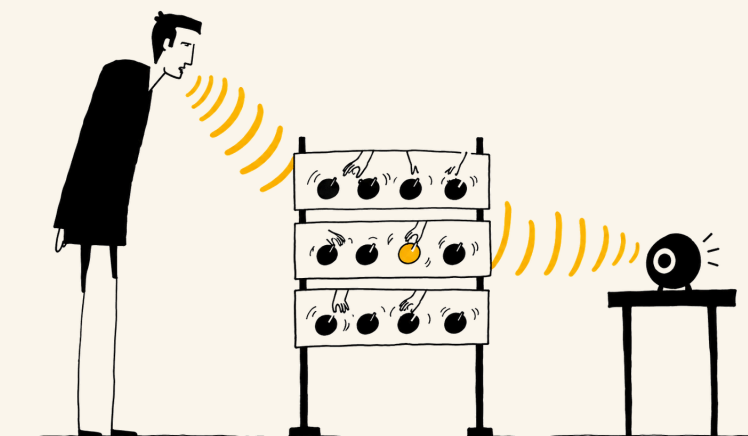


Погружение в глубокое обучение



ton Zhang, Zachary C. Lipton, Mu Li, Alexander J. Smola · 20

Иллюстрированные аудиолекции

Lecturized

Погружение в глубокое обучение

Иллюстрированные аудиолекции

Aston Zhang, Zachary C. Lipton, Mu Li, Alexander J. Smola · 2023

Пересказано в иллюстрированных лекциях Lecturized
lecturized.com

По оригиналу: Dive into Deep Learning (Cambridge University Press, 2023), d2l.ai

Содержание

Правила, которые никогда не учатся

Ручки, которые выучивают слово

Программирование с помощью данных

Правильные данные и честная оценка

Идти вниз по склону, затем учиться на ответах

Сколько? Регрессия

Который из? Классификация

Много меток сразу и лучшее — первым

Лента, которая изучает вас

Когда важен порядок: последовательности

Учиться без учителя

Когда мир отвечает

Учиться действуя: обучение с подкреплением

Старые корни: от церковных дверей до нейронов

Путь к глубокому обучению

Скрытые успехи и видимые

Чем ИИ не является и что делает обучение глубоким

Пробуй и делись

Правила, которые никогда не учатся

Откройте на телефоне любое приложение магазина, положите в корзину одну пару носков и посмотрите, что произойдёт. На значке корзины появляется маленькая цифра, носки возникают в списке, а их цена прибавляется к сумме. Никто в магазине не видел вашего нажатия; всё это сделала программа, и сделала точно так же, как для миллионов покупателей до вас. Эта программа написана как набор правил: когда случается вот это, делай вот то.



Интернет-магазин как набор правил

Книга начинается именно с такой программы, интернет-магазина, и спрашивает, как бы его построили создатели. Они постояли бы несколько часов у доски и разбили работу на три части. Во-первых, есть лицо магазина — страницы, которые вы видите в браузере или в приложении. Во-вторых, есть база данных — большая, аккуратно ведущаяся таблица, которая помнит, кто вы, что лежит в вашей корзине и всё, что кто-либо когда-либо покупал. В-третьих, посередине находится то, что программисты называют бизнес-логикой, а авторы, полуслушая, — мозгом приложения. Мозг — это длинный список правил, которые для любой ситуации, какую только можно придумать, говорят, что должна сделать программа.

Вот одно такое правило, и оно ровно настолько скучное, насколько звучит. Когда покупатель нажимает «в корзину», программа записывает новую строку в таблицу корзины, связывая номер покупателя с номером товара. Потом команда по одному перебирает странные пограничные случаи. Что должно произойти, если кто-то нажмёт «купить» при пустой корзине? Для этого тоже пишут правило, проверяют его, исправляют то, что сломалось, и проверяют снова. То же самое с простыми настройками вашего телефона: «если заряд ниже двадцати процентов, приглушить экран» — это правило, которое кто-то однажды набрал, и само оно никогда не изменится.

Удивительно, говорит книга, что это работает. Люди способны спроектировать такие системы в голове, часто для ситуаций, с которыми никто раньше не сталкивался, и запустить их ещё до прихода первого настоящего покупателя. И авторы дают простой совет: если вы можете написать правила, которые верны в 100% случаев, машинное обучение вам вообще не нужно. Правило,

которое всегда верно, дешевле, понятнее и надёжнее всего, чему можно научить машину.

Теперь представьте ту же умную команду у той же доски, но с другой задачей. Книга предлагает четыре таких задачи.

Предсказать завтрашнюю погоду по картам, спутниковым снимкам и погоде последних дней. Ответить на любой короткий фактический вопрос, который человек набрал своими словами.

Посмотреть на фотографию и обвести контуром каждого человека на ней. Показать каждому покупателю товары, которые ему, скорее всего, очень понравятся, но которые он никогда не нашёл бы сам, просматривая каталог. С последними двумя вы встречаетесь каждый день: галерея в телефоне находит лица ваших друзей, а видеоприложение заполняет ленту тем, чего вы не искали.



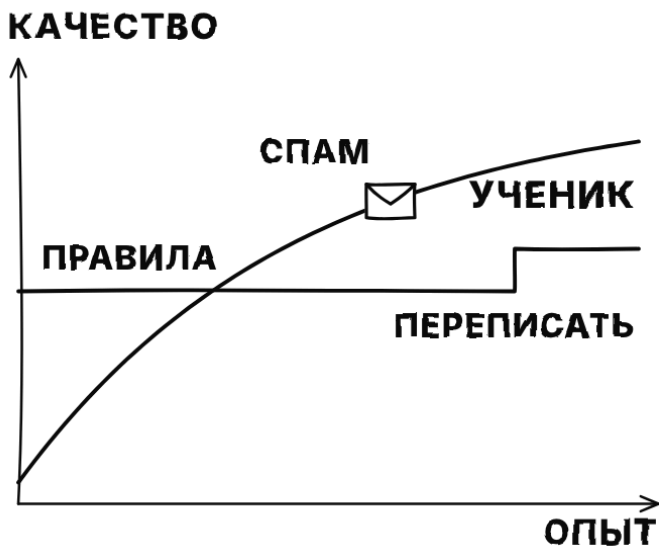
Четыре задачи без правил

Попробуйте написать правила для любой из этих задач, и вы почти сразу застрянете. Книга называет две причины. Иногда правильный ответ всё время сдвигается: то, что нравится покупателям этой весной, не то, что нравилось прошлой, поэтому неизменный список правил быстро устареет. Любое хорошее решение должно следовать за меняющимся миром. В других случаях связь между тем, что поступает на вход, и тем, что получается на выходе, слишком запутанна. Фотография — это всего лишь сетка цветных точек, называемых пикселями, и путь от этих точек к мысли «это человек» может потребовать тысяч или миллионов мелких вычислений, которые никто не умеет записать. Вы узнаете лицо друга в мгновение ока, но не смогли бы шаг за шагом объяснить, как вы это сделали.

Опыт

Попросите кого-нибудь записать в виде правил, как отличить на фото кошку от собаки. Он начнёт с ушей, размера и носа, и через минуту наткнётся на пушистую собаку с острыми ушами или на кошку, сфотографированную со спины. Каждое новое правило ломается на какой-нибудь новой картинке, хотя сам он никогда не спутает кошку с собакой. Этот разрыв между тем, что мы умеем делать, и тем, что мы можем записать, — именно то место, где вступает машинное обучение.

И книга даёт первое определение. Машинное обучение — это наука о программах, которые учатся на опыте: чем больше опыта накапливает такая программа, тем лучше она справляется со своей работой. Опыт обычно приходит в виде данных, например миллионов размеченных фотографий, или из проб в какой-то среде и наблюдения за тем, что получилось. Спам-фильтр — повседневный пример: каждый раз, когда люди помечают письмо как спам, фильтру становится немного больше, на чём учиться, и на следующей неделе он ловит больше мусора, чем на этой. Сравните это с интернет-магазином. Его правила остаются точно такими же, сколько бы покупателей через него ни прошло, и меняются, только когда сами программисты что-то узнают и садятся переписывать код. В магазине учатся люди; в машинном обучении учится программа.



Лучше с опытом

Затем книга говорит, куда она движется. Она научит основам машинного обучения и прежде всего одному мощному семейству его методов, которое называется глубоким обучением. Глубокое обучение стоит за недавними прорывами в компьютерном зрении, в понимании человеческого языка, в медицине и в чтении генома. Когда чат-бот отвечает на ваш вопрос гладкими фразами или голосовой помощник понимает вас сквозь уличный шум, вы слышите работу глубокого обучения.

Запомним

Большинство программ, которыми вы пользуетесь, — это списки правил, написанные людьми, и они ведут себя одинаково всегда, пока люди их не перепишут. Такой подход прекрасно работает, когда правила можно записать и они всегда верны. Он не работает, когда ответ всё время меняется или когда путь от входа к выходу слишком запутан, чтобы его описать, как при узнавании лица на фотографии. Машинное обучение — это наука о программах, которые становятся лучше с опытом, обычно с ростом объёма данных. Эта книга учит машинному обучению, и главным образом глубокому обучению — семейству методов, стоящих за сегодняшним прогрессом в зрении, языке и науке.

Вопросы

1. Правило магазина гласит: когда покупатель нажимает «в корзину», записать новую строку в таблицу корзин. Что станет с этим правилом после того, как им воспользуется миллион покупателей?
 - a. Оно понемногу подстроится под самые частые корзины
 - b. Оно останется точно таким же, пока программисты его не перепишут
 - c. Его автоматически заменит выученное правило

- d. Оно перестанет работать, когда таблица заполнится строками
2. Почему неизменный список правил для рекомендаций товаров покупателям быстро устареет?
- a. Потому что база данных забывает старые покупки
 - b. Потому что правила слишком медленно работают на телефонах
 - c. Потому что вкусы покупателей со временем меняются
 - d. Потому что покупатели никогда не покупают рекомендованные товары
3. Вы мгновенно отличаете кошку от собаки. Почему это не помогает написать правила для программы, которая делает то же самое?
- a. Вы не можете шаг за шагом объяснить, как вы это делаете
 - b. Для камеры кошки и собаки выглядят одинаково
 - c. На фотографиях слишком мало пикселей, чтобы увидеть животное
 - d. Программам нельзя смотреть на фотографии
4. Когда, по совету авторов, не стоит браться за машинное обучение?
- a. Когда программой будут пользоваться миллионы людей
 - b. Когда в задаче участвует база данных
 - c. Когда у вас совсем мало данных
 - d. Когда вы можете написать правила, которые всегда верны

Ответы

1. **b)** Правила магазина остаются точно такими же, сколько бы покупателей через него ни прошло; они меняются, только когда сами программисты что-то узнают и переписывают код.
2. **c)** Иногда правильный ответ всё время сдвигается: то, что нравится покупателям этой весной, не то, что нравилось прошлой, поэтому хорошее решение должно следовать за меняющимся миром.
3. **a)** Мы узнаём лица и животных в мгновение ока, но не можем объяснить, как именно; каждое правило, которое мы записываем, ломается на какой-нибудь новой картинке.
4. **d)** Если вы можете написать правила, которые верны в 100% случаев, машинное обучение вам вообще не нужно.

Задачи

1. Ваш телефон приглушает экран, когда заряд падает ниже двадцати процентов, а видеоприложение заполняет ленту тем, чего вы не искали. Что из этого — неизменное правило, а что требует машинного обучения? Объясните почему, опираясь на причины из лекции.

Подсказка: Спросите себя, можно ли записать правильный ответ один раз так, чтобы он оставался верным всегда.

Ответ: Настройка заряда — неизменное правило: его можно записать один раз, и оно всегда верно, поэтому учиться не нужно. Лента — задача для машинного обучения: то, что понравится каждому человеку, всё время меняется, а связь между историей просмотров и видео, которые ему понравятся, слишком запутанна, чтобы записать её правилами, поэтому программа должна улучшаться на опыте.

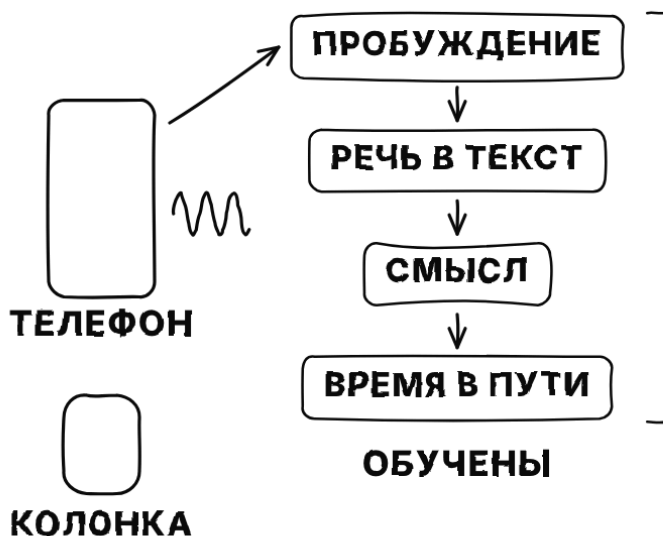
2. Своими словами объясните фразу: «В магазине учатся люди; в машинном обучении учится программа».

Подсказка: Подумайте, что меняет поведение магазина и что меняет поведение спам-фильтра.

Ответ: Программа магазина ведёт себя одинаково, сколько бы ею ни пользовались; она становится лучше, только когда её программисты что-то узнают и переписывают правила. Программа машинного обучения, например спам-фильтр, становится лучше сама, по мере того как набирает опыт, обычно новые данные, и никто не переписывает её правила.

Ручки, которые выучивают слово

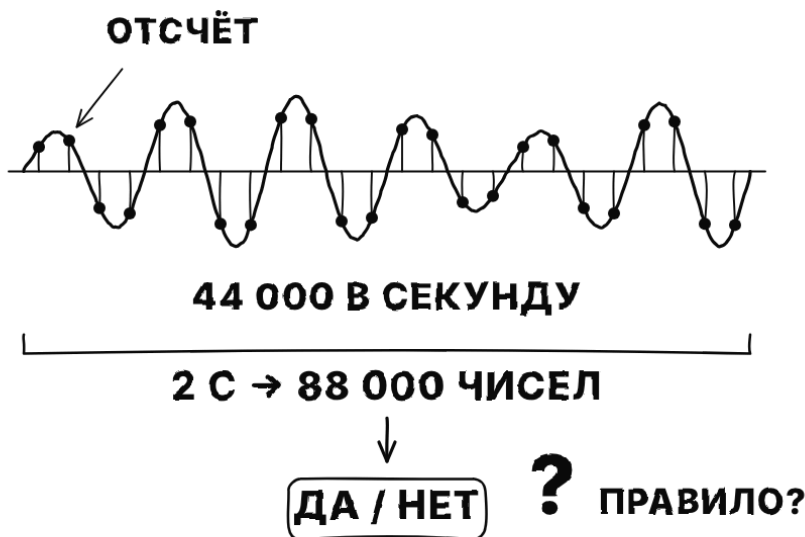
Скажите «Hey Siri» или «OK Google» через всю тихую комнату и посмотрите, как проснётся ваш телефон. Он загорается меньше чем за секунду, хотя лежал экраном вниз и мгновение назад, казалось, ничего не делал. Потом спросите у него дорогу до ближайшей кофейни. Ваши слова появляются на экране текстом, приложение с картами открывается само, а рядом с каждым маршрутом видно, сколько минут он займёт.



Четыре обученные программы в телефоне

Авторы рассказывают именно эту историю о себе: как они ехали за кофе, прежде чем сесть за книгу, и признаются, что придумали её ради наглядности. Суть в том, что за эти несколько секунд телефон использовал несколько программ, написанных не в виде правил. Одна услышала слово пробуждения, другая превратила речь в текст, третья поняла, что вам нужен маршрут, а четвёртая угадала время в пути. Каждая из них выучила свою работу на примерах. То же верно и для голосового помощника в кухонной колонке: он откликается на своё имя, но никто никогда не вводил в него правило о том, как это имя звучит.

Чтобы понять почему, попробуйте сами написать первую из этих программ, ту, что просыпается от слова. Представьте себя одного в комнате, где нет ничего, кроме компьютера; рисунок в книге показывает именно это: человек смотрит на редактор кода и звуковую волну. Начните с того, что программа на самом деле получает. Микрофон не слышит слов; он снова и снова измеряет, насколько воздух толкает его крошечную мембрану. Каждое такое измерение называется отсчётом, и это просто высота, или амплитуда, звуковой волны в данный момент. Телефон делает примерно 44 000 отсчётов в секунду. Значит, двухсекундный фрагмент — это $2 \times 44\,000 = 88\,000$ чисел. Из этих 88 000 чисел программа должна дать один ответ: да, слово пробуждения здесь есть, или нет, его нет.

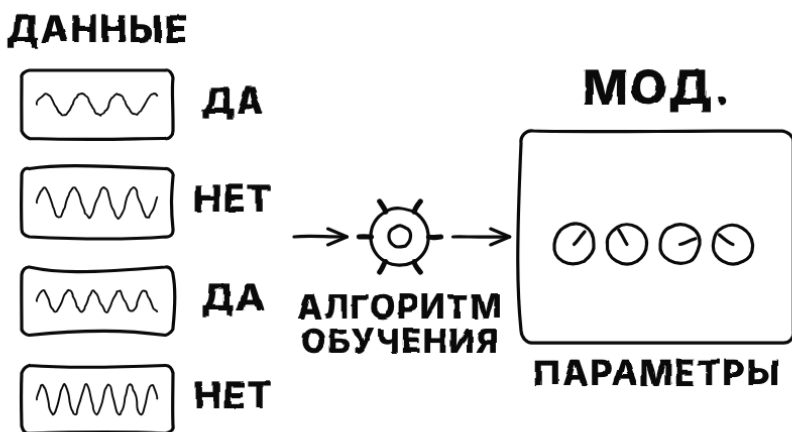


88 000 чисел, один ответ

Какое правило вы бы написали? Может быть, «если звук дважды становится громким и затем тихим, скажи да», но люди говорят громко и тихо, быстро и медленно, с насморком и с акцентом, под музыку на фоне. Книга честно признаёт: авторы тоже не знают, как написать такую программу с нуля. Именно поэтому мы и используем машинное обучение.

Вот хитрость, которую предлагает книга. Вы не можете записать, как вы узнаете слово «Alexa», но узнаете его сразу. Поэтому можно поступить иначе: собрать тысячи коротких записей и самому пометить каждую — «да», если в ней есть это слово, и «нет», если нет. Такой набор примеров называется набором данных, а пометки — метками. Затем вместо распознавателя вы

пишете гибкую программу, поведение которой зависит от множества настраиваемых чисел, называемых параметрами. Наконец вы позволяете набору данных решить, при каких значениях параметров программа лучше всего справляется по выбранной вами мере, например как часто её «да» или «нет» совпадает с вашей меткой.

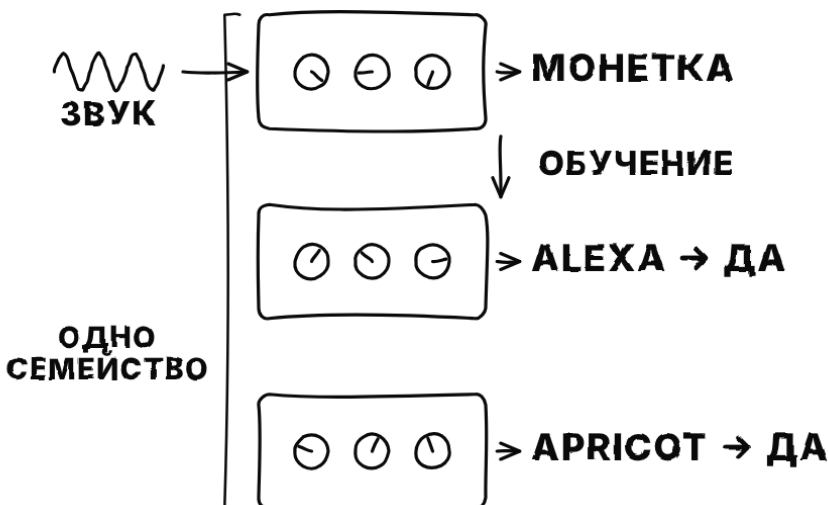


Метки выставляют ручки

Авторы предлагают представлять параметры как ручки на микшерном пульте. Поверните ручки, и программа поведёт себя иначе: тот же входящий звук даст другой ответ. Когда все ручки выставлены, книга называет программу моделью. Все разные программы, которые можно получить, вращая одни и те же ручки, составляют семейство моделей. А программа, которая

смотрит на набор данных и решает, куда выставить ручки, называется алгоритмом обучения; это программа, которая подбирает настройки другой программы. Именно так устроен спам-фильтр: его семейство задано заранее, а письма, которые люди помечают как спам, решают, где окажутся его ручки.

