Теоретически между теорией и практикой
нет никакой разницы. Но на практике она есть.
– Бенджамин Брюстер

Идеальный план проекта –
сначала составить список всех неизвестных.
– Билл Лэнгли

Привлечение средств – это искусственный интеллект.
Найм на работу – машинное обучение.
Реализация – это линейная регрессия.
А отладка – это printf().
– Бэрон Шварц

Содержание

От издательства

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательство «ДМК Пресс» очень серьезно относится к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Вступительное слово

Хочу открыть вам один секрет: когда говорят «машинное обучение», складывается впечатление, будто речь идет только об одной дисциплине. А вот и нет! На самом деле есть два вида машинного обучения, и они различаются так же, как придумывание новых блюд и изобретение кухонных принадлежностей. То и другое занятия достойны уважения, но путать их не надо: не станете же вы поручать шеф-повару разработку кухонной плиты, а инженеру-электрику – печь хлеб!

Увы, почти все путают эти два вида машинного обучения. Поэтому неудивительно, что так много компаний терпят крах на этапе машинного обучения. Начинающим, похоже, никто не объяснил, что все то, что преподают на курсах машинного обучения и пишут в учебниках, – это теоретическое машинное обучение: как конструировать кухонные плиты (а также микроволновки, блендеры, тостеры, чайники … и мойку, в которой все смешивается!) с нуля, а не как придумывать новые блюда и готовить на огромную компанию. Иными словами, если ваша цель – создавать инновационные решения конкретных задач на базе МО, то вам нужно прикладное, а не теоретическое машинное обучение. Поэтому большинство книг – не для вас.

А теперь хорошая новость! Перед вами первая книга, посвященная именно прикладному машинному обучению. Да, вы ее нашли! Настоящая прикладная иголка в стоге теоретического сена. Поздравляю, любезный читатель… если, конечно, вы не искали книгу, которая помогла бы отточить навыки проектирования алгоритмов общего назначения. Но тогда я надеюсь, что автор не будет на меня в обиде за то, что я посоветую вам купить чуть ли не любую другую книгу по МО. А эта от них отличается.

В 2016 году, разрабатывая курс Google по прикладному МО «Как подружиться с машинным обучением», полюбившийся нашим инженерам и руководителям групп – а их больше десяти тысяч, – я придала ему примерно такую же структуру, как в этой книге. А все потому, что действовать в правильном порядке очень важно в прикладных дисциплинах. Попытка выполнить определенные шаги, прежде чем будут завершены другие, может иметь печальные последствия: от зря потраченного времени до феерического краха проекта. На самом деле сходство оглавления этой книги и моего курса как раз и побудило меня прочитать ее. И в полном согласии с теорией параллельной эволюции я узрела в авторе единомышленника, терзаемого по ночам отсутствием ресурсов по прикладному машинному обучению, одной из потенциально самых полезных и вместе с тем самых недопонятых отраслей инженерной практики, – терзаемого с такой силой, что он захотел как-то исправить ситуацию. Так что если вы вознамерились захлопнуть книгу, то, пожалуйста, сделайте мне одолжение – задумайтесь на минутку, почему главы следуют именно в таком порядке. Обещаю, уже это принесет полезные плоды.

Так что же будет в книге дальше? Машинное обучение можно сравнить с руководством по приготовлению еды в массовом масштабе. Поскольку вы

еще не прочитали книгу, опишу содержание книги в кулинарных терминах. Вам нужно определить, что имеет смысл приготовить и каковы цели (*принятие решений и управление продуктом*), понять, кто является поставщиками и клиентами (*знакомство с предметной областью и деловое чутье*), как обрабатывать ингредиенты в большом количестве (*инженерия и анализ данных*), как быстро оценивать много разных сочетаний ингредиентов, пригодных для создания потенциальных рецептов блюд (*инженерная разработка прототипа*), как проверить качество рецепта (*статистика*), как эффективно превратить потенциальный рецепт в миллионы порций (*организация производства*) и как гарантировать, что блюда останутся высококачественными, даже если грузовик привез тонну картошки вместо заказанного риса (*обеспечение надежности*). Эта книга – одна из немногих, где рассматриваются все этапы технологического процесса.

