются общей целью, которую он пытается достигнуть. На рис. 1.9 представлена условная схема того, как входная информация преобразуется в действия.

А как нам измерить эффективность действий рационального агента? Кто-то скажет, что такой мерой могла бы служить степень достижения успеха. Агент настраивается на решение определенной задачи, поэтому оценка эффективности зависит от того, какой процент задачи выполнен. Но мы должны задуматься над тем, что именно составляет сущность рационализма во всей полноте этого понятия. Если говорить только о результатах, то может ли агент предпринять какие-либо действия для получения данного результата?

Несомненно, что способность делать правильные умозаключения является необходимым компонентом рационального поведения, поскольку агент должен действовать рационально для достижения своих целей. Это свойство позволит ему приходить к правильным выводам, которыми можно последовательно руководствоваться. А что можно сказать о ситуациях, в которых

отсутствует возможность предпринимать доказуемо правильные действия? Бывают такие ситуации, в которых агент не знает, что ему делать, но в то же время он должен как-то действовать. В подобных случаях мы не может использовать концепцию логических выводов для определения рационального поведения.

Рис. 1.9

Универсальный решатель задач

Универсальный решатель задач (General Problem Solver — GPS) — программа ИИ, предложенная Гербертом Саймоном, Джоном Клиффордом Шоу и Алленом Ньюэллом. Это была первая компьютерная программа, появившаяся в мире ИИ. Она разрабатывалась для создания универсальной машины, способной решать задачи. Разумеется, к тому времени существовало множество компьютерных программ, но все они выполняли вполне определенные задачи. GPS была первой программой, предназначенной для решения задач произвольного типа. По замыслу ее авторов для решения любой задачи программа должна была использовать один и тот же алгоритм.

Совершенно очевидно, что реализовать эту амбициозную цель оказалось не так-то просто. Для этого авторам программы потребовалось создать новый язык — IPL (Information Processing Language — язык обработки информации). Работа программы базировалась на возможности формулирования любой задачи с помощью набора формул. Эти формулы становились частью направленного графа, имеющего множество источников и стоков. Применительно к графам термин *источник* относится к начальному узлу, а термин *сток* — к конечному. В случае GPS источниками служат аксиомы, а стоками — логические выводы.

Хотя GPS задумывалась в качестве универсального средства, она могла решать лишь такие хорошо определенные задачи, как доказательство геометрических и логических теорем. Она также могла решать головоломки и играть в шахматы. Это обусловлено тем, что подобные задачи могли быть в достаточной степени формализованы. Но в случае реальных задач формализация очень быстро становится трудноразрешимой проблемой из-за наличия большого количества возможных дальнейших шагов, которые приходится учитывать. При попытке решения задачи методом грубой силы, т.е. путем подсчета количества путей в графе, вычисление решения становится практически нереализуемым.

Решение задач с помощью GPS

Рассмотрим структурирование конкретной задачи для получения ее решения с помощью GPS.

1. Первый шаг состоит в определении целей. Пусть нашей целью будет покупка молока в магазине.
2. Следующим шагом является определение предусловий. Эти предусловия связаны с целями. Чтобы доставить молоко из магазина, оно должно быть в нем в наличии и мы должны располагать средством его транспортировки.
3. После этого мы должны определить операторы. Если средством транспортировки является автомобиль, который нуждается в дозаправке горючим, то мы должны быть уверены в том, что сможем заплатить нужную сумму на автозаправке. Мы также должны убедиться в том, что у нас останется достаточно денег для того, чтобы заплатить за молоко в магазине.

Оператор заботится об условиях и обо всем, что на них влияет. Он состоит из действий, предусловий и изменений, являющихся результатом предпринимаемых действий. В данном случае действием является уплата денег магазину.

Конечно же, возможность совершения этого действия зависит в первую очередь от наличия денег, что служит предусловием. Выплачивая магазину деньги, вы изменяете их состояние, в результате чего получаете молоко.

GPS будет работать только в том случае, если вам удастся структурировать задачу, как мы это только что сделали. Основным ограничением является необходимость проведения трудоемкой процедуры поиска, без которой GPS не может обойтись в процессе выполнения своей работы и которая отнимает много времени при решении любой реальной задачи.

Создание интеллектуальных агентов

Существуют разные способы создания интеллектуальных агентов. К числу наиболее распространенных из них относятся машинное обучение, базы знаний, наборы правил и др. В этом разделе мы ограничимся подходом, основанным на машинном обучении. Данный метод предполагает наделение агента интеллектуальными способностями посредством тренировки (обучения) на известных данных.

На рис. 1.10 представлена общая схема взаимодействия интеллектуального агента с окружением.