Прежде чем начнётся обучение, кто-то должен точно сформулировать задачу. Здесь вход — это фрагмент звука, те самые десятки тысяч чисел, а выход — один из двух ответов, «да» или «нет». Затем вы выбираете семейство моделей, подходящее для такого входа и выхода. Если семейство хорошее, найдётся такое положение ручек, при котором модель говорит «да» каждый раз, когда слышит «Alexa». А поскольку слово пробуждения может быть любым, хорошее семейство достаточно богато, чтобы при другом положении ручек говорить «да» только на «Apricot». Обе задачи кажутся похожими, так что одно семейство должно подойти для обеих. Но если вы захотите переходить от фотографий к подписям или от английских предложений к китайским, входы и выходы будут совсем другого рода, и вам может понадобиться совершенно другое семейство.

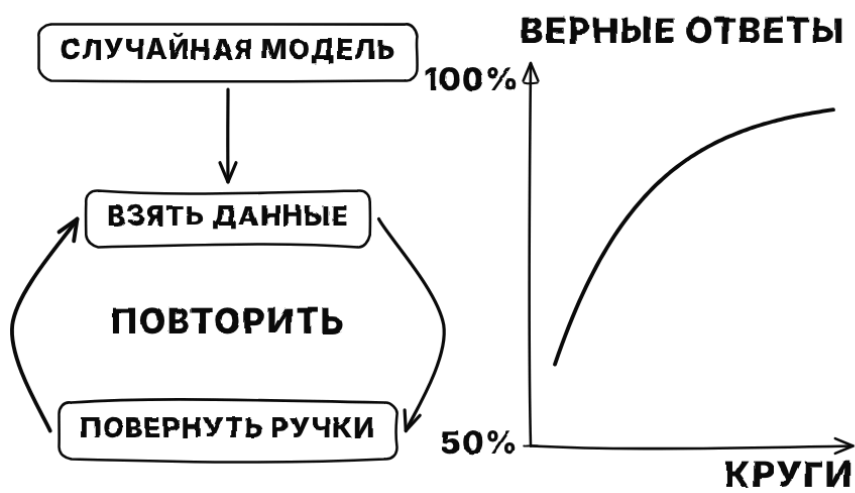


Одно семейство, много настроек

Так где же ручки стоят вначале? Если выставить их наугад, модель почти наверняка не узнает ни «Alexa», ни «Apricot», ни любое другое слово; она будет говорить «да» и «нет» более или менее случайно. В машинном обучении обучение означает именно это: найти такое положение ручек, при котором модель делает то, что нам нужно. Ещё говорят, что мы обучаем модель на данных.

Второй рисунок книги изображает обучение как цикл из четырёх шагов. Первый: начать с модели, ручки которой выставлены наугад и которая бесполезна. Второй: взять часть данных, например горстку записей с их метками «да» или «нет». Третий: чуть повернуть ручки, чтобы модель отвечала правильно на

большее число этих примеров. Четвёртый: вернуться ко второму шагу и ходить по кругу, пока, по словам авторов, «модель не станет великолепной». Представьте, что за раз поступает несколько сотен записей, после каждой порции все ручки чуть поворачиваются, а доля правильных ответов ползёт вверх примерно от половины, то есть от подбрасывания монетки, почти до всех.



Цикл обучения

Опыт

Попросите друга спрятать в комнате предмет и подсказывать вам только словами «теплее» и «холоднее». Вы начинаете идти в случайном направлении, что бесполезно; после каждого шага слышите одно слово и немного поправляетесь; и так снова. Вы так и не узнаете правила, где лежит предмет, но после достаточного числа кругов оказываетесь прямо рядом с ним. Обучение модели устроено так же, только она может двигаться сразу в миллионах направлений, а «теплее» получается из сравнения её ответов с метками.

Запомним

Ваш телефон просыпается от слова, потому что модель выучила его на примерах, а не потому, что кто-то написал для этого правило. Вход огромен, около 44 000 отсчётов амплитуды в секунду, а выход — одно-единственное «да» или «нет». Мы не можем описать, как слышим слово, но можем пометить записи, и именно этот размеченный набор данных учит программу. Модель — это гибкая программа, у которой ручки, то есть параметры, выставлены в определённые значения; алгоритм обучения — это то, что находит эти значения. Обучение — это цикл: начать наугад, посмотреть на несколько размеченных примеров, повернуть ручки, чтобы лучше с ними справиться, и повторить.

Вопросы

1. Никто не может записать, как распознать слово «Alexa». Что всё же позволяет научить этому программу?
 - a. Люди узнают это слово и могут пометить записи «да» или «нет»
 - b. Микрофон измеряет слово напрямую
 - c. У этого слова всегда одинаковый рисунок громкости
 - d. Телефон хранит запись каждого возможного голоса
2. У программы слова пробуждения все ручки выставлены наугад. Чего следует ожидать?
 - a. Она узнаёт «Alexa», но не другие слова
 - b. Она вообще отказывается отвечать
 - c. Она отвечает «да» и «нет» более или менее случайно
 - d. Она одинаково хорошо узнаёт любое слово
3. Почему одно и то же семейство моделей может подойти для «Alexa» и «Apricot», но не для превращения фотографий в подписи?
 - a. Подписи длиннее слов пробуждения
 - b. Фотографии и подписи — это другой род входа и выхода
 - c. «Apricot» — не настоящее слово пробуждения
 - d. Алгоритм обучения не умеет работать с фотографиями
4. Что из этого — алгоритм обучения в примере со словом пробуждения?
 - a. Микрофон, который собирает отсчёты

- b. Готовая модель с выставленными ручками
- c. Размеченные записи «да» и «нет»
- d. Программа, которая по набору данных выставляет ручки

Ответы

1. a) Вы не можете записать, как узнаёте слово, но узнаёте его сразу, поэтому можете пометить тысячи записей и дать этому набору данных научить программу.
2. c) При ручках, выставленных наугад, модель почти наверняка не узнает ни одного слова; обучение — это поиск положения, при котором она делает то, что нам нужно.
3. b) Два слова пробуждения похожи, поэтому одно семейство подходит для обоих; у перехода от фото к подписям входы и выходы совсем другого рода, и может понадобиться другое семейство.
4. d) Программа, которая смотрит на набор данных и решает, куда выставить ручки, — это алгоритм обучения, программа, подбирающая настройки другой программы.

Разобранные примеры

Сколько чисел слышит модель слова пробуждения

Микрофон телефона делает около 44 000 отсчётов в секунду.

Сколько чисел получает модель слова пробуждения из трёхсекундного фрагмента и сколько ответов она должна дать?

Каждый отсчёт — одно число, высота звуковой волны в данный момент.

За одну секунду около 44 000 отсчётов.

За 3 секунды: $3 \times 44\,000 = 132\,000$ отсчётов.

Задача поставлена так: на входе звук, на выходе «да» или «нет», поэтому все эти числа ведут к одному-единственному ответу.

Ответ: Около 132 000 чисел на входе, один ответ «да» или «нет» на выходе

Задачи

1. Расставьте по порядку четыре шага обучения и скажите, что делает каждый: повторить; подкрутить ручки; начать со случайной модели; взять немного размеченных данных.

Подсказка: Цикл не может начаться, пока нет модели, пусть даже бесполезной.

Ответ: Начать с модели, ручки которой выставлены наугад и которая бесполезна; взять немного данных с метками; подкрутить ручки, чтобы модель отвечала правильно на большее число этих примеров; повторять два последних шага, пока модель не станет хорошей.

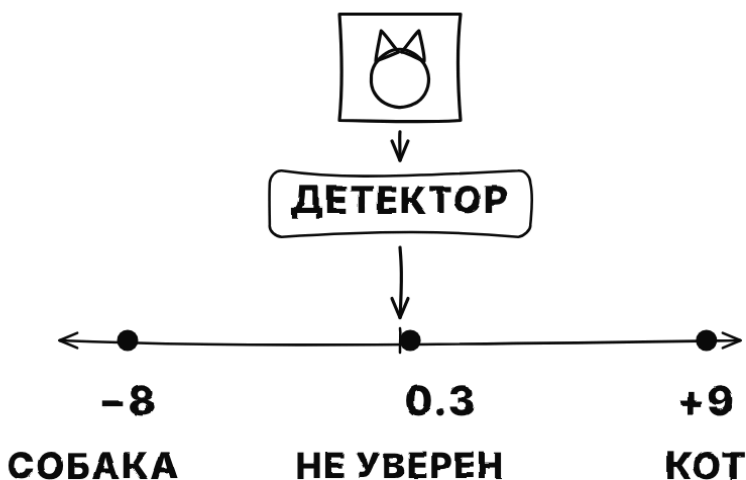
2. Фрагмент, поданный модели слова пробуждения, содержит 88 000 отсчётов. Какова длина фрагмента?

Подсказка: Разделите на число отсчётов, которые делаются за одну секунду.

Ответ: $88\,000 \div 44\,000 = 2$, значит, фрагмент длится около 2 секунд.

Программирование с помощью данных

Сфотографируйте телефоном свою кошку, затем откройте галерею и наберите в поиске «кошка». Фотография найдётся вместе со всеми другими кошками, которых вы когда-либо снимали, и ни одной собаки. Никто не писал для этого поиска правила, описывающего усы и уши. Где-то программе показали очень много фотографий кошек и собак, каждую с правильным ответом, и она по ним выучила разницу.



Оценка детектора кошек

Именно для этого и нужен цикл обучения из прошлой части. Вместо того чтобы писать распознаватель слова пробуждения, мы пишем программу, которая может научиться распознавать слова пробуждения, если дать ей большой размеченный набор данных. Авторы называют это программированием с помощью данных: поведение программы определяют не строки кода, а примеры, которые мы ей показываем. Точно так же можно «запрограммировать» детектор кошек, просто дав ему много картинок с кошками и собаками.

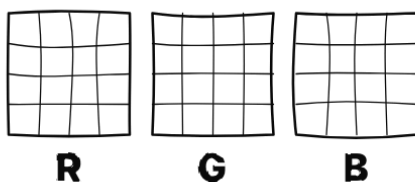
Книга говорит, что в итоге делает такой детектор. Для каждой картинki он выдаёт одно число. Очень большое положительное число означает «кошка», очень большое отрицательное — «собака», а что-то близкое к нулю — «я не уверен». Так что если он выдаёт +9 для одной картинki, -8 для другой и 0,3 для третьей, он уверен, что это кошка, уверен, что это собака, и колеблется насчёт последней, возможно, размытой лисы. Книга добавляет, что это лишь верхушка айсберга и что глубокое обучение — только один из многих популярных способов решать такие задачи.

Теперь книга делает шаг назад и называет тот вид задачи, который мы решали. У нас были записи с известным ответом, «да» или «нет», и мы хотели предсказать неизвестный ответ для новых записей. Обучение предсказывать выбранную неизвестную метку по известным входам на примерах с известными метками называется обучением с учителем. Это лишь один вид машинного обучения, и в следующих частях встретятся другие. Но каким бы ни был вид, говорят авторы, всегда присутствуют четыре составляющие. Есть данные, на которых учатся. Есть модель, которая превращает данные в

ответ. Есть целевая функция — оценка, которая говорит, насколько хорошо или плохо работает модель. И есть алгоритм, который крутит ручки модели так, чтобы оценка улучшалась.

В этой части речь о первой составляющей — данных. Набор данных — это собрание примеров, и компьютер может работать с ними только после того, как они превращены в числа. У каждого примера есть набор атрибутов, называемых признаками, по которым модель должна сделать предсказание. В обучении с учителем есть ещё один особый атрибут — метка, то есть то, что мы хотим предсказать; модели на вход она не подаётся. Книга упоминает, что у этих понятий есть и другие названия: пример называют также точкой данных или образцом, а признаки — входами или ковариатами.

ПРИМЕР



$$200 \times 200 = 40,000$$

$$200 \times 200 \times 3 \\ = 120,000$$

Фотография как 120 000 чисел

Для картинок признаки — это сама фотография, записанная числами, а метка — число, обозначающее категорию, скажем 1 для кошки и 2 для собаки. Цветная фотография превращается в три сетки чисел: одна для яркости красного света в каждом пикселе, одна для зелёного и одна для синего. Книга делает подсчёт для фотографии размером 200×200 пикселей. В одной сетке $200 \times 200 = 40\,000$ чисел, а сеток три, так что вся фотография — это $200 \times 200 \times 3 = 120\,000$ чисел. Одна маленькая картинка — это уже список из ста двадцати тысяч значений. Фотография прямо с современного телефона содержит миллионы пикселей, так что прежде чем программа сможет на ней учиться, она превращается во многие миллионы чисел.

Данные не обязательно картинки. Второй пример книги — больничные карты, где задача — предсказать, проживёт ли пациент следующие 30 дней. Признаки — это то, что и так записывается: возраст, жизненные показатели вроде пульса и давления, другие болезни, принимаемые лекарства, недавние операции. Метка для каждого пациента из прошлых записей — просто «да» или «нет»: прожил ли этот пациент 30 дней?



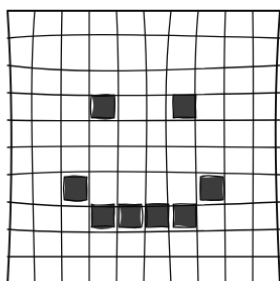
Фиксированная длина и переменная длина

В обоих примерах у каждого случая одно и то же число признаков. Каждая фотография 200 × 200 пикселей даёт ровно 120 000 чисел, а каждая карта пациента — один и тот же список измерений. Такие входы называют векторами фиксированной длины, где вектор — упорядоченный список чисел, а длину этого

списка называют размерностью данных. Значит, размерность наших данных о фотографиях — 120 000. Фиксированная длина удобна, говорит книга, потому что это одной сложностью меньше.

Настоящие данные часто не желают быть такими аккуратными. Снимки с микроскопа выходят из одного и того же прибора и одного размера, а картинки, собранные в интернете, бывают любых размеров и форм. Можно обрезать их все до стандартного квадрата, но всё отрезанное потеряется. Текст ещё упрямее. Вспомните отзывы в магазине приложений: в одном сказано только «гадость!», а следующий тянется на страницы о каждой функции. Естественной фиксированной длины у отзыва нет. Книга считает одним из больших преимуществ глубокого обучения то, что его модели сравнительно легко справляются с данными переменной длины.

Наконец, книга говорит о количестве. Как правило, чем больше данных, тем легче работа. Больше данных позволяет обучать более мощные модели и меньше опираться на предположения, придуманные заранее. Переход от довольно малых наборов данных к очень большим — одна из главных причин успеха современного глубокого обучения. Многие из самых впечатляющих его моделей просто не работают без больших наборов данных, а некоторые, что работают и на малых данных, не лучше старых методов.



0 0 0 0 0 0 0 0 0 0

⋮

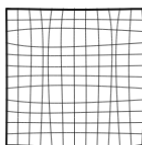
0 0 0 1 0 0 1 0 0 0

⋮

0 0 0 1 1 1 1 0 0 0

⋮

**100 ЧИСЕЛ
РАЗМЕРНОСТЬ 100**



**144
≠ 100**

Рожица как 100 чисел

Опыт

Возьмите лист в клетку, десять клеток на десять, и закрасьте в нескольких клетках простую улыбающуюся рожицу.

Напишите 1 в каждой закрашенной клетке и 0 в каждой пустой и прочитайте числа строка за строкой. Теперь у вас есть список из 100 чисел, который и есть картинка, — вектор фиксированной длины размерности 100. Нарисуйте вторую рожицу на листе двенадцать на двенадцать, и получится 144 числа, которые уже не помещаются в тот же список; это и есть проблема переменной длины в миниатюре.

Запомним

Программирование с помощью данных — это задание поведения программы примерами, которые мы ей показываем, а не правилами, которые мы пишем. В каждой задаче машинного обучения четыре составляющие: данные, модель, целевая функция, оценивающая модель, и алгоритм, улучшающий оценку. В обучении с учителем у каждого примера есть признаки, которые видит модель, и метка, которую она должна предсказать. Цветная фотография 200×200 — это 120 000 чисел; когда у каждого примера одинаковое число признаков, это число и есть размерность. Глубокое обучение хорошо справляется с данными переменной длины, и чтобы показать свою силу, ему нужны большие наборы данных.

Вопросы

1. Детектор кошек выдаёт для фотографии –7. О чём это говорит?
 - a. Он довольно уверен, что на фото собака
 - b. Он довольно уверен, что на фото кошка
 - c. Он не уверен, что на фото
 - d. На фото слишком мало пикселей, чтобы судить
2. Что из этого в больничном наборе данных для предсказания выживания в течение 30 дней является

- меткой, а не признаком?
- a. Возраст пациента
 - b. Лекарства, которые принимает пациент
 - c. Прожил ли пациент 30 дней
 - d. Давление пациента
3. Почему отзывы из магазина приложений трудно превратить в векторы фиксированной длины?
- a. В них нет чисел
 - b. Они бывают от двух слов до многих страниц
 - c. Они все написаны на разных языках
 - d. Они слишком коротки, чтобы на них учиться
4. Что из этого НЕ входит в четыре составляющие любой задачи машинного обучения?
- a. Целевая функция, которая оценивает модель
 - b. Данные, на которых учатся
 - c. Алгоритм, который улучшает оценку
 - d. Правило, написанное вручную для каждого случая

Ответы

1. a) Очень большое положительное число означает кошку, очень большое отрицательное — собаку, а что-то около нуля — «не уверен».
2. c) Признаки — то, что и так записывается, например возраст, жизненные показатели и лекарства; метка — прожил ли каждый прошлый пациент 30 дней.
3. b) В одном отзыве сказано только «гадость!», а другой тянется на страницы, так что естественной фиксированной длины у отзыва нет.

4. d) Четыре составляющие — данные, модель, целевая функция и алгоритм, который крутит ручки; написанные вручную правила — это как раз то, что заменяет машинное обучение.

Разобранные примеры

Числа в фотографии 200×200 Цветная фотография размером 200×200 пикселей записана тремя сетками чисел — для красного, зелёного и синего. Сколько это чисел и какова размерность набора данных из таких фотографий?

В одной сетке по одному значению яркости на пиксель: $200 \times 200 = 40\,000$ чисел.

Сеток три, по одной на каждый цвет света: $40\,000 \times 3 = 120\,000$.

Каждая фотография такого размера даёт одно и то же число значений, поэтому входы — векторы фиксированной длины.

Размерность — это длина этих векторов.

Ответ: 120 000 чисел на фотографию; размерность равна 120 000

Задачи

1. Фотография размером 300×200 пикселей записана тремя сетками для красного, зелёного и синего. Сколько это чисел?

Подсказка: Сначала посчитайте пиксели, затем умножьте на число цветовых сеток.

Ответ: $300 \times 200 = 60\,000$ пикселей, умножить на 3 сетки — 180 000 чисел.

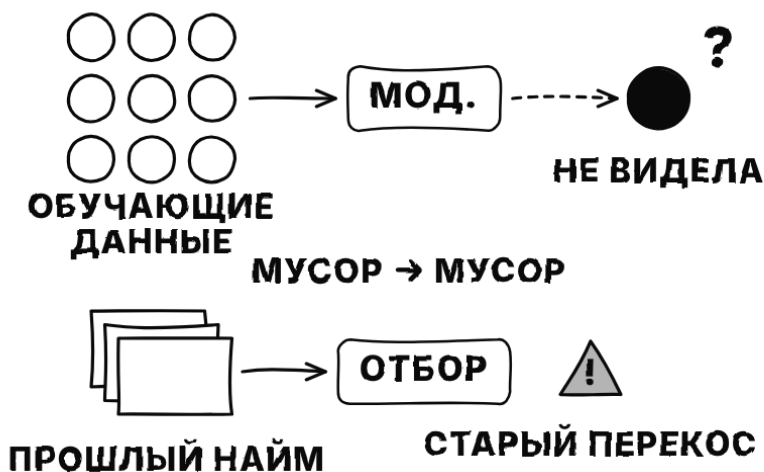
2. Рожица закрашена на листе в клетку 10 на 10 клеток и записана единицами и нулями. Вторая нарисована на листе 12 на 12. Почему их нельзя поместить в один набор данных из векторов фиксированной длины, ничего не изменив?

Подсказка: Посчитайте, сколько чисел даёт каждый лист.

Ответ: Первый лист даёт $10 \times 10 = 100$ чисел, а второй $12 \times 12 = 144$, поэтому векторы разной длины; пришлось бы обрезать или менять масштаб, рискуя потерять информацию, или взять модель, которая справляется с переменной длиной.

Правильные данные и честная оценка

Представьте приложение для проверки кожи, которое смотрит на фотографию родинки и говорит, стоит ли показать её врачу. Его обучили на многих тысячах фотографий, и на испытаниях оно было очень точным. И вот человек с тёмной кожей фотографирует родинку и получает явно неверный ответ. Книга приводит именно этот случай — систему распознавания рака кожи, которая никогда не видела тёмной кожи, — и называет причину в трёх словах: данные были неправильными.