Теперь самое время подпустить немного дегтя в бочку меда. Это хорошая книга. Правда. Но не идеальная. Иногда автор срезает углы – как часто делают инженеры, профессионально занимающиеся машинным обучением, – хотя в целом все изложено правильно. А поскольку предмет книги быстро эволюционирует, автор и не пытается всегда быть на переднем крае. Но даже в отсутствие совершенства книгу все равно стоит прочитать. Учитывая, как мало на рынке подробных руководств по прикладному машинному обучению, ясное и последовательное введение в эту тему дорогого стоит. Я очень рада, что такая книга появилась!

Что мне в ней очень нравится, так это полнота, с которой излагается самая важная вещь, которую нужно знать о машинном обучении: ошибки возможны… и иногда болезненные ошибки. Как любят говорить мои коллеги, занимающиеся надежностью: «Надежда на лучшее – это не стратегия». Надеяться, что ошибок не будет, – худший из возможных подходов. Эта книга устроена иначе – и лучше. Она быстро заставляет расстаться с ложным чувством безопасности, которое вы могли лелеять, надеясь построить систему ИИ, более «умную», чем вы сами. (Забудьте об этом, ничего не получится.) Затем автор становится вашим заботливым проводником по пути, на котором можно наделать самых разнообразных ошибок, и объясняет, как их предотвратить, обнаружить и исправить. В книге прекрасно подчеркнута важность мониторинга, описаны подходы к сопровождению модели, рассказано, что делать, если что-то поломается, как спланировать резервные стратегии на случай разного рода отказов и как управлять ожиданиями пользователей (есть также раздел о том, что делать, если пользователями являются машины). Эти вопросы очень важны при практическом применении машинного обучения, но в других книгах для них часто не находится места. Впрочем, эта книга не из их числа.

Если вы собираетесь использовать машинное обучение для решения крупномасштабных практических задач, могу только порадоваться, что вы наткнулись на эту книгу. Читайте с удовольствием!

Кэсси Козырков,
главный специалист по теории принятия решений в Google,
автор курса «MakingFriends with Machine Learning»
на облачной платформе Google

Сентябрь 2020

Предисловие

За последние несколько лет машинное обучение (МО) для многих стало синонимом искусственного интеллекта. И хотя машинное обучение как научная дисциплина существует уже несколько десятков лет, в мире найдется лишь горстка организаций, в полной мере осознавших его потенциал.

Несмотря на доступность современных библиотек, пакетов и каркасов машинного обучения с открытым исходным кодом, которые поддерживаются ведущими организациями и широким сообществом ученых и программистов, большинство компаний все еще испытывают трудности с применением машинного обучения к решению практических деловых задач.

Одна из проблем – нехватка кадров. Но, даже располагая талантливыми специалистами по машинному обучению и анализу данных, в 2020 году большинство организаций все еще тратят от 31 до 90 дней на развертывание всего одной модели, а 18 % – более 90 дней, у некоторых на доведение идеи до продукта уходит даже больше года. Основные проблемы, с которыми приходится сталкиваться при освоении потенциала МО, например управление версиями модели, воспроизводимость результатов и масштабирование, имеют скорее инженерную, чем научную природу.

Существует много книг по машинному обучению – как теоретических, так и практических. Из типичного учебника вы можете узнать о разных типах машинного обучения, об основных семействах алгоритмов, о том, как они работают и как с их помощью создавать модели из данных.

В типичном учебнике меньше внимания уделяется инженерным аспектам реализации проектов машинного обучения. Такие вопросы, как сбор, хранение и предобработка данных, конструирование признаков, а также тестирование и отладка моделей, развертывание и вывод из эксплуатации, сопровождение на этапе выполнения и в процессе эксплуатации, зачастую остаются за кадром.

Эта книга призвана заполнить пробел.

На кого рассчитана эта книга

Я предполагаю, что читатель знаком с основами машинного обучения и способен построить модель при наличии подходящим образом отформатированного набора данных, применяя свой любимый язык программирования или библиотеку. Если вы неуверенно владеете применением алгоритмов машинного обучения к данным и не видите разницы между логистической регрессией, методом опорных векторов и случайным лесом, то я рекомендую предварительно прочитать мою книгу «The Hundred-Page Machine Learning Book»[1].

---

[1] *Бурков А.* Машинное обучение без лишних слов. СПб.: Питер, 2020.