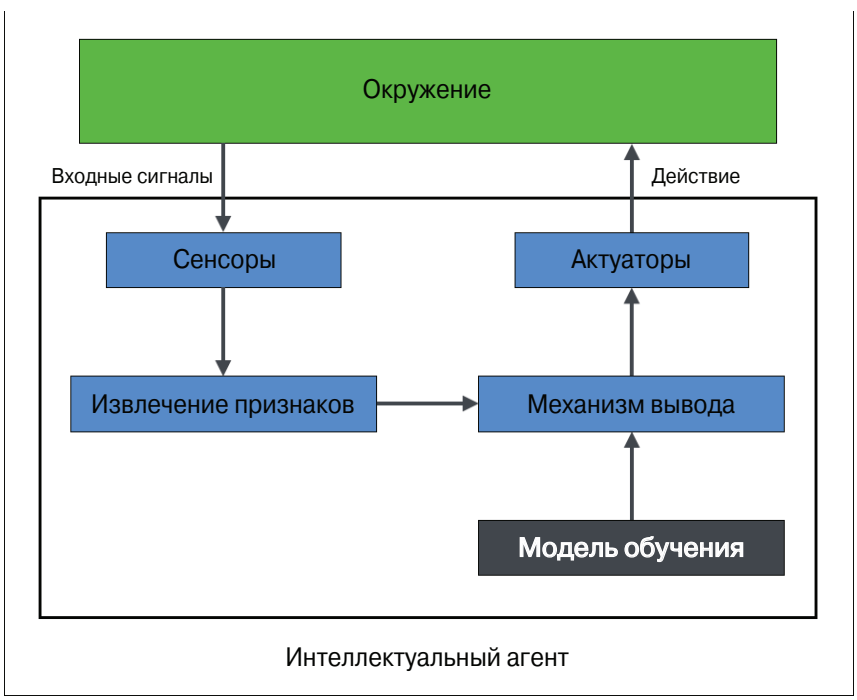

Рис. 1.10

Применяя машинное обучение, мы стремимся программировать машины так, чтобы они могли использовать маркированные данные для решения любой конкретной задачи. Прогоняя через машину данные и ассоциированные с ними маркеры, мы учим ее распознавать образы и отношения между ними.

В предыдущем примере работа интеллектуального агента зависит от модели обучения, используемой для работы механизма логического вывода. Получив входной сигнал, сенсор посылает его в блок извлечения признаков. После извлечения соответствующих признаков тренируемый механизм вывода строит соответствующий прогноз на основании модели обучения. Эта модель обучения создается с использованием машинного обучения. Затем механизм вывода принимает решение и передает его актуатору, который выполняет требуемое действие в реальном окружении.

В настоящее время машинное обучение находит множество применений. Оно используется для распознавания образов и речи, прогнозирования рыночных тенденций, в робототехнике и других областях. Чтобы понять сущность машинного обучения и научиться создавать полные решения, вам потребуется знакомство со множеством технологий, таких как распознавание образов, искусственные нейронные сети, добыча данных, статистика и т.п.

Типы моделей

В мире ИИ существует два типа моделей: аналитические и обучаемые. До появления машин, способных выполнять необходимые вычисления, обычно использовались *аналитические модели*. Они основывались на математических формулировках, представляющих собой, по сути, описание последовательных шагов, которые требовалось выполнить для получения окончательного уравнения. Проблемой этого подхода является то, что он зависит от суждений человека. Как следствие, подобные модели были упрощенными и страдали неточностью, обусловленной недостаточно большим количеством параметров.

Затем наступила эра компьютеров. Компьютеры хорошо справлялись с анализом данных. Поэтому со временем все шире стали использоваться *обучаемые модели*. Такие модели создаются посредством тренировки. Для получения уравнения в процессе тренировки машины просматривают множество примеров входных и соответствующих выходных данных. Подобным обучаемым моделям свойственны сложность и высокая точность, поскольку в них учитываются тысячи параметров. Это приводит к тому, что результирующее уравнение, управляющее данными, оказывается чрезвычайно сложным.

Методы машинного обучения позволяют получать такие обучаемые модели, которые могут быть использованы в механизме вывода. Наиболее благоприятным для нас следствием этого факта является то, что в данном случае

мы избавлены от необходимости выводить базовые математические формулы. От нас не требуется владение сложным математическим аппаратом, поскольку машина извлекает эти формулы на основании данных. Все, что мы должны сделать, — предоставить соответствующие списки входных и выходных значений. Обученная модель, которую мы при этом получаем, всего лишь отражает отношения между маркированными входными и желаемыми выходными значениями.