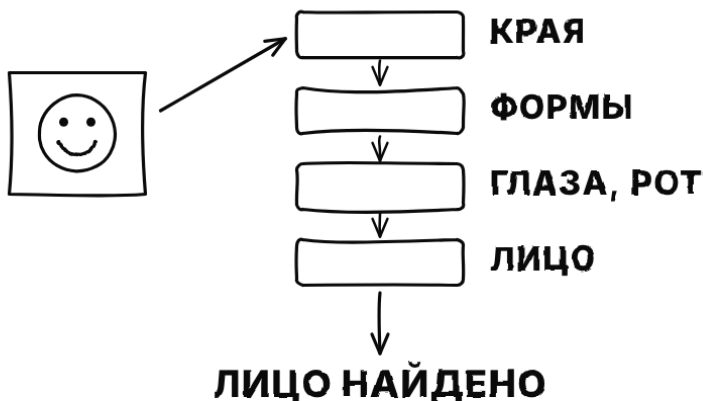
Пропущенные группы, старые предрассудки

Предыдущая часть была о том, чтобы данных было достаточно и чтобы превратить их в числа. Здесь авторы добавляют, что этого мало: нужны правильные данные. Если примеры полны ошибок или выбранные признаки просто ничего не говорят о том, что мы хотим предсказать, обучение проваливается, каким бы умным ни был метод. У программистов для этого есть избитая фраза: «мусор на входе — мусор на выходе». И плохая оценка — не самое худшее, что может случиться. Книга называет области, где неправильные данные приносят реальный вред: предсказание преступлений для полиции, отбор кандидатов на работу и решения о том, кому дать кредит.

Часто встречаются два вида ошибок. Первый — какой-то группы людей просто нет в обучающих данных, как тёмной кожи в нашем приложении. Модель никогда не видела таких случаев, поэтому ей не на что опереться. Второй тоньше: в данных есть все, но они записывают старые предрассудки. Обучите программу на многолетних прошлых решениях о найме, и если те решения были несправедливы, программа выучит несправедливость и будет повторять её автоматически и в больших масштабах. Чат-бот, обученный на текстах из интернета, может перенять те же перекосы, что есть в этих текстах, поэтому его создатели проверяют его на таких случаях. Книга подчёркивает, что для всего этого не нужен злодей: всё может случиться без умысла специалиста по данным и даже без того, чтобы он это заметил.

Вторая составляющая — модель. Почти всё машинное обучение превращает данные одного рода в предсказание другого рода. Система может принимать фотографии и говорить, насколько улыбается каждый человек, или принимать показания датчиков

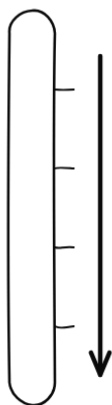
на станке и говорить, насколько они нормальны или необычны. Модель — это и есть этот механизм от входа до предсказания, и в этой книге это статистическая модель, настройки которой оцениваются по данным. Простые модели прекрасно подходят для простых задач. Но задачи, которые интересуют эту книгу, выходят за пределы возможностей классических методов, а глубокое обучение выделяется прежде всего мощными моделями, которые оно использует. Эти модели пропускают данные через много преобразований подряд, каждое из которых питает следующее, сложенных стопкой сверху вниз; эта глубина и дала название глубокому обучению. Так работает приложение для фотографий, которое находит лица: первые стадии выделяют края, следующие — формы, ещё следующие — глаза и рты, а последняя говорит: «лицо».



Много стадий, одна за другой

Третья составляющая отвечает на вопрос, который звучит философски. Мы сказали, что учиться — значит становиться лучше в какой-то задаче. Но кто решает, что такое «лучше»? Если один человек предлагает изменить модель, другой вполне может возразить, что стало хуже. Чтобы разрешать такие споры, машинному обучению нужно число, измеряющее, насколько модель хороша или плоха. Это число называется целевой функцией.

ПОТЕРИ



ЛУЧШЕ

оценка $\uparrow$

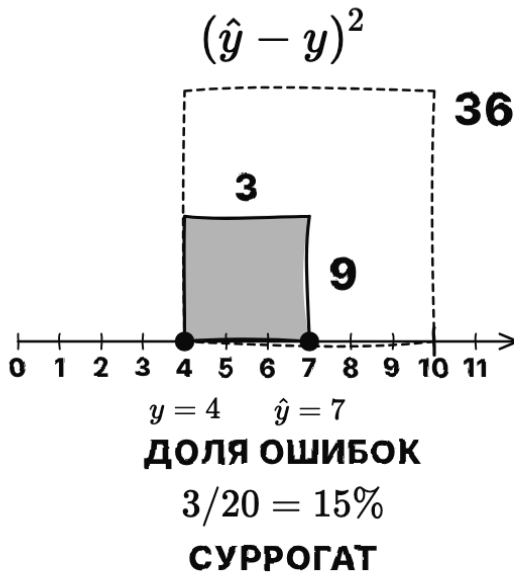
потери = $-$ оценка $\downarrow$

Меньше — лучше

По соглашению целевые функции устраивают так, чтобы меньше означало лучше. Это всего лишь привычка, говорит книга. Если у вас оценка, где больше — лучше, например доля правильных ответов, достаточно поставить перед ней знак минус, и теперь меньше — лучше, а больше ничего не изменилось. Поскольку меньше — лучше, такие функции называют также функциями потерь: чем меньше потери, тем лучше модель.

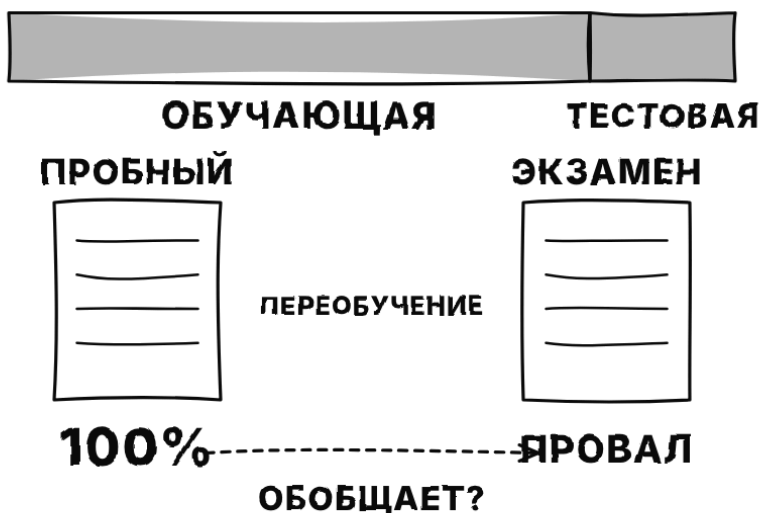
Книга называет две самые распространённые функции потерь. Когда мы предсказываем число, обычная функция потерь — квадрат ошибки: берём предсказание, вычитаем истинное значение и возводим разность в квадрат. В виде формулы: если предсказание равно $\hat{y}$, а истинное значение y , квадрат ошибки

равен $(\hat{y} - y)^2$. Допустим, модель предсказывает, что доставка займёт 7 дней, а она занимает 4. Разность равна $7 - 4 = 3$, а квадрат ошибки $3^2 = 9$. Возведение в квадрат делает потери положительными, в какую бы сторону мы ни промахнулись, и наказывает большой промах гораздо сильнее, чем маленький: промах на 6 стоит 36, вчетверо больше, чем промах на 3. Когда мы распределяем объекты по классам, естественная оценка — доля ошибок, то есть доля примеров, на которых мы ошиблись; 3 ошибки из 20 — это $3/20 = 0,15$, или 15 процентов. Книга предупреждает, что одни функции потерь, как квадрат ошибки, легко уменьшать шаг за шагом, а другие, как доля ошибок, неудобны: если чуть повернуть ручки, обычно не меняется ни один ответ, и доля скачет лишь время от времени. В таких случаях оптимизируют заменитель, с которым легче работать, — так называемую суррогатную цель.



Квадрат ошибки и доля ошибок

Пока идёт обучение, мы смотрим на потери как на зависящие от ручек модели, то есть параметров, а данные держим неизменными. Мы собираем набор примеров для обучения и ищем такие положения ручек, при которых потери на них как можно меньше. Но малые потери на этих примерах не обещают хороших ответов на новых. Поэтому данные делят на две части: обучающую выборку, на которой выставляют ручки, и тестовую выборку, которую откладывают и используют только для проверки результата. В конце сообщают обе оценки.



Обучающая выборка, тестовая выборка, переобучение

Опыт

Представьте студента, который готовится к экзамену по сборнику пробных вопросов. Через несколько недель он каждый раз получает по сборнику высший балл, потому что выучил ответы наизусть. Потом приходит экзамен с новыми вопросами, и он его проваливает. Балл по сборнику был оценкой на обучающей выборке, а экзамен — тестом.

Эта история — собственная иллюстрация книги к тому, что идёт не так. Когда модель хорошо работает на обучающей выборке, но не может перенести это на данные, которых не видела, мы

говорим, что она переобучается на обучающих данных. На самом деле мы хотим обратного: модель, которая обобщает, то есть работает на новых примерах примерно так же хорошо, как на тех, на которых училась. Спам-фильтр, который блокирует только те самые письма, что ему показали, переобучился бы; полезный фильтр блокирует и новый обман следующей недели.

Запомним

Много данных недостаточно; они должны быть правильными, потому что мусор на входе даёт мусор на выходе. Данные, в которых не хватает каких-то людей или которые несут старые предрассудки, порождают модели, которые ошибаются или несправедливо обращаются с людьми, даже если никто этого не хотел. Модель превращает вход в предсказание, а модели глубокого обучения сцепляют много преобразований одно за другим. Целевая функция, или функция потерь, оценивает модель, и меньше — лучше; квадрат ошибки $(\hat{y} - y)^2$ — обычная функция для чисел, доля ошибок — для классов. Мы обучаем на обучающей выборке и проверяем на отдельной тестовой, потому что хорошо работать только на обучающих данных — это переобучение.

Вопросы

1. Модель найма обучили на многолетних прошлых решениях, в которых были учтены все, но которые были

несправедливы. Что, по словам лекции, может случиться?

- a. Модель выучит несправедливость и будет повторять её автоматически
- b. Модель ошибётся, потому что не хватает какой-то группы
- c. Модель сама исправит несправедливость
- d. Ничего, если только специалист по данным этого не хотел

2. Модель предсказывает 10, а истинное значение равно 13. Каков квадрат ошибки?

- a. 3
- b. 6
- c. 9
- d. -9

3. Почему вместо доли ошибок часто оптимизируют суррогатную цель?

- a. Доля ошибок всегда слишком велика
- b. Долю ошибок нельзя вычислить на тестовой выборке
- c. Доля ошибок поощряет переобучение
- d. Небольшие повороты ручек обычно не меняют долю ошибок, поэтому её неудобно улучшать шаг за шагом

4. Спам-фильтр блокирует все письма, на которых его обучали, но пропускает большинство новых мошеннических писем. Что происходит?

- a. Он использует не ту функцию потерь
- b. Он переобучился на обучающих данных

- с. Его обучающая выборка была слишком мала, чтобы содержать спам
- д. Его оценивали по принципу «меньше — лучше»

Ответы

- 1. **а)** Данные могут включать всех и всё равно записывать старые предубеждения; программа, обученная на несправедливых прошлых решениях, выучивает и повторяет их, даже если никто этого не хотел.
- 2. **с)** Квадрат ошибки — это предсказание минус истинное значение, возведённое в квадрат: $10 - 13 = -3$, и $(-3)^2 = 9$; квадрат делает величину положительной, в какую бы сторону мы ни промахнулись.
- 3. **д)** Если чуть повернуть ручки, обычно не меняется ни один ответ, и доля ошибок скачет лишь время от времени; поэтому оптимизируют заменитель, с которым легче работать.
- 4. **б)** Хорошо работать на обучающей выборке, но не переносить это на невиданные данные — это переобучение, как у студента, который вызубрил сборник пробных вопросов.

Разобранные примеры

Два промаха, два квадрата ошибки Модель времени доставки предсказывает 7 дней для одной посылки, которая идёт 4 дня, и 7 дней для другой, которая идёт 1 день. Каковы два квадрата ошибки и как они соотносятся?

Квадрат ошибки — это (предсказание – истинное значение)².

Первая посылка: $7 - 4 = 3$, значит, квадрат ошибки $3^2 = 9$.

Вторая посылка: $7 - 1 = 6$, значит, квадрат ошибки $6^2 = 36$.

Второй промах вдвое больше, но стоит в $36 \div 9 = 4$ раза дороже: квадрат гораздо сильнее наказывает большие промахи.

Ответ: 9 и 36; большой промах стоит вчетверо дороже

Задачи

1. Классификатор проверяют на 50 фотографиях, и на 4 из них он ошибается. Какова доля ошибок?

Подсказка: Доля ошибок — это доля примеров, предсказанных неверно.

Ответ: $4 \div 50 = 0,08$, или 8 процентов.

2. Ваша оценка — «доля правильных ответов», где больше — лучше. Как превратить её в функцию потерь, где меньше — лучше, не меняя того, что она измеряет?

Подсказка: Подумайте о знаке.

Ответ: Поставить перед ней знак минус: доля 0,9 превращается в $-0,9$, а 0,8 — в $-0,8$, так что у лучшей модели теперь меньшее число, а больше ничего не изменилось.

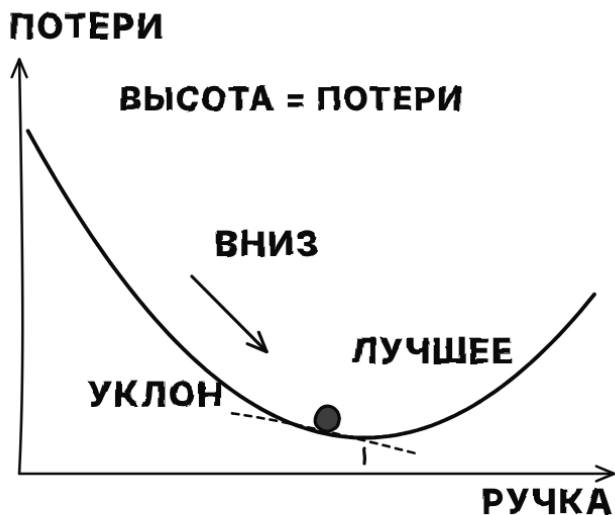
3. Своими словами объясните, почему приложение для проверки кожи, которое было очень точным на испытаниях, всё же может грубо ошибаться для некоторых людей.

Подсказка: Подумайте, кто был на обучающих и тестовых фотографиях.

Ответ: Если людей с тёмной кожей не было ни на фотографиях, на которых оно училось, ни в тестах, модель никогда не видела таких случаев и ей не на что было опереться; хорошая оценка на тестах относилась только к тем людям, которые были в данных. Мусор на входе — мусор на выходе.

Идти вниз по склону, затем учиться на ответах

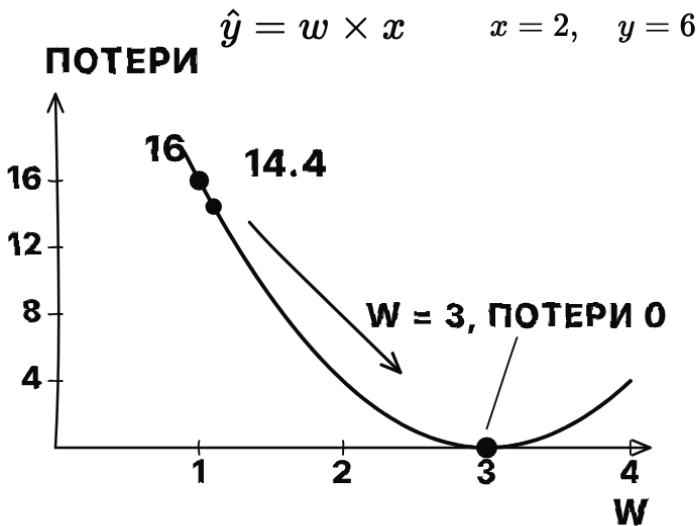
Встаньте с завязанными глазами на холмистом поле и попробуйте добраться до самой низкой точки. Вы не видите долины, но чувствуете землю под ногами. Чуть коснитесь ногой земли впереди, сзади, по сторонам и почувствуйте, куда она уходит вниз. Сделайте туда маленький шаг и снова пощупайте землю. Шаг за шагом вы оказываетесь на дне, так ни разу его и не увидев.



Нащупать уклон вниз

Это и есть четвёртая составляющая машинного обучения — алгоритм, который находит хорошие положения ручек. У нас уже есть данные, модель с ручками и функция потерь, которая говорит, насколько плохо работает модель. Не хватает лишь способа искать параметры, при которых потери как можно меньше. Методы, которыми чаще всего пользуются в глубоком обучении, построены на одной идее, называемой градиентным спуском. На каждом шаге он спрашивает про каждый параметр по очереди: если чуть-чуть сдвинуть этот параметр, потери на обучающей выборке вырастут или уменьшатся, и насколько? Затем он немного сдвигает каждый параметр в ту сторону, в которой потери становятся меньше. Высота поля — это потери, ваше положение — это положение ручек, а уклон, который вы чувствуете ногой, — то, что измеряет метод.

Проделаем это один раз с числами на модели с единственной ручкой w . Модель предсказывает время доставки, умножая расстояние на w , так что её предсказание $\hat{y} = w \times x$. У нас один обучающий пример: расстояние $x = 2$, доставка заняла $y = 6$ дней. Начнём с $w = 1$: предсказание равно 2, промах $2 - 6 = -4$, а квадрат ошибки 16. Теперь сдвинем ручку до $w = 1,1$: предсказание становится 2,2, промах $-3,8$, а потери около 14,4. Потери уменьшились, значит, увеличивать w — это направление вниз по склону, и метод делает шаг в эту сторону. Повторяйте, и w постепенно подбирается к 3, где предсказание равно 6, а потери равны нулю. Настоящая модель делает то же самое сразу с миллионами ручек, и на каждом шаге каждая сдвигается в своё направление вниз по склону. Языковую модель, стоящую за чат-ботом, обучали именно так — миллиардами таких маленьких шагов вниз по её ручкам.

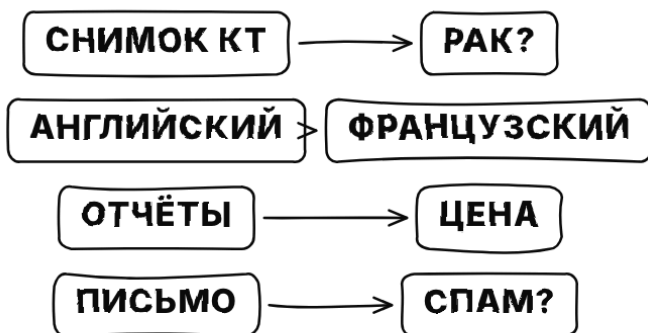


Одна ручка, один пример

Имея четыре составляющие, книга переходит к карте видов задач, которые может решать машинное обучение. Слово пробуждения было лишь одной из них, и авторы хотят дать нам общий словарь на всю оставшуюся книгу.



$$P(\text{метка} \mid \text{признаки})$$



Обучение с учителем

Первый и самый распространённый вид — обучение с учителем. Нам дан набор данных, в котором каждый пример — пара: признаки вместе с меткой, и мы должны построить модель, предсказывающую метку по признакам. Оно называется обучением с учителем, потому что мы, учителя, даём модели эти размеченные примеры, когда выставляются её ручки. На языке вероятностей мы обычно хотим знать шансы каждой возможной метки при тех признаках, которые видим. Обучение с учителем — не единственный вид, но на него приходится большинство успешных применений машинного обучения в промышленности. Причина в том, что очень многие важные задачи укладываются в схему «предсказать неизвестное по имеющимся данным». Книга приводит три: рак или не рак по снимку компьютерной

томографии; правильный перевод английского предложения на французский; цена акции в следующем месяце по финансовой отчётности этого месяца. Спам-фильтр вашей почты тоже подходит: признаки — это письмо, а метка — «спам» или «не спам».

Одно короткое описание охватывает всё обучение с учителем, но за ним стоит множество решений, которые зависят от того, каковы входы и выходы, насколько они велики и сколько их. Последовательность любой длины, например предложение, требует модели другого рода, чем список чисел фиксированной длины.

Затем книга набрасывает весь процесс от начала до конца. Собрать много примеров с известными признаками, выбрать случайное подмножество и получить для каждого истинную метку. Иногда метки уже есть в записях, например умер ли пациент в течение следующего года; иногда людям приходится платить за то, чтобы они их проставили, например разложив картинки по категориям. Эти входы с их метками и составляют обучающую выборку. Обучающая выборка поступает в алгоритм обучения с учителем, а это функция необычного рода: вы даёте ей целый набор данных, а она возвращает вам другую функцию — обученную модель. Наконец, в модель поступают новые входы, которых она никогда не видела, и то, что выходит, принимается за предсказание их меток. Рисунок в книге изображает это как цепочку: размеченные данные, затем алгоритм обучения, затем модель, а новые данные проходят через модель и становятся предсказаниями.

Запомним

Градиентный спуск находит хорошие параметры, нащупывая уклон: для каждой ручки он проверяет, как изменились бы потери на обучающей выборке после крошечного сдвига, и затем немного сдвигает каждую ручку вниз по склону. С одной ручкой и одним примером несколько таких шагов ведут потери от 16 к нулю. Обучение с учителем — это предсказание меток по признакам, обучение на примерах, которые приходят парами «признаки — метка». Большинство практических успехов машинного обучения относятся именно к этому виду. Алгоритм принимает размеченный набор данных и возвращает модель, которая затем предсказывает метки для новых входов.