Целевая аудитория этой книги – аналитики данных, стремящиеся стать инженерами по машинному обучению, инженеры по машинному обучению, стремящиеся привнести больше порядка в свою работу, студенты, изучающие машинное обучение, а также архитекторы программных систем, которым приходится иметь дело с моделями, разработанными аналитиками данных и инженерами по машинному обучению.

Как использовать эту книгу

Эта книга представляет собой подробный обзор передовых практик и паттернов проектирования в области прикладного машинного обучения. Я рекомендую читать ее с начала до конца. Но можете читать главы в любом порядке, поскольку в них рассматриваются различные аспекты жизненного цикла проекта с использованием машинного обучения и прямых зависимостей между главами нет.

Андрей Бурков

Глава 1

Введение

Хотя предполагается, что читатель этой книги знаком с основами машинного обучения, все же важно начать с определений, чтобы у нас было общее понимание используемых в книге терминов.

Ниже я повторяю некоторые определения из главы 2 книги «Машинное обучение без лишних слов», а также даю несколько новых определений. Если вы читали мою первую книгу, то некоторые части этой главы могут показаться вам знакомыми.

После этой главы мы будем одинаково понимать такие понятия, как обучение с учителем и без учителя. У нас будет общий взгляд на прямо и косвенно используемые данные, первичные и аккуратные данные, обучающие и зарезервированные данные.

Мы будем знать о том, когда следует использовать машинное обучение, а когда этого делать не стоит, а также о различных формах машинного обучения, например: на основе модели и на основе экземпляров, глубокое и поверхностное, классификация и регрессия и т. д.

Наконец, мы определим предмет инженерии машинного обучения и представим жизненный цикл проекта машинного обучения.

1.1. Обозначения и определения

Начнем с базовых математических обозначений и определим термины и понятия, к которым часто будем обращаться в этой книге.

1.1.1. Структуры данных

Скаляром[1] называется простое числовое значение, например 15 или –3.25. Переменные или константы, принимающие скалярные значения, обозначаются курсивом, например *x* или *a*.

Вектором называется упорядоченный список скалярных значений, именуемых атрибутами. Мы обозначаем вектор полужирным шрифтом, например **x** или **w**. Векторы изображаются в виде направленных стрелок, а также

[1] Если термин выделен полужирным шрифтом, значит, он присутствует в алфавитном указателе в конце книги.

точек в многомерном пространстве. Иллюстрации трех двумерных векторов **a** = [2, 3], **b** = [−2, 5] и **c** = [1, 0] приведены на рис. 1.1. Атрибут вектора обозначается курсивной буквой с верхним индексом, например: $w^{(j)}$ или $x^{(j)}$. Индекс j обозначает конкретное **измерение** вектора, т. е. позицию атрибута в списке. Например, в векторе **a**, показанном красным цветом на рис. 1.1, $a^{(1)} = 2$ и $a^{(2)} = 3$.

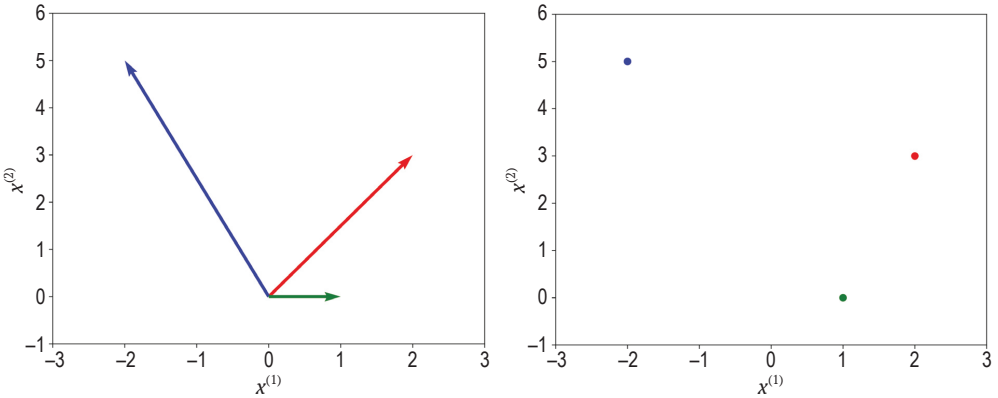

Рис. 1.1 ❖ Три вектора, представленных направленными стрелками и точками

Обозначение $x^{(j)}$ не следует путать с оператором возведения в степень, например 2 в x^2 (возведение в квадрат) или 3 в x^3 (возведение в куб). Если мы хотим применить оператор возведения в степень, скажем, в квадрат, к индексированному атрибуту вектора, то пишем: $(x^{(j)})^2$.