Установка Python 3

Все примеры, приведенные в книге, были получены с использованием Python 3. Убедитесь в том, что в вашей системе установлена последняя версия Python 3. Чтобы это проверить, введите в окне терминала следующую команду:

```
$ python3 --version
```

Если отобразится нечто вроде `Python 3.x.x` (где `x.x` — номера версии), то устанавливать Python не потребуется. В противном случае выполнить процедуру установки будет совсем несложно.

Установка в Ubuntu

В версиях Ubuntu 14.*xx* и выше Python 3 установлен по умолчанию. Если это не так, выполните установку с помощью следующей команды:

```
$ sudo apt-get install python3
```

После этого выполните вышеупомянутую проверку версии.

```
$ python3 --version
```

Номер установленной версии отобразится в окне вашего терминала.

Установка в Mac OS X

Если вы работаете в системе Mac OS X, то для установки Python 3 рекомендуется использовать пакет Homebrew. Это замечательный пакет-установщик, предназначенный для Mac OS X, которым действительно легко пользоваться. В случае отсутствия этого пакета вы сможете установить его с помощью следующей команды:

```
$ ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

После этого можно обновить менеджер пакетов.

```
$ brew update
```

Теперь можно установить Python 3.

```
$ brew install python3
```

Выполните вышеупомянутую проверку версии.

```
$ python3 --version
```

Номер установленной версии отобразится в окне вашего терминала.

Установка в Windows

Если вы работаете в Windows, то рекомендуется использовать дистрибутив Python 3, соответствующий спецификации SciPy-stack. В этом отношении весьма популярен и легок в использовании дистрибутив Anaconda. Соответствующие инструкции по его установке вы найдете по адресу `https://www.continuum.io/downloads`.

Если вы хотите ознакомиться с возможностями других дистрибутивов Python 3, совместимых со спецификацией SciPy-stack, посетите веб-страницу `http://www.scipy.org/install.html`. В этих дистрибутивах хорошо то, что они поставляются со всеми необходимыми предустановленными пакетами. При использовании любой из этих версий вам не придется устанавливать пакеты по отдельности.

Завершив установку, выполните вышеупомянутую команду проверки.

```
$ python3 --version
```

Номер установленной версии отобразится в окне вашего терминала.

Установка пакетов

На протяжении всей книги мы будем использовать различные пакеты, такие как NumPy, SciPy, *scikit-learn* и *matplotlib*. Убедитесь в том, что эти пакеты установлены в вашей системе, прежде чем продолжить чтение.

Если вы используете Ubuntu или Mac OS X, то установка указанных пакетов не составит труда. Каждый из них устанавливается с помощью команды, умещающейся в одной строке в окне терминала. Соответствующие ссылки, касающиеся установки, приведены ниже.

- **NumPy**
  `http://docs.scipy.org/doc/numpy-1.10.1/user/install.html`
- **SciPy**
  `http://www.scipy.org/install.html`

- **scikit-learn**
  `http://scikit-learn.org/stable/install.html`
- **matplotlib**
  `http://matplotlib.org/1.4.2/users/installing.html`

Если вы работаете в Windows, то вам достаточно установить версию Python 3, соответствующую спецификации SciPy-stack.

Загрузка данных

Для построения модели обучения нам нужны данные, представляющие внешний мир. Теперь, когда у нас установлены необходимые пакеты Python, рассмотрим, как использовать их для взаимодействия с данными. Перейдем к командной строке Python, введя в окне терминала следующую команду:

```
$ python3
```

Импортируем пакет, содержащий все наборы данных.

```
>>> from sklearn import datasets
```

Загрузим набор данных с ценами на недвижимость.

```
>>> house_prices = datasets.load_boston()
```

Выведем данные.

```
>>> print(house_prices.data)
```

В окне терминала должен отобразиться следующий вывод (рис. 1.11).

```
>>> print(house_prices.data)
[[  6.32000000e-03   1.80000000e+01   2.31000000e+00 ...,   1.53000000e+01
    3.96900000e+02   4.98000000e+00]
 [  2.73100000e-02   0.00000000e+00   7.07000000e+00 ...,   1.78000000e+01
    3.96900000e+02   9.14000000e+00]
 [  2.72900000e-02   0.00000000e+00   7.07000000e+00 ...,   1.78000000e+01
    3.92830000e+02   4.03000000e+00]
 ...,
 [  6.07600000e-02   0.00000000e+00   1.19300000e+01 ...,   2.10000000e+01
    3.96900000e+02   5.64000000e+00]
 [  1.09590000e-01   0.00000000e+00   1.19300000e+01 ...,   2.10000000e+01
    3.93450000e+02   6.48000000e+00]
 [  4.74100000e-02   0.00000000e+00   1.19300000e+01 ...,   2.10000000e+01
    3.96900000e+02   7.88000000e+00]]
```

Рис. 1.11

Проверим маркерные значения.