Вопросы

1. При градиентном спуске небольшой сдвиг ручки вверх увеличивает потери на обучающей выборке. Что метод сделает с этой ручкой?
 - a. Оставит её точно там, где она есть
 - b. Сильно сдвинет её вверх, чтобы проскочить подъём
 - c. Немного сдвинет её вниз — в сторону уменьшения потерь
 - d. Сбросит её на случайное значение

2. Что в прогулке с завязанными глазами по холмистому полю соответствует потерям?
- a. Высота земли там, где вы стоите
 - b. Направление, куда вы смотрите
 - c. Длина вашего шага
 - d. Число сделанных шагов
3. Почему задача «спам или не спам по тексту письма» — пример обучения с учителем?
- a. Потому что фильтр находит группы писем без всяких меток
 - b. Потому что фильтр учится на наградах, которые получает со временем
 - c. Потому что у писем фиксированная длина
 - d. Потому что он учится на примерах, где письмо дано в паре со своей меткой
4. Что возвращает алгоритм обучения с учителем, когда ему дают размеченный набор данных?
- a. Одно предсказание
 - b. Обученную модель, которая сама является функцией
 - c. Очищенную версию набора данных
 - d. Список меток для обучающих примеров

Ответы

1. c) Для каждого параметра метод проверяет, как изменились бы потери после крошечного сдвига, и затем немного сдвигает его в ту сторону, где потери меньше.

2. а) Высота поля — это потери, ваше положение — положение ручек, а уклон, который вы чувствуете, — то, что измеряет метод.

3. d) Обучение с учителем учится предсказывать метки по признакам на примерах-парах «признаки — метка»; здесь письмо — это признаки, а «спам или не спам» — метка.

4. b) Вы даёте ему целый набор данных, а он возвращает другую функцию — обученную модель, которая затем предсказывает метки для новых входов.

Разобранные примеры

Один шаг градиентного спуска вручную Модель

предсказывает $\hat{y} = w \times x$. Единственный обучающий пример: $x = 2$, $y = 6$. Начав с $w = 1$, сравните квадрат ошибки при $w = 1$ и при $w = 0,9$ и решите, в какую сторону градиентный спуск сдвинет w .

При $w = 1$ предсказание равно $1 \times 2 = 2$, промах $2 - 6 = -4$, квадрат ошибки 16.

При $w = 0,9$ предсказание равно $0,9 \times 2 = 1,8$, промах $-4,2$, квадрат ошибки $4,2^2 = 17,64$.

Сдвиг w вниз увеличил потери с 16 до 17,64.

Значит, направление вниз по склону — вверх: градиентный спуск немного увеличивает w .

Ответ: При уменьшении w потери растут, поэтому w сдвигается вверх, к 3

Задачи

1. Для той же модели $\hat{y} = w \times x$ и примера $x = 2$, $y = 6$ каков квадрат ошибки при $w = 2$ и при $w = 4$? Где дно долины?

Подсказка: Сначала найдите предсказание, затем промах, затем возведите его в квадрат.

Ответ: При $w = 2$ предсказание 4, промах -2 , потери 4. При $w = 4$ предсказание 8, промах $+2$, потери тоже 4. Потери равны нулю при $w = 3$, где предсказание ровно 6, — посередине.

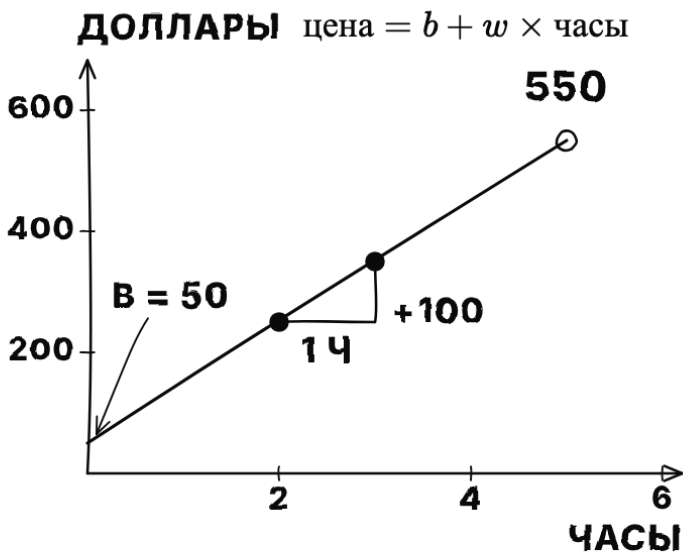
2. Своими словами опишите путь от кучи примеров до предсказания в обучении с учителем.

Подсказка: Начните со сбора примеров с известными признаками.

Ответ: Собрать примеры с известными признаками, выбрать случайное подмножество и получить их истинные метки — из имеющихся записей или от людей, которые их предоставляют. Эти пары составляют обучающую выборку. Алгоритм обучения принимает обучающую выборку и возвращает обученную модель, а новые входы, поданные в эту модель, выходят предсказаниями их меток.

Сколько? Регрессия

У вас засорилась канализация, сантехник тратит 3 часа на прочистку, и в счёте значится 350 долларов. Ваш друг вызывает того же сантехника, тот работает 2 часа и берёт 250 долларов. Теперь сосед спрашивает вас, сколько стоит вызов. Вы никогда не изучали машинное обучение, но уже можете ответить, и книга говорит, что в этом-то и суть: вы только что выполнили регрессию.



Два счёта задают прямую

Вот как вы это сделали, в три шага. Вы предположили, что больше часов стоит больше денег и что сантехник берёт фиксированную плату за выезд плюс фиксированную сумму за час. В виде формулы счёт равен $\text{цена} = b + w \times \text{часы}$, где b — плата за выезд, а w — почасовая ставка. Лишний час, с 2 до 3, добавил $350 - 250 = 100$ долларов, значит, $w = 100$. Тогда 2 часа стоили 200 из 250 долларов, и на выезд остаётся $b = 50$. Проверим по первому счёту: $50 + 3 \times 100 = 350$, ровно. Значит, работа на 5 часов должна стоить $50 + 5 \times 100 = 550$ долларов. Два числа b и w — это ручки модели, а найти их по примерам — это и есть обучение; поскольку формула задаёт прямую линию, это называется линейной регрессией.

Книга называет регрессию, пожалуй, самой простой для понимания задачей обучения с учителем и вводит её на таблице продаж домов. Каждая строка — один дом; каждый столбец — один признак: площадь в квадратных футах, число спален, число ванных и сколько минут идти пешком до центра города. Один дом — один пример, а его строка — его вектор признаков. В Нью-Йорке или Сан-Франциско, шутят авторы, если вы не руководите одним из технологических гигантов, ваше жильё может выглядеть как $[600, 1, 1, 60]$: маленькое, одна спальня, одна ванная, час пешком до центра. В Питтсбурге это может быть $[3000, 4, 3, 10]$. Каждый дом даёт четыре числа в одном и том же порядке — вектор фиксированной длины, — и большинству классических методов нужно именно это.

[кв. фт, спальни, ванны, мин]

ОДИН ДОМ

НЬЮ-ЙОРК

[600, 1, 1, 60]

ПИТТСБУРГ

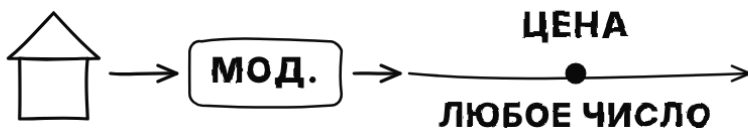
[3000, 4, 3, 10]

ФИКС. ДЛИНА: 4 ЧИСЛА

Дома как векторы признаков

Но регрессией задачу делают не признаки, а метка. Если вы покупаете дом и хотите узнать его справедливую цену, примеры — это прошлые продажи, а метки — фактически уплаченные цены. Когда метка может быть любым числом, пусть даже только в некотором диапазоне, задача — регрессия, и цель — модель, предсказания которой близки к истинным значениям. Книга даёт удобное правило: любой вопрос, начинающийся со «сколько», — скорее всего регрессия. Сколько часов займёт эта операция? Сколько дождя выпадет в этом городе за следующие шесть часов? Сколько дней этот пациент пробудет в больнице? Авторы вспоминают и приз Netflix 2009 года — миллион долларов тому, кто лучше всех предскажет, сколько звёзд пользователи поставят фильмам. Приложение доставки, которое

пишет «прибудет через 23 минуты», отвечает на вопрос регрессии каждый раз, когда вы его открываете.



СКОЛЬКО?

ЧАСЫ ОПЕРАЦИИ

ОСАДКИ

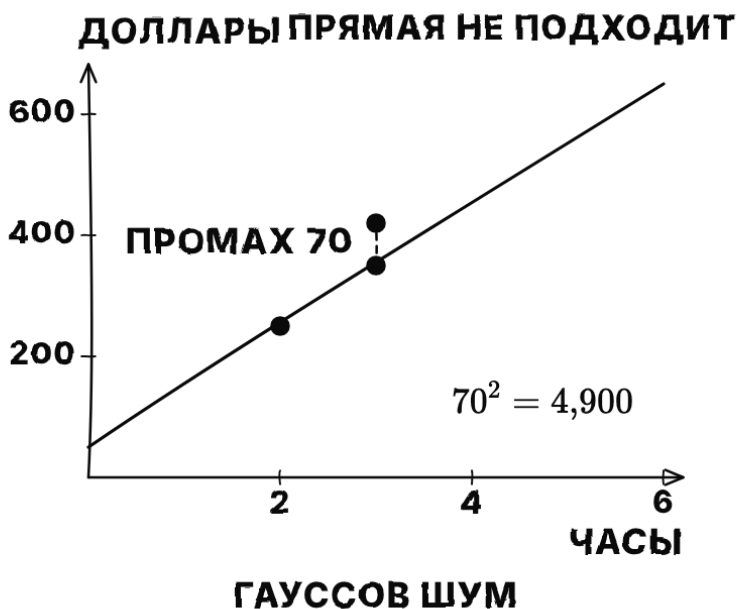
ДНИ В БОЛЬНИЦЕ

ОЦЕНКА ФИЛЬМА

Метка — число

Сантехник был лёгким случаем, потому что два счёта точно задали две ручки. Настоящие данные редко бывают так благосклонны. Третий счёт мог бы оказаться 420 долларов за 3 часа, потому что трубы были старше или работа пришлась на воскресенье, — то, чего одни только часы не учитывают. Тогда никакая прямая не проходит через все счета, и мы довольствуемся прямой, которая промахивается мимо них всех как можно меньше. Книга говорит, что в большинстве её глав «как можно меньше» будет измеряться квадратом ошибки — промахом, возведённым в квадрат и сложенным по всем

примерам. Если прямая предсказывает 350 для той работы за 420 долларов, промах равен 70, и он добавляет к потерям $70^2 = 4900$. Авторы обещают на будущее: выбрать квадрат ошибки — всё равно что предположить, что данные искажены случайным шумом особого колоколообразного вида, называемым гауссовым шумом.



Счёт мимо прямой

Опыт

Найдите два чека одного и того же такси или приложения поездок с разным расстоянием. Вычтите цены и разделите на разницу в километрах, чтобы получить цену за километр, затем вычтите, чтобы найти фиксированную плату за посадку. С помощью своей формулы предскажите третью поездку и сравните с настоящим чеком. Если не сошлось, то повышенный спрос или пробка — это фактор за пределами ваших признаков, а величина промаха — то, что измерил бы квадрат ошибки.

Запомним

Регрессия — это предсказание числа: любой вопрос «сколько». Каждый пример — вектор признаков, например площадь дома, число комнат и расстояние до центра, а его метка — число, например цена продажи. По двум счетам сантехника мы нашли модель $\text{цена} = 50 + 100 \times \text{часы}$: плата 50 долларов и 100 долларов в час, это линейная регрессия. Когда никакая прямая не подходит ко всем примерам, мы выбираем ту, у которой квадрат ошибки наименьший.

Вопросы

1. Что определяет, является ли задача регрессией?
 - а. То, что признаки — числа

- b. То, что метка может быть любым числом
 - c. То, что данные приходят таблицей
 - d. То, что модель — прямая линия
2. Какой из этих вопросов, по правилу книги, — регрессия?
- a. Ядовит ли этот гриб?
 - b. Какая цифра здесь написана?
 - c. Спам ли это письмо?
 - d. Сколько дней этот пациент пробудет в больнице?
3. Третий счёт сантехника не лежит на прямой, проходящей через первые два. Что мы делаем?
- a. Выбираем прямую, которая промахивается мимо всех счетов как можно меньше по квадрату ошибки
 - b. Выбрасываем третий счёт
 - c. Используем только самый свежий счёт
 - d. Проводим отдельную прямую для каждого счёта

Ответы

1. **b)** Регрессией задачу делают не признаки, а метка: когда метка может быть любым числом, задача — регрессия.
2. **d)** Любой вопрос, начинающийся со «сколько», — скорее всего регрессия.
3. **a)** Когда никакая прямая не проходит через все счета, мы довольствуемся той, что промахивается мимо них всех как можно меньше, по квадрату ошибки.

Разобранные примеры

Цены сантехника Сантехник берёт 350 долларов за 3 часа и 250 долларов за 2 часа. Считая, что цена = $b + w \times \text{часы}$, найдите плату за выезд b и почасовую ставку w и предскажите счёт за 5 часов.

Лишний час с 2 до 3 добавляет $350 - 250 = 100$ долларов, значит, $w = 100$.

За 2 часа почасовая часть $2 \times 100 = 200$, значит, $b = 250 - 200 = 50$.

Проверка по первому счёту: $50 + 3 \times 100 = 350$, совпадает.

За 5 часов: $50 + 5 \times 100 = 550$.

Ответ: $b = 50$ долларов, $w = 100$ долларов в час; 5 часов стоят 550 долларов

Задачи

1. Поездка на такси на 4 км стоит 9 евро, а на 10 км — 18 евро. Считая, что есть плата за посадку плюс цена за километр, найдите обе величины и предскажите поездку на 6 км.

Подсказка: Сколько добавляют шесть лишних километров?

Ответ: 6 лишних км добавляют $18 - 9 = 9$ евро, то есть 1,5 евро за км; плата за посадку $9 - 4 \times 1,5 = 3$ евро; поездка на 6 км стоит $3 + 6 \times 1,5 = 12$ евро.

2. Модель сантехника предсказывает 350 долларов за трёхчасовую работу, за которую на самом деле выставили 420 долларов. Сколько эта работа добавляет к квадрату ошибки и о чём говорит промах?

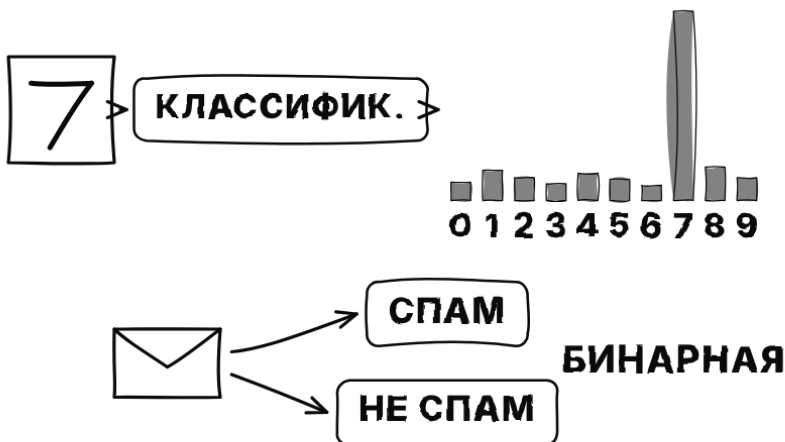
Подсказка: Вычитите, затем возведите в квадрат.

Ответ: Промах $420 - 350 = 70$, так что работа добавляет к потерям $70^2 = 4900$. Он говорит о факторе помимо часов, например старых

трубах или работе в воскресенье, который признаки модели не учитывают.

Который из? Классификация

Поднесите чек к камере банковского приложения, и через мгновение сумма и имя заполнятся сами. Где-то в промежутке приложение нарезало почерк на маленькие фрагменты, по символу в каждом, и для каждого фрагмента ответило на один вопрос: какой это символ? Этот вопрос — не «сколько». Нельзя сказать, что 7 «чуть больше», чем плохо написанная 1; фрагмент — один из фиксированного набора символов, и приложению нужно выбрать правильный.



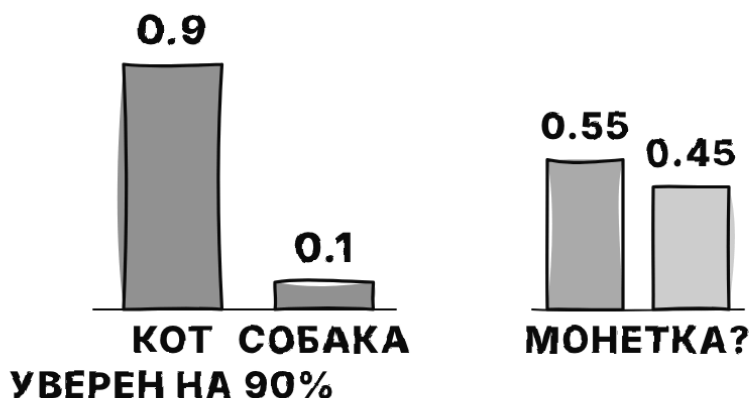
Который из? Десять классов

Книга называет такие задачи «который из?» классификацией и ставит их рядом с задачами «сколько?» — регрессией. Многие инструменты переносятся, но задача другая. Модель по-прежнему смотрит на признаки, например на значения яркости пикселей фрагмента. Но вместо числа она должна назвать категорию, или класс, из заранее заданного списка вариантов. Для рукописных цифр классов десять — цифры от 0 до 9. Модель, выдающую число, называли регрессором; модель, выдающую класс, называют классификатором.

В простейшем случае классов всего два, и это называется бинарной классификацией. Пример книги — картинки животных с метками «кошка» или «собака». Повседневный пример — ваша почта: каждое пришедшее письмо попадает в один из двух классов, «спам» или «не спам». Когда классов больше двух, задача — многоклассовая классификация, как чтение почерка, где классы идут от 0 до 9 и дальше через а, b, с и остальной алфавит.

Теперь поворот, который важен в дальнейшем. Оказывается, трудно обучить модель, которая может сказать только твёрдое «кошка» или твёрдое «собака»; книга обещает объяснить почему, когда станет более технической. Гораздо проще, если модель говорит на языке вероятностей. Получив картинку, модель даёт каждому классу вероятность — числа от 0 до 1, в сумме дающие 1. Она может сказать $P(\text{кошка}) = 0,9$, а значит, $P(\text{собака}) = 0,1$. Мы читаем это так: классификатор на 90 процентов уверен, что на картинке кошка. Насколько велика вероятность победившего класса, говорит о том, насколько модель уверена; 0,9 — уверенно, 0,55 — чуть лучше подброшенной монетки. Для обучения такой модели, говорит

книга, обычно используют не квадрат ошибки, а функцию потерь под названием перекрёстная энтропия; её название станет понятным, когда книга дойдёт до теории информации.



ПОТЕРИ: ПЕРЕКРЁСТНАЯ ЭНТРОПИЯ

Вероятности, а не приговоры

Затем книга делает замечание, которое легко упустить: самый вероятный класс — не всегда тот, по которому стоит действовать. На её рисунке — красивый гриб, растущий на заднем дворе. Предположим, вы обучили классификатор по фотографиям отличать ядовитые грибы от безопасных, и для этого гриба он говорит, что вероятность бледной поганки равна 0,2. Значит, он на 80 процентов уверен, что гриб безопасен. Съели бы вы его? Ответ авторов: только безумец.



ПОГАНКА 0,2
БЕЗОПАСЕН 0,8

СЪЕСТЬ

$$0.2 \times \infty + 0.8 \times 0 = \infty$$

ВЫБРОСИТЬ ✓

$$0.2 \times 0 + 0.8 \times 1 = 0.8$$

Ожидаемая цена гриба

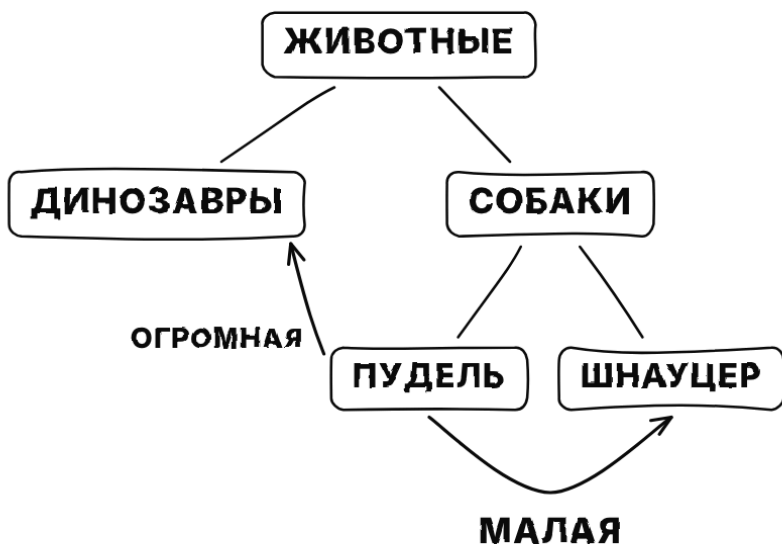
Чтобы понять почему, взвесьте каждое действие: во что оно может обойтись, умноженное на то, насколько это вероятно. Назначим каждому исходу цену. Съесть бледную поганку — значит умереть; цена так велика, что книга записывает её как бесконечность, ∞ . Съесть безопасный гриб ничего не стоит, 0; выбросить безопасный стоит потерянного ужина, скажем 1; выбросить бледную поганку стоит 0. Ожидаемая цена того, чтобы съесть гриб: $0,2 \times \infty + 0,8 \times 0 = \infty$. Ожидаемая цена того, чтобы его выбросить: $0,2 \times 0 + 0,8 \times 1 = 0,8$. Меньше одного потерянного ужина против верной беды: вы выбрасываете гриб. Решение зависит и от вероятностей, и от того, насколько хорош или плох каждый исход. И книга добавляет, что любой знаток

грибов это подтвердит: гриб на рисунке действительно бледная поганка.