Переменная может иметь два и более индексов, например: $x_i^{(j)}$ или $x_{i,j}^{(k)}$. Так, в нейронных сетях $x_{l,u}^{(j)}$ обозначает j-й входной признак u-го блока в l-м слое.

Матрица – это прямоугольный массив чисел, организованный по строкам и столбцам. Ниже приведен пример матрицы с двумя строками и тремя столбцами:

$$\mathbf{A} = \begin{bmatrix} 2 & -2 & 1 \\ 3 & 5 & 0 \end{bmatrix}.$$

Матрицы обозначаются полужирными заглавными буквами, например **A** или **W**. Из примера матрицы **A** выше видно, что матрицы можно трактовать как регулярные структуры, состоящие из векторов. И действительно, столбцами приведенной выше матрицы **A** являются векторы **a**, **b** и **c**, показанные на рис. 1.1.

Множеством называется неупорядоченная коллекция неповторяющихся элементов. Множество обозначается каллиграфической заглавной буквой, например $\mathcal{S}$. Множество чисел может быть конечным (содержать фиксированное количество значений). В этом случае его элементы перечисляются в фигурных скобках, например {1, 3, 18, 23, 235} или $\{x_1, x_2, x_3, x_4, \ldots, x_n\}$. Множество также может быть бесконечным и включать все значения в некотором

интервале. Множество, содержащее все значения между a и b включительно, обозначается $[a, b]$. Если же множество не включает граничные значения a и b, то оно обозначается (a, b). Например, множество $[0, 1]$ включает среди прочего значения 0, 0.0001, 0.25, 0.784, 0.9995 и 1.0. Буквой $\mathbb{R}$ обозначается множество всех чисел от минус бесконечности до плюс бесконечности.

Если элемент x принадлежит множеству $\mathcal{S}$, то мы пишем $x \in \mathcal{S}$. Новое множество $\mathcal{S}_3$ можно получить как **пересечение** двух множеств $\mathcal{S}_1$ и $\mathcal{S}_2$. В этом случае мы пишем $\mathcal{S}_3 \leftarrow \mathcal{S}_1 \cap \mathcal{S}_2$. Например, $\{1, 3, 5, 8\} \cap \{1, 8, 4\}$ дает новое множество $\{1, 8\}$.

Новое множество $\mathcal{S}_3$ можно получить как **объединение** двух множеств $\mathcal{S}_1$ и $\mathcal{S}_2$. В этом случае мы пишем $\mathcal{S}_3 \leftarrow \mathcal{S}_1 \cup \mathcal{S}_2$. Например, $\{1, 3, 5, 8\} \cup \{1, 8, 4\}$ дает новое множество $\{1, 3, 5, 8, 4\}$.

$|\mathcal{S}|$ обозначает размер множества $\mathcal{S}$, то есть число элементов в нем.

1.1.2. Заглавная сигма

Сумма множества чисел $\mathcal{X} = \{x_1, x_2, \ldots, x_{n-1}, x_n\}$ или атрибутов вектора $\mathbf{x} = [x^{(1)}, x^{(2)}, \ldots, x^{(m-1)}, x^{(m)}]$ обозначается следующим образом:

$$\sum_{i=1}^{n} x_i \stackrel{\text{def}}{=} x_1 + x_2 + \ldots + x_{n-1} + x_n,$$

или

$$\sum_{i=1}^{n} x^{(j)} \stackrel{\text{def}}{=} x^{(1)} + x^{(2)} + \ldots + x^{(m-1)} + x^{(m)}.$$

Здесь $\stackrel{\text{def}}{=}$ означает «по определению равно».

Евклидова норма вектора $\mathbf{x}$, обозначаемая $||\mathbf{x}||$, характеризует «размер», или «длину», вектора. Она определяется как $\sqrt{\sum_{j=1}^{D}(x^{(j)})^2}$.

В качестве расстояния между векторами $\mathbf{a}$ и $\mathbf{b}$ берется **евклидово расстояние**:

$$||a - b|| \stackrel{\text{def}}{=} \sqrt{\sum_{i=1}^{N}(a^{(i)} - b^{(i)})^2}.$$

1.2. Что такое машинное обучение

Машинное обучение – это раздел информатики, посвященный построению алгоритмов, которые работают с набором примеров, описывающих какое-то явление. Примеры могут поступать из природы, создаваться людьми или генерироваться другим алгоритмом.