```
>>> print(house_prices.target)
```

В окне терминала должен отобразиться следующий вывод (рис. 1.12).

```
>>> print(house_prices.target)
[ 24.   21.6  34.7  33.4  36.2  28.7  22.9  27.1  16.5  18.9  15.   18.9
  21.7  20.4  18.2  19.9  23.1  17.5  20.2  18.2  13.6  19.6  15.2  14.5
  15.6  13.9  16.6  14.8  18.4  21.   12.7  14.5  13.2  13.1  13.5  18.9
  20.   21.   24.7  30.8  34.9  26.6  25.3  24.7  21.2  19.3  20.   16.6
  14.4  19.4  19.7  20.5  25.   23.4  18.9  35.4  24.7  31.6  23.3  19.6
  18.7  16.   22.2  25.   33.   23.5  19.4  22.   17.4  20.9  24.2  21.7
  22.8  23.4  24.1  21.4  20.   20.8  21.2  20.3  28.   23.9  24.8  22.9
  23.9  26.6  22.5  22.2  23.6  28.7  22.6  22.   22.9  25.   20.6  28.4
  21.4  38.7  43.8  33.2  27.5  26.5  18.6  19.3  20.1  19.5  19.5  20.4
  19.8  19.4  21.7  22.8  18.8  18.7  18.5  18.3  21.2  19.2  20.4  19.3
  22.   20.3  20.5  17.3  18.8  21.4  15.7  16.2  18.   14.3  19.2  19.6
  23.   18.4  15.6  18.1  17.4  17.1  13.3  17.8  14.   14.4  13.4  15.6
  11.8  13.8  15.6  14.6  17.8  15.4  21.5  19.6  15.3  19.4  17.   15.6
  13.1  41.3  24.3  23.3  27.   50.   50.   50.   22.7  25.   50.   23.8
  23.8  22.3  17.4  19.1  23.1  23.6  22.6  29.4  23.2  24.6  29.9  37.2
  39.8  36.2  37.9  32.5  26.4  29.6  50.   32.   29.8  34.9  37.   30.5
  36.4  31.1  29.1  50.   33.3  30.3  34.6  34.9  32.9  24.1  42.3  48.5
  50.   22.6  24.4  22.5  24.4  20.   21.7  19.3  22.4  28.1  23.7  25.
  23.3  28.7  21.5  23.   26.7  21.7  27.5  30.1  44.8  50.   37.6  31.6
  46.7  31.5  24.3  31.7  41.7  48.3  29.   24.   25.1  31.5  23.7  23.3
```

Рис. 1.12

Размер фактического массива больше, поэтому на данном экранном снимке представлена лишь часть его значений.

В пакете `sklearn` также доступны наборы графических данных. Каждый набор представлен матрицей размером 8×8. Загрузим эти изображения.

```
>>> digits = datasets.load_digits()
```

Выведем пятое изображение.

```
>>> print(digits.images[4])
```

В окне терминала отобразится следующий вывод (рис. 1.13).

Нетрудно заметить, что эти данные располагаются в восьми строках и восьми столбцах.

```
>>> print(digits.images[4])
[[  0.   0.   0.   1.  11.   0.   0.   0.]
 [  0.   0.   0.   7.   8.   0.   0.   0.]
 [  0.   0.   1.  13.   6.   2.   2.   0.]
 [  0.   0.   7.  15.   0.   9.   8.   0.]
 [  0.   5.  16.  10.   0.  16.   6.   0.]
 [  0.   4.  15.  16.  13.  16.   1.   0.]
 [  0.   0.   0.   3.  15.  10.   0.   0.]
 [  0.   0.   0.   2.  16.   4.   0.   0.]]
```

Рис. 1.13

Резюме

Из этой главы вы узнали, что такое ИИ и зачем его нужно изучать. Мы обсудили различные применения и подвиды ИИ. Мы рассмотрели тест Тьюринга и показали, как его можно выполнять. Вы узнали о том, как заставить машины рассуждать подобно людям. Мы обсудили понятие рационального агента и рассказали о способах его проектирования. Вы также узнали, что такое универсальный решатель задач (GPS) и как решать задачи с помощью GPS. Мы затронули тему разработки интеллектуальных агентов с помощью машинного обучения, указав при этом на существование различных типов моделей.

Кроме того, мы рассмотрели процесс установки Python 3 в различных операционных системах. Вы узнали о том, как установить пакеты, необходимые для создания приложений ИИ, и использовать их для загрузки данных, доступных в библиотеке `scikit-learn`. В следующей главе вы познакомитесь с методами обучения с учителем и узнаете о том, как создавать модели, предназначенные для решения задач классификации и регрессии.