Опыт

Возьмите приложение погоды, которое обещает 30 процентов вероятности дождя, и решите, брать ли зонт. Назначьте цену каждому исходу: промокнуть в хорошем костюме может стоить 10, носить с собой ненужный зонт — 1. Не брать зонт стоит в среднем $0,3 \times 10 = 3$; брать — не больше 1. Поэтому вы берёте зонт, хотя «дождя не будет» — более вероятный класс, точно так же, как с грибом.

Классификация может быть ещё замысловатее. Иногда классы образуют родословное дерево, и тогда не все ошибки одинаково плохи. Если модели суждено ошибиться, мы предпочли бы, чтобы она выбрала близкого родственника, а не что-то далёкое. Это называется иерархической классификацией, и книга вспоминает Линнея, который разложил животных по такому дереву видов и семейств. Принять пуделя за шнауцера, две породы собак, — небольшая оплошность; принять пуделя за динозавра должно дорого обойтись модели.



Близкие и далёкие ошибки

Но какое дерево важно, зависит от того, для чего нужна модель. Гремучие и подвязочные змеи стоят близко на древе жизни. Но модель, назвавшая гремучую змею безобидной подвязочной, может стоять кому-то жизни, так что для приложения туриста эти две должны быть далеко друг от друга. Приложение для распознавания растений в вашем телефоне стоит перед тем же выбором: спутать два вида ромашек почти не важно, а спутать съедобную черешшу с ядовитым ландышем — очень важно.

Запомним

Классификация отвечает на вопрос «который из?»: она относит каждый пример к одному из фиксированного набора классов, а регрессия отвечает на вопрос «сколько?». При двух классах это бинарная классификация, например «спам или не спам»; при большем числе — многоклассовая, например чтение цифр и букв. Классификаторы обычно дают вероятность для каждого класса, и величина наибольшей вероятности показывает, насколько они уверены; их обычная функция потерь — перекрёстная энтропия. Самый вероятный класс — не всегда тот, по которому надо действовать: взвешивайте каждое действие по вероятностям и ценам, как с грибом, который на 80 процентов безопасен, но всё же не стоит того, чтобы его есть. Когда классы образуют иерархию, одни ошибки хуже других, и какие именно, зависит от того, как будет использоваться модель.

Вопросы

1. Почему распознавание рукописной цифры — задача классификации, а не регрессии?
 - а. Потому что ответ должен быть одним из фиксированного набора символов, а не величиной
 - б. Потому что в изображении слишком много пикселей
 - с. Потому что цифры всегда пишут плохо

- d. Потому что регрессия не может использовать значения пикселей как признаки
2. Классификатор даёт $P(\text{кошка}) = 0,55$ и $P(\text{собака}) = 0,45$. О чём это говорит?
- a. На картинке точно кошка
 - b. На картинке и кошка, и собака
 - c. Он склоняется к кошке, но уверен лишь чуть больше, чем при подбрасывании монетки
 - d. Классификатор сломан, ведь числа должны быть 0 или 1
3. Классификатор грибов на 80 процентов уверен, что гриб безопасен. Почему его всё равно стоит выбросить?
- a. Потому что классификатор обучали на слишком малом числе фотографий
 - b. Потому что 80 процентов ниже доли ошибок
 - c. Потому что безопасные грибы редки
 - d. Потому что ожидаемая цена того, чтобы его съесть, когда один из исходов — смерть, гораздо больше потерянного ужина
4. Почему приложение туриста о змеях может считать гремучую и подвязочную змей далёкими, хотя это близкие родственники?
- a. Потому что они совсем не похожи
 - b. Потому что их спутать может стоить кому-то жизни, а нужная иерархия зависит от цели
 - c. Потому что бинарная классификация не справляется со змеями

d. Потому что древо жизни — не иерархия

Ответы

1. a) Нельзя сказать, что 7 чуть больше плохо написанной 1; фрагмент — один из фиксированного набора символов, и это вопрос «который из?».

2. c) Величина победившей вероятности показывает, насколько модель уверена; 0,9 — уверенно, 0,55 — чуть лучше подброшенной монетки.

3. d) Съесть гриб стоит в среднем $0,2 \times \infty + 0,8 \times 0 = \infty$, а выбросить — $0,2 \times 0 + 0,8 \times 1 = 0,8$; решение зависит от вероятностей и цен вместе.

4. b) Какое дерево важно, зависит от того, для чего нужна модель; назвать гремучую змею безобидной подвязочной может быть смертельно.

Разобранные примеры

Решение о грибе Классификатор даёт вероятность 0,2, что гриб — бледная поганка. Цены: съесть поганку — ∞ , съесть безопасный — 0, выбросить поганку — 0, выбросить безопасный — 1. Найдите ожидаемую цену того, чтобы съесть гриб и чтобы его выбросить.

Вероятность того, что гриб безопасен, равна $1 - 0,2 = 0,8$.

Съесть: $0,2 \times \infty + 0,8 \times 0 = \infty$, поскольку любой шанс бесконечной цены делает сумму бесконечной.

Выбросить: $0,2 \times 0 + 0,8 \times 1 = 0,8$.

Выбираем действие с меньшей ожидаемой ценой.

Ответ: Съесть — ∞ , выбросить — 0,8; выбрасываем

Задачи

1. Приложение погоды даёт 30 процентов вероятности дождя. Промокнуть стоит 10, носить ненужный зонт — 1, остаться сухим с зонтом под дождём — 1. Какова ожидаемая цена каждого выбора?

Подсказка: Без зонта вы платите только если идёт дождь; с зонтом платите 1 в любом случае.

Ответ: Без зонта: $0,3 \times 10 + 0,7 \times 0 = 3$. С зонтом: $0,3 \times 1 + 0,7 \times 1 = 1$. Берите зонт, хотя отсутствие дождя вероятнее.

2. Спам-фильтр выдаёт для письма $P(\text{спам}) = 0,7$. Чему равна $P(\text{не спам})$, и это бинарная или многоклассовая классификация?

Подсказка: Вероятности всех классов в сумме дают 1.

Ответ: $P(\text{не спам}) = 1 - 0,7 = 0,3$, и при двух классах это бинарная классификация.

Много меток сразу и лучшее — первым

Покажите классификатору «кошка или собака» картинку из старой сказки о бременских музыкантах. На рисунке в книге осёл, у него на спине собака, на собаке кошка, а на самом верху петух, и всё это на фоне деревьев. Классификатор может ответить только «кошка» или «собака», так что что бы он ни сказал, он ошибётся; правильный ответ — «кошка, и собака, и осёл, и петух». Сделать модель точнее не поможет, замечает книга: задача была поставлена неверно.



МНОГО МЕТОК

КОШКА ✓

СОБАКА ✓

ОСЁЛ ✓

ПЕТУХ ✓

ОБЛАКО — AWS

МО — GPU

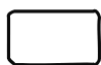
Много меток сразу

Когда один пример может одновременно относиться к нескольким классам, задача называется классификацией с несколькими метками. Классы больше не исключают друг друга; модель отвечает «да» или «нет» для каждой метки отдельно. Это естественная форма разметки тегами. У записи в технологическом блоге может быть от пяти до десяти тегов, например «машинное обучение», «языки программирования», «Linux» или «облачные вычисления». Теги ещё и ходят вместе: запись с тегом «облачные вычисления» часто упоминает AWS, облачный сервис Amazon, а запись с тегом «машинное обучение» часто упоминает GPU. Хорошая система разметки может использовать эти закономерности. Когда приложение для фотографий подписывает один снимок сразу «пляж», «закат» и «собака», оно выполняет классификацию с несколькими метками.

Иногда список возможных тегов огромен. В Национальной медицинской библиотеке США эксперты присваивают каждой статье в базе PubMed набор тегов из списка MeSH, Medical Subject Headings, примерно из 28 000 пунктов. Хорошие теги позволяют исследователям найти всё, что написано по вопросу, но ручная разметка медленная, и статьи обычно ждут около года. Машинное обучение может предлагать предварительные теги, пока их не проверит человек, и соревнование BioASQ уже много лет проводится именно по этой задаче.

Следующий вид задач — о порядке, а не о классах. Когда вы ищете в интернете, подходящими могут оказаться тысячи страниц. Настоящий вопрос — какие из них поставить для вас наверх. Естественный способ — дать каждому кандидату оценку и показать первыми самые высокие.

PAGERANK

 **0.4**

 **0.9**

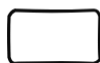
 **0.2**

 **0.7**

 **0.5**

ПЕРВЫМ

 **0.9**

 **0.7**

 **0.5**

 **0.4**

 **0.2**

Оценить, затем ранжировать

Книга вспоминает PageRank, изначальный секретный ингредиент поисковика Google, как раннюю систему оценок. Как ни странно, его оценка не зависела от того, что вы набрали. Сначала простой фильтр отбирал страницы, подходящие под запрос, а затем PageRank ставил более авторитетные страницы — грубо говоря, те, на которые ссылаются многие другие, — впереди остальных. Сегодня, говорит книга, поисковики выучивают свои оценки с помощью машинного обучения и моделей поведения людей, так что оценка зависит от запроса, и этой теме посвящены целые научные конференции.

Опыт

Поищите в своей почте слово вроде «счёт» и посмотрите на порядок результатов. Затем поищите «счёт март» и посмотрите, как изменится порядок. Набор подходящих писем сужается, но меняется и ранжирование, потому что оценка теперь зависит от большей части вашего запроса. В этом и разница между отбором подходящего и ранжированием самого важного.

Запомним

Когда пример может одновременно относиться к нескольким классам, как фотография четырёх животных, задача — классификация с несколькими метками, а её обычная форма — разметка тегами. Теги склонны появляться вместе, а список тегов может быть огромным, как 28 000 медицинских рубрик, по которым индексируется PubMed. Поиск — это ранжирование: дать каждому подходящему кандидату оценку и показать первыми самые высокие. PageRank оценивал страницы, не глядя на запрос; современный поиск выучивает оценки, зависящие от запроса.

Вопросы

1. Классификатору «кошка или собака» показывают бременских музыкантов. Почему повышение его точности не поможет?
 - a. На картинке слишком много деревьев
 - b. Задача поставлена неверно: на картинке сразу несколько классов
 - c. Классификатор никогда не обучали на сказках
 - d. Точность важна только для регрессии
2. Что делает модель для каждой возможной метки в классификации с несколькими метками?
 - a. Отвечает «да» или «нет» для этой метки отдельно
 - b. Выбирает одну самую вероятную метку
 - c. Ранжирует метки и оставляет только верхнюю
 - d. Предсказывает число — сколько всего меток
3. Каков, по словам лекции, главный вопрос в веб-поиске?
 - a. Подходит ли под запрос хоть какая-то страница
 - b. Сколько страниц в интернете
 - c. Какие из множества подходящих страниц поставить наверх
 - d. Какие теги присвоить каждой странице
4. Чем PageRank отличался от оценок, которые поисковики выучивают сегодня?
 - a. Он использовал машинное обучение, а сегодняшние оценки — нет
 - b. Он ранжировал страницы по длине
 - c. Он работал только с медицинскими статьями

d. Его оценка не зависела от запроса, а сегодняшние зависят

Ответы

1. **b)** Правильный ответ — кошка, собака, осёл и петух; модель, которая должна выбрать один класс, ошибётся, что бы ни сказала, поэтому задача должна быть классификацией с несколькими метками.
2. **a)** Классы больше не исключают друг друга; модель отвечает «да» или «нет» для каждой метки отдельно.
3. **c)** Подходящими могут быть тысячи страниц; настоящий вопрос — какие поставить наверх, а для этого кандидатов оценивают и показывают первыми самые высокие оценки.
4. **d)** Оценка PageRank не зависела от того, что вы набрали; фильтр отбирал подходящие страницы, а PageRank ставил авторитетные первыми. Сегодняшние выученные оценки зависят от запроса.

Задачи

1. Для каждой задачи скажите, многоклассовая это классификация или с несколькими метками, и почему: прочитать, какая цифра написана в клетке; разметить новостную статью темами; перечислить всех животных на фотографии фермы.

Подсказка: Спросите себя, может ли у одного примера быть больше одного правильного ответа.

Ответ: Цифра — многоклассовая: в каждой клетке ровно одна цифра. Разметка статьи — с несколькими метками: она может быть о нескольких темах. Фотография фермы — с несколькими метками: на ней может быть сразу несколько животных.

2. Своими словами объясните разницу между отбором и ранжированием в поиске на собственном примере.

Подсказка: Одно решает, что попадёт в список, другое — в каком порядке.

Ответ: Отбор решает, какие объекты вообще подходят, например все письма со словом «счёт». Ранжирование даёт каждому подходящему объекту оценку и ставит лучшие первыми, например самый свежий или лучше всего подходящий счёт наверху.

Лента, которая изучает вас

Откройте видеоприложение на своём телефоне и на телефоне друга, рядом. Поля поиска одинаковые, а главные страницы совсем не похожи. Ваша может быть полна кулинарии и футбола, а его — кошек и старых фильмов, хотя никто ничего не набирал.



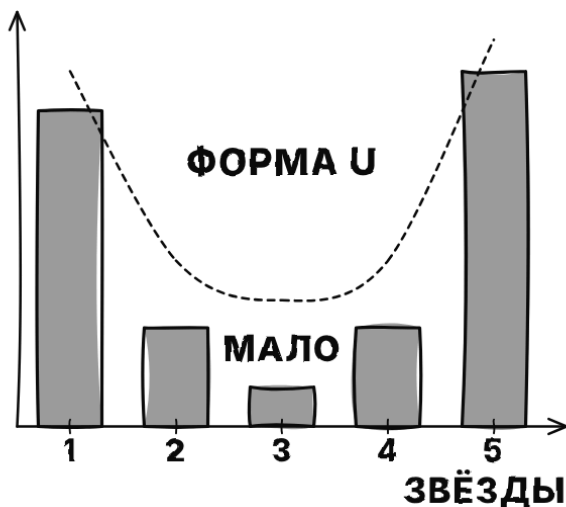
Одно приложение, разные ленты

Это рекомендательные системы — следующий вид задач в книге. Они близкие родственники поиска: и там и там цель — показать список объектов, важных для пользователя. Разница — в персонализации. Для фильмов, говорят авторы, страница

поклонника научной фантастики и страница любителя комедий с Питером Селлерсом должны выглядеть совсем по-разному, и то же верно для магазинов, музыки и новостей.

Система учится на том, что ей сообщают люди, двумя способами. Явная обратная связь высказана вслух: оценки звёздами и отзывы на Amazon, IMDb или Goodreads. Неявная обратная связь считывается по поведению, например по тому, что вы пропускаете песню в плейлисте. Её труднее истолковать: пропуск может значить, что песня вам не нравится, а может — лишь то, что она была не к месту. В простейшем варианте система учится предсказывать оценку для каждой пары «человек — объект», скажем, сколько звёзд вы бы поставили или вероятность того, что вы это купите. Затем для вас она выбирает объекты с самыми высокими предсказанными оценками и показывает их. Настоящие системы используют гораздо больше, например подробные записи того, что вы делаете, и того, каков каждый объект. На рисунке в книге показаны книги по глубокому обучению, которые Amazon рекомендовал одному из авторов, Астону.

Рекомендательные системы стоят огромных денег, но книга предупреждает, что если наивно строить их поверх предсказательной модели, у них будут глубокие изъяны. Первый — обратная связь цензурирована: мы видим лишь часть мнений. Люди в основном оценивают то, что вызвало у них сильные чувства. По шкале от одной до пяти звёзд много единиц и много пятёрок и странно мало троек — не потому, что средних фильмов мало, а потому, что люди редко утруждают себя оценкой среднего фильма.



Середину оценивают немногие

Опыт

Откройте отзывы о любом популярном товаре в интернет-магазине и посмотрите на столбчатую диаграмму оценок. Посчитайте долю отзывов в одну и в пять звёзд и долю отзывов в три звезды. Обычно вы увидите провал посередине, форму буквы U. Этот провал и есть цензурированная обратная связь, о которой говорит книга: тихая середина молчит.

Второй изъян — петля. То, что люди покупают сегодня, — отчасти то, что им показала текущая рекомендательная система.

Предположим, система продвигает один товар чуть больше других. Его видит больше людей, поэтому его больше покупают, поэтому следующий цикл обучения считает его лучше, поэтому его рекомендуют ещё чаще. Товар поднимается потому, что его продвигали, а не потому, что он лучший. Песня, которую музыкальное приложение всё время ставит в вашу подборку, становится «популярной» отчасти потому, что её туда поставило приложение. Книга говорит, что как справляться с цензурированием, стимулами и такими петлями обратной связи — всё ещё открытый вопрос исследований.



Петля обратной связи

Запомним

Рекомендательные системы похожи на поиск, но персонализированы: разные люди видят разные списки. Они учатся на явной обратной связи, например оценках, и на неявной, например пропусках, и предсказывают оценку для каждого человека и объекта. Обратная связь цензурирована, потому что люди в основном оценивают то, что любят или ненавидят, поэтому оценок в три звезды мало. Рекомендации влияют на то, что покупают люди, поэтому система может создать петлю обратной связи, которая продолжает продвигать одни и те же товары.

Вопросы

1. Что главным образом отличает рекомендательную систему от поисковика?
 - a. Рекомендательная система никогда не использует оценки
 - b. Рекомендательная система персонализирует список для каждого пользователя
 - c. Рекомендательная система показывает только то, что никто не оценивал
 - d. Рекомендательная система работает только для фильмов
2. Почему на пропуске песни учиться труднее, чем на оценке в одну звезду?

- a. Пропуски никогда не записываются
 - b. Пропуски бывают только в музыкальных приложениях
 - c. Пропуск может означать неприязнь, а может — лишь то, что песня была не к месту
 - d. Пропуск всегда значит, что песня очень понравилась
3. Почему оценок в три звезды странно мало?
- a. Люди редко утруждают себя оценкой того, к чему относятся прохладно
 - b. Средних товаров мало
 - c. Рекомендательные системы удаляют оценки в три звезды
 - d. На шкале оценок нет среднего значения
4. Рекомендательная система магазина продвигает один товар чуть больше других. Что может случиться за несколько циклов обучения?
- a. Товар исчезнет из магазина
 - b. Оценки товара станут цензурированными
 - c. Ничего, ведь обучение использует только оценки
 - d. Товар больше покупают, считают лучше и продвигают ещё сильнее

Ответы

1. **b)** Оба показывают список важных объектов; разница в персонализации, поэтому любитель фантастики и поклонник Питера Селлерса видят разные страницы.
2. **c)** Неявная обратная связь считывается по поведению, и пропуск может значить, что песня вам не нравится, а может — лишь то, что она

была не к месту.

3. а) Обратная связь цензурирована: люди в основном оценивают то, что вызывает сильные чувства, поэтому тихая середина редко высказывается.

4. d) Его видит больше людей, поэтому больше покупают, поэтому следующий цикл считает его лучше и рекомендует ещё чаще: петля обратной связи.

Задачи

1. Распределите по явной и неявной обратной связи: отзыв на книгу в пять звёзд; остановка фильма через десять минут; добавление песни в избранное нажатием на сердечко; покупка товара.

Подсказка: Явная обратная связь высказывает мнение; неявную приходится угадывать по поведению.

Ответ: Явная: отзыв в пять звёзд и нажатие на сердечко — оба прямо высказывают мнение. Неявная: остановка фильма и покупка товара — мнение приходится угадывать по действию.

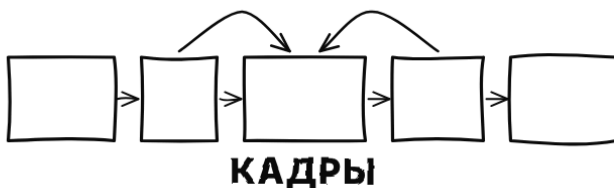
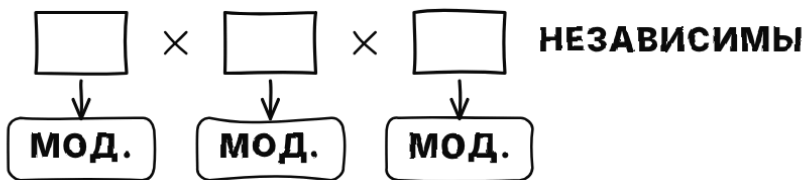
2. Своими словами объясните, почему рекомендательная система, обученная на том, что люди купили в прошлом месяце, может просто выучить свои же прежние решения.