Машинное обучение также можно определить как процесс решения практической задачи путем

1) сбора набора данных и
2) алгоритмического обучения **статистической модели** на этом наборе.

Предполагается, что эта статистическая модель каким-то образом используется для решения практической задачи. Для краткости я буду использовать термины «обучение» и «машинное обучение» как синонимы. По той же причине я буду часто говорить «модель», имея в виду статистическую модель.

Обучение бывает с учителем, без учителя, с частичным привлечением учителя и с подкреплением.

1.2.1. Обучение с учителем

В случае **обучения с учителем** аналитик работает с набором **помеченных примеров** $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$. Каждый элемент x_i называется **вектором признаков**. В информатике вектор – это одномерный массив. Одномерный массив, в свою очередь, является упорядоченной и индексированной последовательностью значений. Длина этой последовательности значений, D, называется **размерностью** вектора.

Вектор признаков – это вектор, в котором каждое измерение $j = 1 \ldots D$ содержит значение, описывающее пример. Каждое такое значение называется **признаком** и обозначается $x^{(j)}$. Например, если каждый пример x представляет человека, то первый признак, $x^{(1)}$, может содержать рост в сантиметрах, второй признак, $x^{(2)}$, – вес в килограммах, $x^{(3)}$ – пол и т. д. Для всех примеров в наборе данных значение в позиции j вектора признаков всегда имеет один и тот же тип. Это означает, что если $x_i^{(2)}$ содержит вес в килограммах для некоторого i, то и для любого k от 1 до N $x_k^{(2)}$ будет содержать вес в килограммах. **Метка** y_i может быть либо элементом конечного множества классов $\{1, 2, \ldots, C\}$, либо вещественным числом, либо более сложной структурой, такой как вектор, матрица, дерево или граф. Если явно не оговорено противное, то в этой книге y_i является либо элементом конечного множества классов, либо вещественным числом[1]. Можно считать, что класс – это категория, к которой относится пример.

Скажем, если примерами являются сообщения электронной почты, а наша задача заключается в обнаружении спама, то есть два класса: спам и не спам. В случае обучения с учителем задача предсказания класса называется **классификацией**, а задача предсказания вещественного числа называется **регрессией**. Значение, которое должно быть предсказано моделью, обученной с учителем, называется **целевым показателем**, или целью. Примером регрессии является задача предсказания заработной платы сотрудника с учетом его опыта работы и знаний. Примером классификации является ситуация, когда врач вводит характеристики пациента в приложение, а то возвращает диагноз.

Различие между классификацией и регрессией показано на рис. 1.2. В случае классификации алгоритм обучения ищет линию (или, в более общем слу-

[1] Вещественное число – это величина, представляющая расстояние вдоль прямой от некоторой начальной точки на ней. Примеры: 0, –256.34, 1000, 1000.2.

чае, гиперповерхность), которая разделяет примеры разных классов. С другой стороны, в случае регрессии алгоритм обучения стремится отыскать линию или гиперповерхность, которая хорошо соответствует обучающим примерам.