Подсказка: Спросите себя, откуда взялись покупки прошлого месяца.

Ответ: Многие покупки прошлого месяца случились потому, что рекомендательная система показала эти товары. Обучение на них делает эти товары лучше, чем они есть, и система рекомендует их ещё чаще: она учится на данных, которые сама сформировала, — это петля обратной связи.

Когда важен порядок: последовательности

Остановите футбольный матч на одном кадре: игрок в прыжке, мяч у его ноги. Он бьёт по воротам, отдаёт пас или падает после фола? По одному кадру не понять; по секунде до и секунде после вы понимаете сразу. Смысл каждого мгновения зависит от мгновений вокруг него.



ВХОД И ВЫХОД — ПОСЛЕДОВАТЕЛЬНОСТИ

Независимые входы и последовательности

Всё, что было в курсе до сих пор, это игнорировало. Дом приходил с фиксированным набором признаков, четырьмя

числами, и получал одну цену. Картинка фиксированного размера получала по одной вероятности на каждый класс из фиксированного списка. Оценку звёздами предсказывали лишь по номеру пользователя и номеру товара. И как только модель ответила для одного примера, она полностью забывала его, прежде чем приходил следующий. Книга говорит, что за этим стоит предположение: один пример ничего не говорит о следующем, так что помнить незачем.

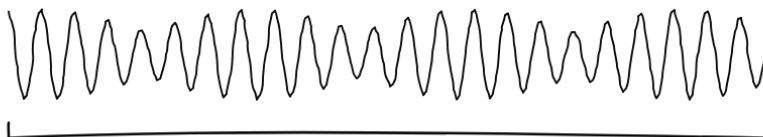
Видео нарушает это предположение дважды. В каждом ролике может быть разное число кадров, так что фиксированной длины нет. И догадка об одном кадре гораздо лучше, если посмотреть на кадры до и после него. С языком то же самое: предложение может быть любой длины, а слово понимают по словам вокруг него. И с медициной тоже. Представьте модель в реанимации, которая должна поднимать тревогу всякий раз, когда риск того, что пациент умрёт в ближайшие 24 часа, поднимается выше некоторого уровня. Было бы нелепо, если бы она каждый час забывала историю пациента и судила только по последним давлению и пульсу.

Такие задачи называют обучением на последовательностях, и книга относит их к самым захватывающим применениям машинного обучения. Модель должна принимать последовательность входов, или выдавать последовательность выходов, или и то и другое. Когда и вход, и выход — последовательности переменной длины, это называется обучением «последовательность в последовательность»; перевод предложений и превращение речи в текст — стандартные примеры. Книга выделяет несколько частных случаев.

ВЫРОВНЕННЫ

TOM HAS DINNER IN
ENT — — —

WASHINGTON WITH SALLY
ENT — ENT



48 000 ОТСЧЁТОВ



ШЕСТЬ СЛОВ

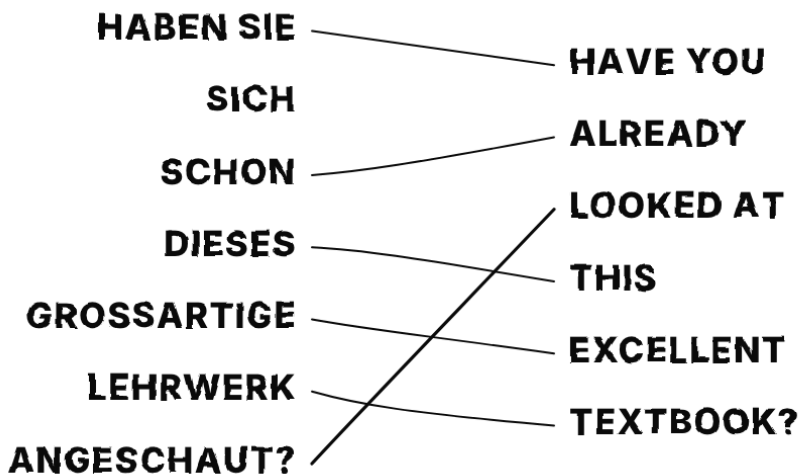
Выровненные метки, невыровненный звук

Первый — разметка и разбор, где каждое слово текста получает пометку. Здесь вход и выход выровнены: по одной пометке на слово, в том же порядке, что и слова. При разметке частей речи каждое слово получает свою грамматическую роль, например «существительное» или «глагол». В более простом варианте каждое слово помечается тем, входит ли оно в имя человека, места или организации. Пример книги — предложение «Tom has dinner in Washington with Sally», где Tom, Washington и Sally помечены как имена, а остальные четыре слова — нет. Семь слов на входе, семь пометок на выходе.

Второй — автоматическое распознавание речи: на входе запись, на выходе написанные слова. Здесь длины сильно различаются.

Звук обычно записывают с частотой 8 кГц или 16 кГц, то есть восемь или шестнадцать тысяч отсчётов в секунду. Предложение из шести слов, произнесённое за три секунды при 16 кГц, — это $3 \times 16\,000 = 48\,000$ отсчётов. Значит, 48 000 чисел должны превратиться в шесть слов, примерно по 8000 отсчётов на слово, и аккуратного взаимно однозначного соответствия между кусками звука и буквами нет. На рисунке в книге слова «Deer Learning» растянуты вдоль звуковой волны, и каждый звук размазан по длинному её участку. Люди легко понимают речь даже по плохой телефонной линии; для компьютеров, говорит книга, это серьёзнейшая задача. Голосовой помощник, который понимает вас сквозь уличный шум, решает именно эту задачу.

Третий случай — та же задача наоборот: синтез речи. На входе короткий текст, на выходе длинный звуковой файл. Когда навигатор произносит «через двести метров поверните налево», несколько слов превращаются в десятки тысяч отсчётов.



Перевод переставляет слова

Четвёртый — машинный перевод, и он приносит новую трудность. В речи звуки и слова хотя бы идут в одном порядке. В переводе два предложения могут быть разной длины, а соответствующие части — стоять в разном порядке. Книга показывает это на немецком, который любит ставить глагол в конец. «Haben Sie sich schon dieses grossartige Lehrwerk angeschaut?» значит «Вы уже посмотрели этот прекрасный учебник?» Если переводить слово за словом, сохраняя немецкий порядок, получится «Have you yourself already this excellent textbook looked at?» — это неправильное выравнивание. Глагол «looked at» должен переместиться из конца немецкого предложения в середину английского. Чат-бот, который

переводит вам сообщение, должен выучить такие перестановки для каждой пары языков.

Опыт

Возьмите короткое предложение на знакомом вам языке, помимо русского, или попросите его у друга. Запишите его по-русски слово за словом, сохраняя исходный порядок, а под ним — хороший русский перевод. Соедините линиями слова, которые значат одно и то же. Там, где линии пересекаются, порядок изменился; именно это пересечение делает перевод невыровненным.

В конце книга бегло касается более трудных случаев. Определить, в каком порядке человек читает веб-страницу, — задача в двух измерениях, а не вдоль одной линии. А в диалоге, чтобы решить, что сказать дальше, нужны знания о мире и память о том, что было сказано задолго до этого. Это, говорят авторы, всё ещё области активных исследований.

Запомним

Во многих прежних задачах входы были фиксированного размера, а каждый пример считался независимым и забывался после ответа. Обучение на последовательностях работает с входами или выходами, которые являются последовательностями — видео, предложения или история пациента, — где важны контекст и длина. При разметке вход и выход выровнены: по одной пометке на слово, по порядку. Распознавание речи превращает десятки тысяч отсчётов в секунду в несколько слов; синтез речи делает обратное. Машинный перевод не выровнен: длины различаются, а соответствующие части могут идти в другом порядке, как немецкий глагол в конце.

Вопросы

1. Какое предположение прежних задач нарушает видео?
 - a. Что каждый пример независим и его можно забыть после ответа
 - b. Что признаки — числа
 - c. Что метки обучающих примеров известны
 - d. Что у моделей есть параметры
2. Почему разметку слов предложения называют выровненной задачей?
 - a. Выход гораздо короче входа

- b. Слова могут идти в любом порядке
 - c. На каждое слово одна пометка, в том же порядке, что и слова
 - d. Пометки получают только имена
3. Почему при распознавании речи нет взаимно однозначного соответствия между звуком и буквами?
- a. Потому что запись текста длиннее аудиозаписи
 - b. Потому что десятки тысяч отсчётов превращаются всего в несколько слов
 - c. Потому что речь всегда зашумлена
 - d. Потому что каждое слово длится ровно один отсчёт
4. Что делает машинный перевод невыровненным?
- a. Выход всегда короче входа
 - b. Каждое слово нужно пометить как имя
 - c. Переводят со звука
 - d. Длины различаются, а соответствующие части могут идти в другом порядке

Ответы

1. а) Прежние задачи предполагали, что один пример ничего не говорит о следующем, поэтому модель забывала каждый; в видео число кадров меняется, и каждый понимают по соседним.
2. с) При разметке вход и выход выровнены: по одной пометке на слово, в том же порядке; семь слов на входе, семь пометок на выходе.
3. b) При 16 кГц три секунды речи — это 48 000 отсчётов, превращающихся в шесть слов, примерно по 8000 отсчётов на слово.

4. d) В переводе два предложения могут быть разной длины, а соответствующие части — стоять в разном порядке, как немецкий глагол в конце, переезжающий в середину английской фразы.

Разобранные примеры

Отсчётов на слово Звук записывается с частотой 16 кГц, то есть 16 000 отсчётов в секунду. Предложение из шести слов произносится за три секунды. Сколько это отсчётов и примерно сколько на слово?

16 кГц означает 16 000 отсчётов каждую секунду.

Три секунды дают $3 \times 16\,000 = 48\,000$ отсчётов.

Их делят шесть слов, так что на каждое приходится около $48\,000 \div 6 = 8000$ отсчётов.

Значит, тысячам входных чисел соответствует одно выходное слово: длины сильно различаются.

Ответ: 48 000 отсчётов, около 8000 на слово

Задачи

1. Телефон записывает с частотой 8 кГц. Сколько отсчётов в пятисекундном голосовом сообщении?

Подсказка: 8 кГц — это 8000 отсчётов каждую секунду.

Ответ: $5 \times 8000 = 40\,000$ отсчётов.

2. Разметьте предложение «Anna flew from Paris to Rome» по простой схеме пометки имён и скажите, сколько слов на входе и сколько пометок на выходе.

Подсказка: Люди и места — это имена.

Ответ: Anna, Paris и Rome помечены как имена; flew, from и to — нет. На входе шесть слов и на выходе шесть пометок, потому что

разметка выровнена.

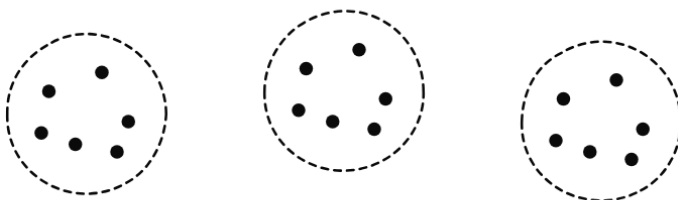
Учиться без учителя

Откройте фотогалерею в телефоне и поищите «пляж». А теперь посмотрите альбом «Люди», где телефон собрал все фотографии одного и того же друга в одну стопку. Никто никогда не говорил ему имени вашего друга и того, какие фотографии относятся друг к другу; группы он нашёл сам. Это другой вид обучения, чем всё, что было до сих пор.

Книга объясняет разницу на примере двух начальников. Обучение с учителем — это как работать у очень властного начальника, который стоит у вас за спиной и в каждой ситуации говорит, что именно делать, пока вы не научитесь ему подражать. Скучно, признают авторы, но легко: заметь закономерность и подражай. Противоположный начальник понятия не имеет, чего хочет. Он сваливает вам на стол огромную кучу данных и говорит: «Займитесь с этим наукой о данных!» Звучит расплывчато, потому что это и есть расплывчато, и специалисту по данным лучше к этому привыкнуть. Такие задачи без меток называются обучением без учителя, и единственное ограничение на вопросы, которые можно задать, — ваша фантазия.

Книга приводит несколько таких вопросов. Могут ли несколько типичных примеров обобщить все данные — например, разложить кучу фотографий на пейзажи, собак, младенцев, кошек и горные вершины или разбить пользователей по тому, как они ходят по сайтам? Это кластеризация. Альбом «Люди» в вашем телефоне и музыкальное приложение, объединяющее слушателей со схожим вкусом, — оба занимаются

кластеризацией. Могут ли несколько чисел передать то, что важно в каждом примере? Полёт мяча хорошо описывается его скоростью, размером и массой, а портные описывают человеческое тело для подгонки одежды горсткой мерок. Это оценка подпространства, а когда числа зависят от данных линейно, это называется методом главных компонент.



КЛАСТЕРЫ

ПО НЕСКОЛЬКУ ЧИСЕЛ.

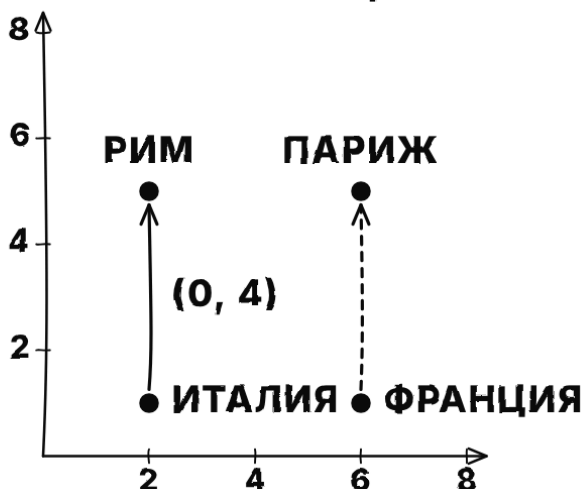


Группы, найденные без меток

Третий вопрос — можно ли расположить слова и вещи точками в пространстве так, чтобы их отношения превратились в арифметику. Пример книги — Рим — Италия + Франция = Париж. Вот придуманная картинка того, как это может работать, где каждое слово — точка с двумя числами. Скажем, Рим находится в точке (2, 5), Италия — в (2, 1), а Франция — в (6, 1).

Рим минус Италия — это $(2 - 2, 5 - 1) = (0, 4)$, шаг, который означает «столица». Прибавим этот шаг к Франции: $(6 + 0, 1 + 4) = (6, 5)$, и если Париж находится в $(6, 5)$, сумма попадает точно в него. Настоящие системы используют сотни чисел на слово, но идея та же.

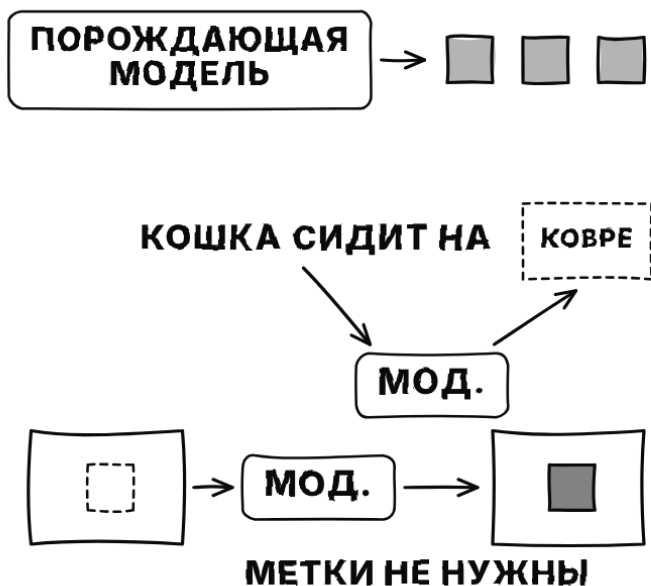
РИМ – ИТАЛИЯ + ФРАНЦИЯ = ПАРИЖ



Рим – Италия + Франция = Париж

Четвёртый вопрос — о причинах. Имея данные о ценах на жильё, загрязнении, преступности, местоположении, образовании и зарплатах, можно ли по одним только числам выяснить, что на что влияет? Такими вопросами занимается изучение причинности и вероятностных графических моделей.

Книга выделяет одно недавнее и увлекательное направление — глубокие порождающие модели. Они выучивают, как распределены данные, какие примеры часты, а какие редки, иногда явно, а иногда лишь неявно. Обученная такая модель умеет две вещи: сказать, насколько вероятен данный пример, или создать совершенно новые примеры, которые выглядят так, будто взяты из тех же данных. Первыми прорывами были вариационные автокодировщики и порождающие состязательные сети; к более новым относятся нормализующие потоки и диффузионные модели. Приложения, которые рисуют картинку по набранной вами фразе, построены на диффузионных моделях.



Наконец, книга описывает самообучение — способ бесплатно получить метки из неразмеченных данных. В тексте вы случайным образом прячете некоторые слова и обучаете модель заполнять пропуски по окружающим словам. Миллиарды предложений становятся обучающими примерами без какой-либо разметки людьми, потому что спрятанное слово и есть метка. На картинках модель может учиться определять, как два вырезанных куса одной фотографии располагались друг относительно друга, дорисовывать закрытую часть по остальному или говорить, получены ли две изменённые картинки из одного оригинала. Представления, которые такая модель выучивает по пути, затем используются повторно: модель дообучают, немного подстраивают под задачу, которая нас действительно интересует. Языковую модель внутри чат-бота сначала обучали именно так — угадывая пропущенные и следующие слова в огромных объёмах текста.

Опыт

Возьмите абзац из газеты и закройте каждое седьмое слово полоской бумаги. Прочитайте его и попробуйте угадать каждое спрятанное слово, а потом поднимите полоски.

Большинство вы угадаете, потому что слова вокруг каждого пропуска многое о нём говорят. Делая ровно это миллионы раз, самообучающаяся модель выучивает язык без единой метки.

Запомним

При обучении с учителем есть начальник, который даёт каждый ответ; при обучении без учителя есть только куча данных, и структуру надо найти самому. Кластеризация находит группы, оценка подпространства находит несколько чисел, описывающих каждый пример, а представления слов могут превращать отношения в арифметику, как в Рим – Италия + Франция = Париж. Глубокие порождающие модели выучивают, как распределены данные, и могут оценивать примеры или создавать новые. Самообучение само делает метки из данных, например скрытые слова, а выученные им представления затем дообучают для других задач.

Вопросы

1. Что, если судить по картинке с двумя начальниками, делает задачу задачей без учителя?
 - a. Начальник проверяет каждый ответ
 - b. Вы получаете только кучу данных, без меток, говорящих, что делать
 - c. Вы повторяете то, что начальник делает в каждой ситуации
 - d. Данные приходят с фиксированным числом признаков

2. Музыкальное приложение объединяет слушателей со схожим вкусом, хотя никаких групп ему не называли. Что это за задача?
- a. Оценка подпространства
 - b. Самообучение
 - c. Кластеризация
 - d. Порождающая модель
3. Что умеет обученная глубокая порождающая модель?
- a. Сказать, насколько вероятен пример, или создать новые примеры, похожие на данные
 - b. Только раскладывать примеры по размеченным классам
 - c. Только находить причины данных
 - d. Только заполнять спрятанные слова
4. Откуда берётся метка при самообучении на тексте?
- a. От людей, читающих каждое предложение
 - b. От начальника, который даёт каждый ответ
 - c. Из отдельного размеченного набора данных
 - d. От самого спрятанного слова, взятого из текста

Ответы

1. **b)** Противоположный начальник сваливает на стол огромную кучу данных и велит заняться с ней наукой о данных; такие задачи без меток — обучение без учителя.
2. **c)** Найти несколько типичных групп, обобщающих данные, например слушателей со схожим вкусом, — это кластеризация.

3. а) Обученная такая модель может сказать, насколько вероятен данный пример, или создать совершенно новые примеры, которые выглядят так, будто взяты из тех же данных.

4. d) Вы прячете некоторые слова и обучаете модель заполнять пропуски; спрятанное слово и есть метка, так что размечать ничего не нужно.

Разобранные примеры

Арифметика слов на игрушечной карте На придуманной карте из двух чисел Рим находится в (2, 5), Италия — в (2, 1), Франция — в (6, 1), а Париж — в (6, 5). Вычислите Рим – Италия + Франция и проверьте, попадёт ли результат в Париж.

Вычитаем по числам: Рим – Италия = $(2 - 2, 5 - 1) = (0, 4)$ — шаг, означающий «столица».

Прибавляем этот шаг к Франции: $(6 + 0, 1 + 4) = (6, 5)$.

Париж находится в (6, 5), так что сумма попадает точно в Париж.

Ответ: (6, 5), то есть Париж

Задачи

1. На такой же игрушечной карте Германия находится в (4, 1), Берлин — в (4, 5), Испания — в (8, 1), а Мадрид — в (8, 5). Вычислите Берлин – Германия + Испания.

Подсказка: Сначала найдите шаг от Германии к Берлину.

Ответ: Берлин – Германия = $(0, 4)$; прибавив его к Испании, получаем (8, 5), то есть Мадрид.

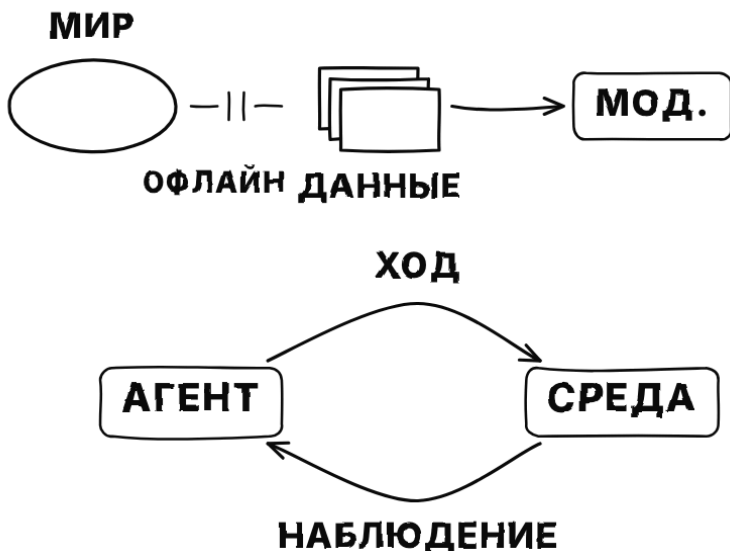
2. Своими словами объясните, почему самообучение может использовать миллиарды предложений, а обучение с учителем обычно нет.

Подсказка: Кто должен поставлять метки в каждом случае?

Ответ: Обучению с учителем нужна метка для каждого примера, обычно её ставят люди, а это долго и дорого. Самообучение делает метки из самих данных, например пряча слова и прося модель их угадать, поэтому можно использовать сколько угодно текста без разметки.

Когда мир отвечает

Подумайте, откуда до сих пор брались все наши данные. Кто-то однажды собрал большую кучу примеров, а затем на ней запустили обучающуюся машину, которая больше никогда не касалась мира. Книга замечает, что обучение с учителем и без учителя почти не спрашивают, откуда берутся данные и что происходит после того, как модель дала ответ. Поскольку всё обучение происходит после того, как программу отрезали от её окружения, это называется офлайн-обучением. Рисунок в книге изображает это просто: среда предоставляет данные, данные собирают, и модель учится на этой куче.



Офлайн-обучение и действие

В этом есть настоящая прелесть. Можно изучать распознавание закономерностей само по себе, не беспокоясь о мире, который движется и реагирует. Но это ограничивает. Всякий, кто вырос на рассказах Айзека Азимова о роботах, представляет себе разумные машины, которые не только предсказывают, но и действуют в мире. Поэтому мы хотим думать об агентах, а не только о моделях, которые предсказывают, и о выборе действий, а не только о предсказаниях. Разница в том, что действия меняют мир. Действия агента меняют то, что он увидит дальше, поэтому он не может один раз научиться на неизменной куче и на этом закончить. Робот-пылесос — повседневный пример: от того, куда он едет, зависит, какую часть комнаты он увидит дальше, а значит, что он сможет узнать о комнате.

Как только мир откликается, возникают новые вопросы, и книга перечисляет несколько. Помнит ли среда, что мы делали раньше? Хочет ли она нам помочь, как человек, который чётко говорит в распознаватель речи? Хочет ли она нас обыграть, как спамеры, которые всё время переписывают свои письма, чтобы проскользнуть мимо спам-фильтров? Ваш спам-фильтр действительно играет в эту игру: каждый раз, когда он выучивает уловку, отправители меняют уловку. И меняется ли мир со временем, так что будущее не похоже на прошлое — само по себе или из-за тех самых инструментов, которые мы внедряем?



Обучили на одном, проверили на другом

Все эти вопросы ведут к одной проблеме — сдвигу распределения: данные, на которых модель проверяют или

используют, отличаются от данных, на которых её обучали. Авторы приводят пример, знакомый многим студентам. Домашние задания составляли ассистенты, поэтому вы тренировались на их вопросах. А экзамен написал лектор, и вопросы выглядели иначе. Вы учились на одном распределении, а проверяли вас на другом.

Опыт

Попросите одного друга написать десять арифметических задач в своём стиле, а другого — ещё десять в другом стиле, например текстовые задачи вместо голых примеров. Тренируйтесь только на первом наборе, пока не станете решать быстро, а затем засекайте время на втором. Обычно на непривычном стиле вы будете решать медленнее, хотя арифметика та же. Это замедление и есть сдвиг распределения в миниатюре.

Запомним

Офлайн-обучение один раз собирает кучу данных и учится на ней, больше никогда не действуя в мире. Это удобно, но действия агента меняют мир и тем самым меняют то, что он видит дальше. Среда может помнить, помогать, сопротивляться или меняться со временем. Когда данные при использовании отличаются от обучающих, возникает сдвиг распределения — как домашние задания от ассистентов и экзамен от лектора.

Вопросы

1. Почему офлайн-обучения недостаточно для агента вроде робота-пылесоса?
 - a. Потому что действия агента меняют то, что он увидит дальше
 - b. Потому что роботы не умеют хранить данные
 - c. Потому что офлайн-обучение работает только с текстом
 - d. Потому что агент никогда не делает предсказаний
2. Спамеры всё время переписывают письма, чтобы проскользнуть мимо фильтров. Что это за среда?
 - a. Та, что хочет нам помочь
 - b. Та, что хочет нас обыграть
 - c. Та, что никогда не меняется
 - d. Та, что всё забывает
3. Вы тренировались на домашних заданиях от ассистентов, а затем сдавали экзамен, написанный лектором. Пример чего это?
 - a. Переобучения на тестовой выборке
 - b. Офлайн-обучения
 - c. Сдвига распределения
 - d. Кластеризации

Ответы

1. а) Действия меняют мир: от того, куда едет робот, зависит, какую часть комнаты он увидит дальше, поэтому он не может один раз научиться на неизменной куче.
2. б) Пример книги о среде, которая хочет нас обыграть, — спамеры, переписывающие письма, чтобы обойти спам-фильтры.
3. с) Вы учились на одном распределении — вопросах ассистентов, а проверяли вас на другом — вопросах лектора: это сдвиг распределения.

Задачи

1. Модель, предсказывающую, что люди покупают в интернете, обучили на данных до большой праздничной распродажи. Объясните, как на неё может повлиять сдвиг распределения во время распродажи.

Подсказка: Похожи ли данные во время распродажи на те, на которых она училась?

Ответ: Во время распродажи люди покупают другие вещи, в других количествах и по другим ценам, поэтому данные при использовании отличаются от обучающих. Предсказания модели могут быть плохими, потому что она выучила закономерности, которые больше не выполняются.

2. Для каждой среды скажите, помогает ли она, сопротивляется или просто меняется со временем: человек, чётко говорящий голосовому помощнику; мошенники, меняющие уловки, чтобы обмануть банковский детектор; модные тенденции, меняющиеся из года в год.

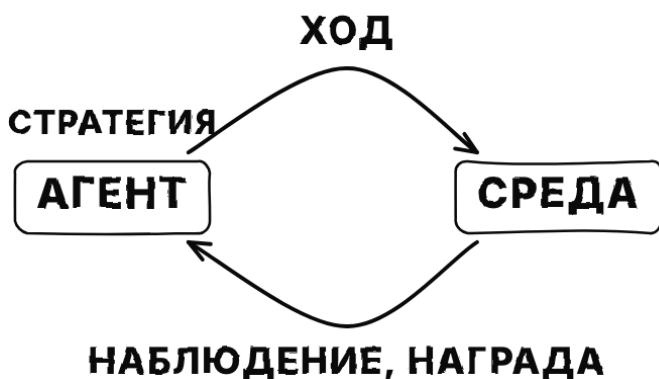
Подсказка: Спросите себя, чего хочет среда, если вообще чего-то хочет.

Ответ: Человек, говорящий чётко, помогает системе. Мошенники сопротивляются, подстраиваясь, чтобы обмануть детектор. Мода

просто меняется со временем, так что будущие данные не похожи на прошлые.

Учиться действуя: обучение с подкреплением

Когда вы учите собаку садиться, вы ничего не объясняете. Вы ждёте, пока она случайно сядет, даёте ей лакомство и повторяете. За много попыток собака понимает, какие из её действий приносят лакомства. Ей никогда не говорят правильного ответа; она лишь узнаёт потом, хорошо ли всё прошло.



СТРАТЕГИЯ: НАБЛЮДЕНИЕ → ДЕЙСТВИЕ

Цикл обучения с подкреплением

Когда машинное обучение используют, чтобы построить агента, действующего в мире, говорит книга, скорее всего вы придёте

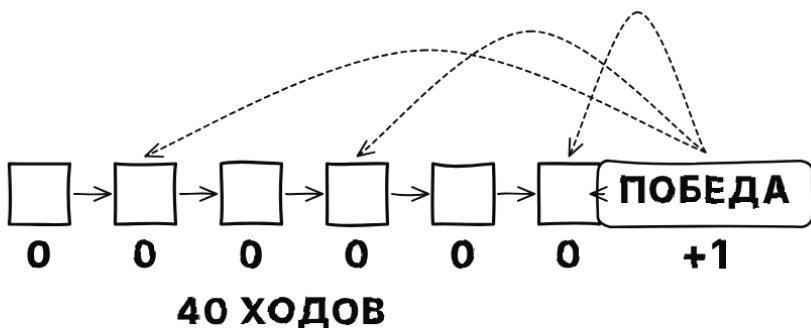
именно к такому обучению, называемому обучением с подкреплением. Среди его применений — роботы, диалоговые системы и игроки в видеоигры. Соединившись с глубоким обучением, оно стало глубоким обучением с подкреплением и дважды попало в заголовки. Программа под названием глубокая Q-сеть научилась обыгрывать людей в видеоигры Atari, видя только картинку на экране. А AlphaGo обыграла чемпиона мира по настольной игре го.

Постановка простая и очень общая. Агент и среда ходят по очереди, шаг за шагом. На каждом шаге агент получает наблюдение — то, что он может видеть в мире. Он выбирает действие, которое передаётся обратно в мир через некий механизм, иногда называемый исполнительным устройством, например через мотор робота. Затем он получает награду — число, говорящее, насколько хорош был этот ход, и цикл начинается снова со следующего наблюдения. Рисунок в книге изображает это кругом: агент отправляет действия в среду, а среда возвращает наблюдения и награды. Тем, что делает агент, управляет его стратегия — просто правило, функция, которая отображает то, что он наблюдает, в то, что он делает. Цель обучения с подкреплением — найти хорошие стратегии. Робот-пылесос, изучающий планировку вашей квартиры, находится в этом цикле: он ощущает комнату, выбирает, куда ехать, и получает награду за убранный пол.

Эта схема настолько общая, что поглощает даже обучение с учителем. Возьмите задачу классификации с десятью классами. Дайте агенту десять действий, по одному на класс, и пусть среда платит награду, взятую прямо из функции потерь исходной задачи, так что награда оценивает каждый выбор так же, как

потери оценивали каждое предсказание. Тогда выбирать действия ради награды — то же самое, что классифицировать.

ЗАСЛУГА?



Награда только в конце

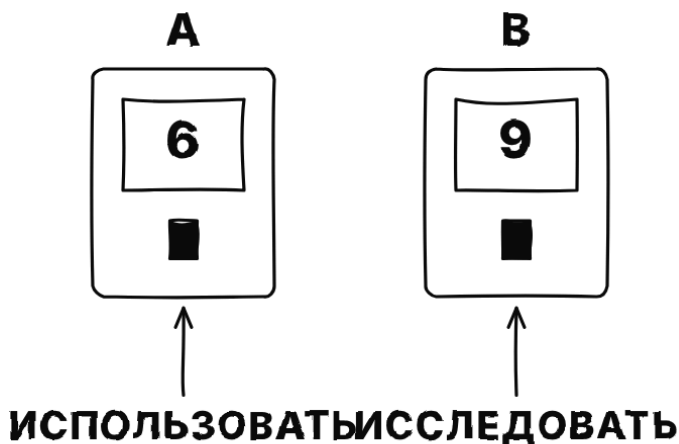
И оно справляется с тем, что обучению с учителем не под силу. В обучении с учителем каждый обучающий вход приходит со своей правильной меткой. В обучении с подкреплением никто не говорит агенту лучшего действия для каждого наблюдения; он просто получает какую-то награду. Хуже того, мир может не сообщать, какие действия принесли эту награду.

Книга показывает эту трудность на шахматах. Единственная настоящая награда приходит в самом конце, скажем +1 за победу и -1 за поражение. Партия может длиться 40 ходов, и +1

приходит только после последнего из них. Какой из этих 40 ходов выиграл партию, а какой чуть её не проиграл? Выяснение того, каким действиям принадлежит заслуга или вина, называется задачей распределения заслуг. Человеческий пример книги — сотрудник, которого повысили 11 октября: повышение отражает много хороших решений за прошедший год, и чтобы снова получить повышение, нужно понять, какие из них были важны.

Вторая трудность — частичная наблюдаемость: то, что агент видит сейчас, может не говорить ему, где он находится.

Представьте, что ваш робот-уборщик застрял в одной из нескольких одинаковых кладовок вашего дома. Изнутри они все выглядят одинаково; чтобы узнать, в какой он, ему нужно помнить, что он видел до того, как туда попал.



Использовать или исследовать

Третья — выбор между использованием и поиском. В любой момент агент может знать одну хорошую стратегию, тогда как существуют лучшие, которых он никогда не пробовал. Он может использовать, держась лучшей известной стратегии, или исследовать, пробуя другие и жертвуя частью награды сейчас ради знаний.

Опыт

Представьте два кофейных автомата на работе, которые вы ещё не пробовали. В понедельник вы пробуете автомат А, и кофе нормальный, скажем на 6 из 10. Если вы только используете, вы навсегда пьёте из А и так и не узнаете, что автомат В делает кофе на 9. Если вы потратите вторник на исследование В, вы рискуете в этот день плохой чашкой, но каждый следующий день может быть на 9. Этот размен между верными 6 сейчас и возможными 9 потом и есть выбор между исследованием и использованием.

Всё это вместе — действия, влияющие на последующие наблюдения, награды, видимые только для выполненных действий, мир, видимый полностью или лишь частично, — слишком много, чтобы справиться со всем сразу. И не в каждой практической задаче всё это есть. Поэтому исследователи изучают частные случаи, и книга называет три. Когда агент видит всё состояние мира, задача называется марковским процессом принятия решений. Когда состояние не зависит от того, что агент делал раньше, это задача контекстного бандита. А когда состояния нет вовсе, а есть лишь набор действий, награды за которые поначалу неизвестны, это классическая задача многорукого бандита, названная в честь ряда игровых автоматов, у каждого из которых свой неизвестный выигрыш; два наших кофейных автомата — крошечный её пример. Новостное приложение, которое показывает разным читателям два заголовка, чтобы увидеть, какой соберёт больше кликов, решает задачу бандита.

Запомним

В обучении с подкреплением агент действует шаг за шагом: наблюдает, выбирает действие и получает награду. Его стратегия отображает наблюдения в действия, а цель — хорошая стратегия. В отличие от обучения с учителем, ему никогда не говорят правильного действия, а награда может прийти поздно, поэтому он должен решать задачу распределения заслуг, как в шахматах с +1 или -1 только в конце. Он может видеть лишь часть мира и должен уравнивать использование того, что работает, с исследованием того, что может сработать лучше. Частные случаи — марковские процессы принятия решений, контекстные бандиты и многоаружные бандиты.

Вопросы

1. Что делает стратегия агента обучения с подкреплением?
 - a. Отображает то, что агент наблюдает, в то, что он делает
 - b. Решает, насколько велика награда
 - c. Размечает каждый обучающий пример
 - d. Хранит каждое наблюдение навсегда
2. Шахматная программа выигрывает партию в 40 ходов и получает +1 в конце. Какую задачу ей нужно решить, чтобы на этом учиться?
 - a. Частичной наблюдаемости

- b. Сдвига распределения
 - c. Распределения заслуг: какие из ходов заслуживают заслуги
 - d. Кластеризации ходов по группам
3. Почему робот-уборщик, застрявший в одной из нескольких одинаковых кладовок, сталкивается с частичной наблюдаемостью?
- a. Его награда слишком мала
 - b. То, что он видит сейчас, не говорит ему, в какой он кладовке
 - c. У него слишком много действий на выбор
 - d. Ему на каждом шаге говорят правильное действие
4. В задаче нет состояния вовсе, а есть лишь набор действий с поначалу неизвестными наградами. Как она называется?
- a. Марковский процесс принятия решений
 - b. Задача контекстного бандита
 - c. Обучение с учителем
 - d. Задача многорукого бандита

Ответы

1. a) Тем, что делает агент, управляет его стратегия — функция, отображающая то, что он наблюдает, в то, что он делает; цель — найти хорошие стратегии.
2. c) Награда приходит только после последнего хода, поэтому обучающийся должен понять, каким действиям принадлежит заслуга или вина.

3. b) Изнутри все кладовки выглядят одинаково; чтобы узнать, в какой он, ему нужно помнить, что он видел до того, как туда попал.

4. d) Когда состояния нет, а есть лишь действия с поначалу неизвестными наградами, это классическая задача многорукого бандита.

Разобранные примеры

Стоит ли исследовать? За 10 рабочих дней вы выпиваете по одной чашке кофе в день. Автомат А, опробованный в первый день, получает 6 баллов. Автомат В не опробован. Сравните общую сумму баллов, если всегда пить из А и если во второй день попробовать В, а затем пить из лучшего, в случае когда В получает 9 баллов и когда он получает 2.

Всегда использовать А: $10 \times 6 = 60$.

Исследовать В во второй день, если В получает 9: $6 + 9 + 8 \times 9 = 87$.

Исследовать В во второй день, если В получает 2: $6 + 2 + 8 \times 6 = 56$.

Исследование стоит не больше 4 баллов, если В плох, но даёт 27, если В хорош: отказ от небольшой краткосрочной награды может купить ценное знание.

Ответ: Всегда А: 60; с исследованием: 87, если В хорош, 56, если В плох

Задачи

1. Назовите наблюдение, действие и награду для робота-пылесоса, убирающего квартиру.

Подсказка: Что он ощущает, что делает и за что получает награду?

Ответ: Наблюдение — то, что его датчики сообщают о комнате вокруг; действие — куда он поедет дальше; награда — площадь убранного пола.

2. Своими словами объясните, как задачу классификации с десятью классами можно превратить в обучение с подкреплением.

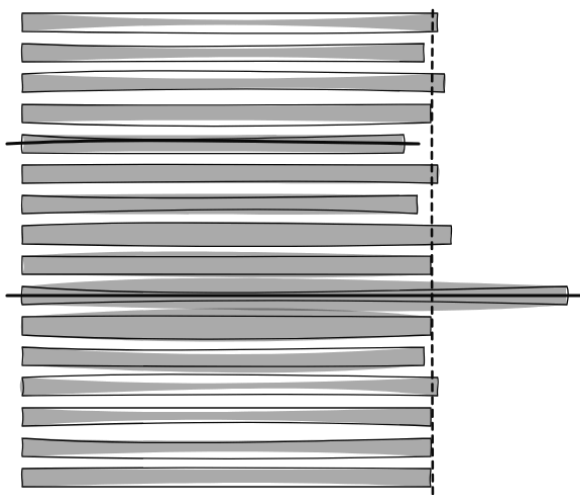
Подсказка: Сделайте каждый класс действием.

Ответ: Дайте агенту десять действий, по одному на класс, и пусть среда платит награду, взятую из функции потерь исходной задачи. Тогда награда оценивает каждый выбор класса так же, как потери оценивали каждое предсказание, и научиться хорошо действовать — значит научиться классифицировать.

Старые корни: от церковных дверей до нейронов

Представьте воскресенье в немецком городке XVI века. Служба заканчивается, и когда люди выходят из церкви, шестнадцать взрослых мужчин просят встать в ряд, пятка к носку. Общую длину их шестнадцати ступней измеряют и делят на шестнадцать. Результат объявляют одним «футом» — единицей длины этого города.

30,1, УСЕЧЁННОЕ



Шестнадцать ступней, один фут

Книга берёт эту сцену из книги по геометрии Якоба Кёбеля, жившего с 1460 по 1533 год, и на её рисунке мужчины стоят в ряд. Смысл в том, что обучение на данных гораздо старше компьютеров. Люди всегда хотели читать данные и предсказывать, что будет дальше, и это желание лежит в основе значительной части науки и математики. Даже этот простой рецепт позже улучшили, чтобы справиться со странными ступнями: мужчин с самой короткой и самой длинной ступнёй отпускали, а усредняли только остальных четырнадцать. Это одно из самых ранних известных усечённых средних — среднее, которое не учитывает крайние значения.