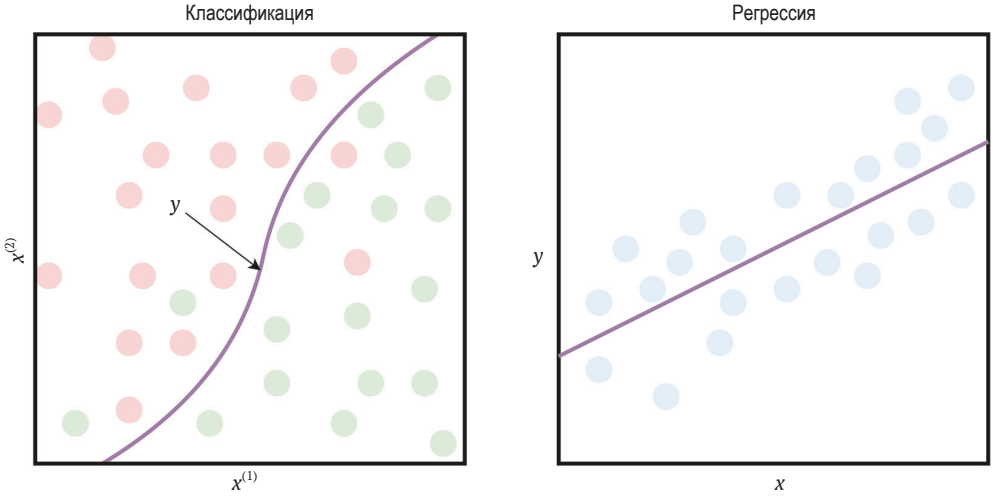

Рис. 1.2 ❖ Разница между классификацией и регрессией

Цель **алгоритма обучения с учителем** – использовать набор данных для порождения модели, которая на входе принимает вектор признаков **x**, а на выходе выдает информацию, позволяющую вывести метку для этого вектора признаков. Например, модель, созданная с использованием набора данных о пациентах, может на входе принимать вектор признаков, описывающий пациента, и на выходе выдавать вероятность наличия у пациента рака.

Даже если модель традиционно представляет собой математическую функцию, размышляя о действиях модели с входными данными, удобно думать, что модель «смотрит» на значения некоторых признаков на входе и, основываясь на опыте работы с аналогичными примерами, выводит значение. Это выходное значение представляет собой число или класс, «наиболее похожий» на метки, которые мы видели в прошлом в примерах с похожими значениями признаков. Такое описание выглядит упрощенно, но именно так работает модель решающего дерева и алгоритм *k* ближайших соседей.

1.2.2. Обучение без учителя

В случае **обучения без учителя** набор данных содержит **непомеченные примеры** $\{x_1, x_2, \ldots, x_N\}$. Как и раньше, **x** – вектор признаков, а цель **алгоритма обучения без учителя** заключается в порождении модели, которая на входе принимает вектор признаков **x** и преобразовывает его либо в другой вектор, либо в значение, используемое для решения практической задачи. Например, в случае **кластеризации** для каждого вектора признаков из на-

бора данных модель возвращает ИД кластера. Кластеризация применяется для отыскания групп похожих объектов в большом наборе объектов, например изображений или текстовых документов. Используя кластеризацию, аналитик может, например, выбрать достаточно репрезентативное, но малое подмножество непомеченных примеров из большого набора, чтобы потом пометить их вручную: из каждого кластера выбирается всего несколько примеров, вместо того чтобы отбирать непосредственно из большого набора с риском выбрать очень похожие примеры.

В задаче **понижения размерности** выходом модели является вектор признаков с меньшим числом измерений, чем на входе. Например, у исследователя имеется вектор признаков, слишком сложный для визуализации (поскольку число измерений больше трех). Модель понижения размерности может преобразовать этот вектор в другой (сохраняя часть информации) – двумерный или трехмерный. Этот новый вектор признаков можно изобразить на графике.

В задаче **обнаружения выбросов** выходом является действительное число, показывающее, насколько входной вектор признаков отличается от «типичного» примера в наборе данных. Обнаружение выбросов применяется для решения задачи о проникновении в сеть (путем обнаружения аномальных сетевых пакетов, которые отличаются от типичного пакета в «нормальном» трафике) или обнаружения новизны (например, документа, отличающегося от других документов в наборе).

1.2.3. Обучение с частичным привлечением учителя

В случае **обучения с частичным привлечением учителя** набор данных содержит как помеченные, так и непомеченные примеры. Обычно число непомеченных примеров намного превышает число помеченных. Цель **алгоритма с частичным привлечением учителя** такая же, как у алгоритма обучения с учителем. Мы надеемся, что, располагая большим числом непомеченных примеров, алгоритм обучения сможет отыскать (можно было бы сказать «породить» или «вычислить») лучшую модель.

1.2.4. Обучение с подкреплением

Обучение с подкреплением – это раздел машинного обучения, в котором рассматривается ситуация, когда машина (именуемая агентом) «обитает» в окружающей среде и способна воспринимать состояние этой среды как вектор признаков. Машина может выполнять действия в незаключительных состояниях. Разные действия приносят разные вознаграждения, а также могут переводить машину в другое состояние окружающей среды. Типичная цель алгоритма обучения с подкреплением – обучиться оптимальной **политике**.

Оптимальная политика – это функция (аналогичная модели в обучении с учителем), которая на входе принимает вектор признаков состояния, а на