Посмотрим на придуманных числах, почему это помогает. Скажем, у пятнадцати мужчин ступни около 30 сантиметров, в сумме 450, а у шестнадцатого, великана, ступня 40. Обычное среднее $(450 + 40) / 16 \approx 30,6$ сантиметра — его тянет вверх один человек. Отбросьте самую длинную ступню, 40, и самую короткую, скажем 28, и усредните оставшиеся четырнадцать: $(450 - 28) / 14 \approx 30,1$. Одна необычная ступня больше не сдвигает городскую единицу. Оценки в магазине приложений связаны с той же заботой: несколько гневных или поддельных отзывов могут утянуть среднее вниз, поэтому некоторые сайты отбрасывают их или придают им меньший вес.

Книга называет и других предков. Якоб Бернулли, 1655–1705, и Карл Фридрих Гаусс, 1777–1855, дали свои имена распределениям вероятностей, которые до сих пор используются повсюду. Гаусс также изобрёл метод наименьших квадратов — выбор прямой, при которой сумма квадратов промахов наименьшая; его применяют до сих пор во всём, от страхования до медицинской диагностики. Такие инструменты укрепили

экспериментальную науку. Закон Ома для резистора, $V = R \times I$, напряжение равно сопротивлению, умноженному на ток, — идеальная линейная модель: удвойте ток через резистор в 10 Ом с 0,5 А до 1 А, и напряжение удвоится с 5 до 10 вольт.

По-настоящему статистика развернулась, когда данные стало можно собирать в больших количествах. Рональд Фишер, 1890–1962, построил значительную часть её теории и применил её к генетике; методы и понятия, названные его именем, до сих пор лежат в основе этой области. Его набор данных об ирисах 1936 года, измерения трёх видов цветков ириса, до сих пор используют, чтобы показать работу алгоритмов обучения. Книга добавляет предупреждение: Фишер продвигал и евгенику, а это напоминает, что науку о данных столь же долго используют для нравственно сомнительных целей, как и для добрых.

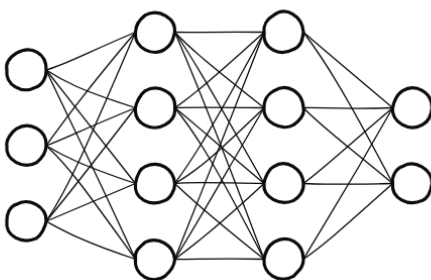
Другие корни — в теории информации Клода Шеннона и в теории вычислений Алана Тьюринга. Тьюринг спросил: «Может ли машина мыслить?» — и предложил то, что мы теперь называем тестом Тьюринга: машина считается разумной, если человеку, переписывающемуся с ней, трудно отличить её ответы от ответов человека. Многим людям, общающимся с современным чат-ботом, было бы трудно вынести вердикт по этому тесту, и отчасти поэтому о нём всё ещё спорят.

Ещё одни корни — в изучении мозга и психики. Люди явно разумны, поэтому учёные спрашивали, можно ли эту способность объяснить и скопировать. Дональд Хебб, 1904–1985, предположил в своей книге «Организация поведения», что нейроны учатся через положительное подкрепление: связи, которые используются вместе, становятся сильнее. Это правило

Хебба вдохновило перцептрон Розенблатта, раннюю обучающуюся машину, и лежит в основе идеи, на которой строятся методы обучения сегодняшнего глубокого обучения: подкреплять то, что работает, ослаблять то, что не работает, и так приходить к хорошим значениям параметров сети.



ВМЕСТЕ



ЛИНЕЙНЫЙ ЛИНЕЙНЫЙ
НЕЛИНЕЙНЫЙ

**ЦЕПНОЕ
ПРАВИЛО**



Слои и обратное распространение

Отсюда и название — нейронная сеть. Больше века, начиная с моделей Александра Бэна 1873 года и Джеймса Шеррингтона 1890 года, как их называет книга, исследователи пытаются строить схемы, похожие на сети нейронов. Со временем связь с биологией ослабла, но название осталось. В большинстве современных сетей есть два принципа. Первый — слои, в которых чередуются простые линейные шаги и нелинейные. Второй —

цепное правило математического анализа, известное в этой области как обратное распространение ошибки, которое настраивает каждый параметр всей сети сразу.

Опыт

Возьмите линейку и попросите друга бросить её, а вы ловите как можно быстрее, десять раз. Сначала вы ловите медленно и неровно; после нескольких попыток — увереннее. Никто не говорил вашему мозгу, какие нервные связи укреплять; те, что привели к удачному захвату, просто использовались снова. В этом и дух правила Хебба, вдохновившего первые обучающиеся машины.

После быстрого раннего прогресса исследования нейронных сетей застопорились примерно с 1995 по 2005 год по двум причинам. Обучение сети требует очень много вычислений, а в конце прошлого века памяти было достаточно, но вычислительной мощности не хватало. И наборы данных были малы. Данные об ирисах Фишера 1936 года, 150 цветков, всё ещё были обычной проверкой, а собрание MNIST из 60 000 рукописных цифр считалось огромным. При малом объёме данных и вычислений другие методы — ядерные методы, деревья решений и графические модели — во многих применениях просто работали лучше. Им не требовались недели обучения, и они давали предсказуемые результаты, подкреплённые сильной теорией.

Запомним

Обучению на данных много веков: шестнадцать ступней Кёбеля, усреднённые, а позже очищенные от крайних значений, — ранняя оценка. Бернулли, Гаусс, Фишер, Шеннон и Тьюринг заложили основы, а метод наименьших квадратов Гаусса и линейные модели вроде закона Ома используются до сих пор. Идея Хебба о том, что нейроны учатся через подкрепление, привела к перцептрону и дала нейронным сетям их название. Сегодняшние сети опираются на чередование линейных и нелинейных слоёв и на обратное распространение. Примерно с 1995 по 2005 год нейронные сети застопорились, потому что вычислений не хватало, а наборы данных были малы, и более простые и быстрые методы работали лучше.

Вопросы

1. Зачем по рецепту Кёбеля отпускали мужчин с самой короткой и самой длинной ступнёр?
 - a. Чтобы одна необычная ступня не сильно сдвигала городскую единицу
 - b. Потому что на церковных ступенях помещалось только четырнадцать мужчин
 - c. Чтобы было проще делить
 - d. Потому что самые длинные ступни были у приезжих

2. Что Тьюринг предложил в качестве проверки разумности машины?
 - a. Чтобы она обыгрывала человека в шахматы
 - b. Чтобы она училась через подкрепление
 - c. Чтобы человеку, переписывающемуся с ней, было трудно отличить её ответы от человеческих
 - d. Чтобы она точно измеряла ступни
3. Какие два принципа есть в большинстве современных нейронных сетей?
 - a. Усечённое среднее и метод наименьших квадратов
 - b. Чередование линейных и нелинейных слоёв и обратное распространение
 - c. Правило Хебба и тест Тьюринга
 - d. Ядерные методы и деревья решений
4. Почему примерно с 1995 по 2005 год ядерные методы и деревья решений работали лучше нейронных сетей?
 - a. Они были вдохновлены мозгом
 - b. Им нужно было больше данных
 - c. Их изобрели позже
 - d. При малых данных и вычислениях они работали лучше, быстро обучались и имели сильную теорию

Ответы

1. a) Отбрасывание крайних значений перед усреднением, то есть усечённое среднее, не даёт одной странной ступне тянуть единицу вверх или вниз.

2. c) Тест Тьюринга признаёт машину разумной, если человеку, переписывающемуся с ней, трудно отличить её ответы от ответов человека.

3. b) В большинстве сетей есть слои, где чередуются простые линейные и нелинейные шаги, и используется цепное правило, обратное распространение, чтобы настроить все параметры сразу.

4. d) При малом объёме данных и вычислений они просто работали лучше, не требовали недель обучения и давали предсказуемые результаты, подкреплённые сильной теорией.

Разобранные примеры

Обычное и усечённое среднее Ступни пятнадцати мужчин в сумме дают 450 сантиметров; шестнадцатая ступня — 40 сантиметров. Самая короткая ступня — 28 сантиметров. Вычислите обычное среднее всех шестнадцати и усечённое среднее, отбросив 40 и 28.

Обычное среднее: $(450 + 40) \div 16 = 490 \div 16 \approx 30,6$ сантиметра.

Усечение убирает 40 и 28; 28 входит в число пятнадцати, так что оставшиеся четырнадцать в сумме дают $450 - 28 = 422$.

Усечённое среднее: $422 \div 14 \approx 30,1$ сантиметра.

Одна длинная ступня больше не тянет результат вверх.

Ответ: Обычное среднее около 30,6 см; усечённое среднее около 30,1 см

Задачи

1. По закону Ома $V = R \times I$: какое напряжение на резисторе в 20 Ом при токе 0,3 ампера? Что будет с напряжением, если ток утроить?

Подсказка: Умножьте сопротивление на ток; закон линейный.

Ответ: $V = 20 \times 0,3 = 6$ вольт. Если утроить ток до 0,9 ампера, напряжение утроится до 18 вольт, потому что модель линейная.

2. Назовите две причины, по которым исследования нейронных сетей застопорились примерно между 1995 и 2005 годами, и скажите, чем каждая ограничивала сети.

Подсказка: Одна про компьютеры, другая про данные.

Ответ: Не хватало вычислительной мощности, а обучение сети требует очень много вычислений, поэтому сети нельзя было хорошо обучить. И наборы данных были малы, типичными были данные об ирисах и 60 000 цифр MNIST, так что большим сетям было почти не на чем учиться.

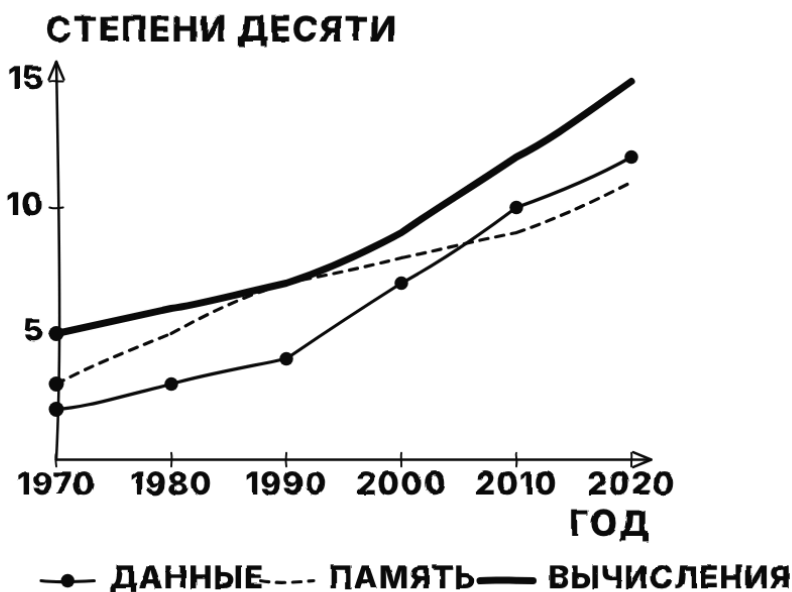
Путь к глубокому обучению

Сравните телефон в вашем кармане с настольным компьютером 1990 года. Ваш телефон хранит в тысячи раз больше фотографий, чем мог вместить тот компьютер, и за секунду выполняет больше вычислений, чем та машина за часы. А теперь подумайте, сколько фотографий, сообщений и кликов весь мир производит каждый день. Книга говорит, что именно эта двойная перемена — больше данных и гораздо больше вычислений — разбудила нейронные сети от долгого сна.

Сошлось несколько вещей. Всемирная паутина и компании, обслуживающие сотни миллионов людей онлайн, дали данные в небывалых количествах. Дешёвые и хорошие датчики распространились повсюду, от камер до микрофонов. Хранение становилось всё дешевле — эту тенденцию книга называет законом Крайдера, — а вычисления тоже, по закону Мура. А одно устройство изменило правила игры: графический процессор, или GPU, созданный для компьютерных игр. GPU выполняет тысячи простых вычислений одновременно, а это как раз то, что нужно для обучения нейронной сети. Внезапно модели, которые казались непомерно дорогими для вычислений, стали достигаемы. Тот же вид чипа, что рисует пейзаж в видеоигре, теперь обучает модели, стоящие за чат-ботами и голосовыми помощниками.

Книга показывает перемену в таблице по десятилетиям, с тремя столбцами: насколько велик был типичный набор данных, сколько памяти было у компьютера и сколько вычислений он

делал в секунду. В 1970 году набор данных означал около сотни примеров — ирисы Фишера; памяти был один килобайт; процессор делал около ста тысяч вычислений в секунду. К 1990 году наборы данных содержали около десяти тысяч примеров, например отсканированный почерк; памяти было десять мегабайт; процессор делал десять миллионов вычислений в секунду. К 2010 году наборы данных содержали около десяти миллиардов примеров из онлайн-рекламы; памяти был один гигабайт; GPU делал триллион вычислений в секунду. А к 2020 году набор данных из социальной сети мог содержать триллион примеров при ста гигабайтах памяти и квадриллионе вычислений в секунду.

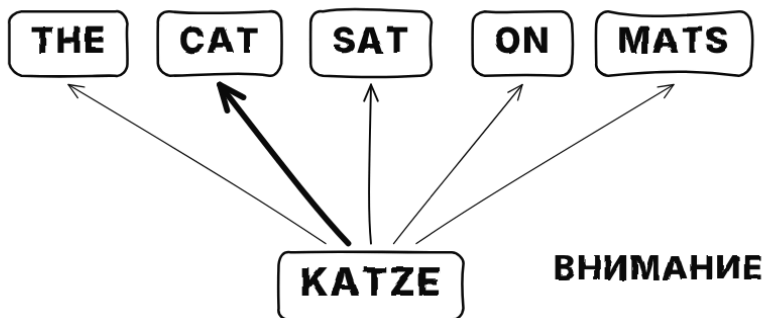


Данные, память и вычисления, 1970–2020

Прочитаем таблицу за все пятьдесят лет. Данные выросли со ста примеров до триллиона, то есть с 10^2 до 10^{12} , в 10^{10} раз — в десять миллиардов. Память выросла с одного килобайта до ста гигабайт, в 10^8 раз — в сто миллионов. Число вычислений в секунду выросло со ста тысяч до квадриллиона, снова в 10^{10} раз. Значит, память отстала от роста данных в сто раз, а вычислительная мощность, говорит книга, росла даже быстрее наборов данных. Книга делает вывод: модели должны расходовать память экономно, но могут позволить себе тратить гораздо больше вычислений на настройку параметров. Это сместило золотую середину машинного обучения от линейных моделей и ядерных методов, подходивших старым условиям, к глубоким нейронным сетям. Поэтому же многие классические инструменты глубокого обучения, такие как многослойные перцептроны, свёрточные сети, долгая краткосрочная память и Q-обучение, по сути были открыты заново в 2010-х после многих лет спячки.

Но авторы подчёркивают, что прогресс — это не только старые идеи на новом железе. Они сравнивают последнее десятилетие с кембрийским взрывом — моментом в истории жизни, когда новые виды животных появились стремительно. Новые идеи множились, и книга перечисляет некоторые из них, признавая, что список лишь слегка касается темы.

ДРОПАУТ: ШУМ ПРИ ОБУЧЕНИИ



ТРАНСФОРМЕР: ТОЛЬКО ВНИМАНИЕ

Внимание как указатель

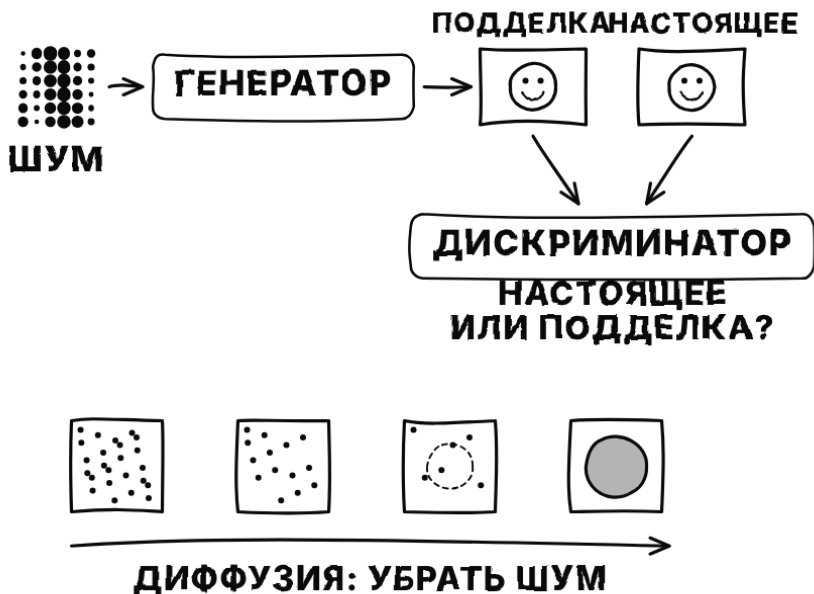
Первая — лучшая борьба с переобучением с помощью методов вроде дропаута. Во время обучения в сеть повсюду вносится случайный шум, например случайным отключением её частей, так что она не может просто запомнить обучающие примеры.

Вторая — внимание. Статистика долго билась над загадкой: как дать системе больше памяти и сложности, не добавляя обучаемых параметров. Старые модели перевода должны были втиснуть всё предложение в одну сводку фиксированного размера, прежде чем начать перевод, и длинные предложения страдали. Внимание позволяет модели вместо этого держать своего рода обучаемый указатель и по ходу дела оглядываться на нужную часть входа. Точность на длинных предложениях резко

выросла, потому что модели больше не нужно было помнить всё, прежде чем начать выдавать перевод.

Третья выросла из второй — трансформер, архитектура, целиком построенная на внимании. Его поразительное свойство — то, как он масштабируется: он продолжает улучшаться с ростом данных, модели и бюджета вычислений. Трансформеры сейчас успешны в языке, зрении, распознавании речи, обучении с подкреплением и не только. Книга упоминает один трансформер, обученный на тексте, изображениях, движениях суставов робота и нажатиях кнопок, который умеет играть в игры Atari, описывать картинки, вести беседу и управлять рукой робота.

Четвёртая — языковые модели, которые выучивают вероятность текста и поэтому могут предсказывать текст по другому тексту. Масштабирование данных, модели и вычислений открыло всё больше и больше способностей. ChatGPT, который называет книга, — это языковая модель, настроенная следовать намерениям людей; с ней можно поговорить, чтобы исправить ошибку в коде или набросать рассказ.



Генератор против дискриминатора

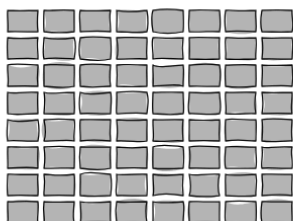
Пятая — многоступенчатые конструкции, такие как сети с памятью и нейронный программист-интерпретатор. Они позволяют сети снова и снова, шаг за шагом, менять собственное внутреннее состояние, подобно тому как процессор обновляет память при вычислении, и так следовать цепочке рассуждений.

Шестая — новый способ порождать данные, порождающие состязательные сети. Старым методам приходилось записывать точную формулу вероятности, а затем искать способ извлекать из неё образцы, и это их ограничивало. Порождающая состязательная сеть заменяет алгоритм выборки любой настраиваемой программой, генератором, и сталкивает её со второй сетью, дискриминатором, которая пытается отличить

поддельные примеры от настоящих. Генератор настраивают, пока дискриминатор больше не может заметить разницы. Среди результатов — видео скачущих зебр и убедительные лица людей, которых никогда не существовало; даже любители могут превратить грубый набросок в похожую на фотографию картинку.

Седьмая — диффузионные модели. Диффузия медленно добавляет к картинке случайный шум, пока не останется только шум; диффузионная модель учится прогонять этот процесс в обратную сторону, шаг за шагом убирая шум, поэтому может начать с чистого шума и закончить новой картинкой. Эти модели начали вытеснять состязательные сети, например в DALL-E 2 и Imagen, которые создают изображения по письменному описанию. Приложения, которые превращают фразу вроде «кошка в скафандре» в картинку, построены на таких моделях.

Восьмая — обучение сразу на многих GPU. Часто одного GPU не хватает на все данные. Рабочая лошадка обучения, стохастический градиентный спуск, обрабатывает за раз небольшие порции примеров, называемые мини-пакетами, но малые пакеты не дают использовать всю мощь GPU. Поэтому работу распределяют: при 1024 GPU, каждый с мини-пакетом из 32 изображений, один шаг обрабатывает $1024 \times 32 = 32\,768$ изображений, около тридцати двух тысяч. Последующие работы довели это до шестидесяти четырёх тысяч изображений за шаг и сократили обучение известной модели изображений, ResNet-50, на собрании ImageNet до менее чем семи минут, тогда как когда-то оно занимало дни.



1024 GPU

$$1,024 \times 32 = 32,768$$

ИЗОБРАЖЕНИЙ ЗА ШАГ

ДНИ



■ МИНУТЫ

THEANO → TENSORFLOW → PYTORCH, JAX

Много GPU, один большой пакет

Девятая — обучение с подкреплением, которое тоже выиграло от параллельных вычислений. Компьютеры достигли сверхчеловеческой игры в го, игры Atari и Starcraft и хорошо проявили себя в физических симуляциях — везде, где есть симулятор среды. Причина проста: обучение с подкреплением работает лучше всего при обилии примеров состояния, действия и награды, а симулятор может произвести их сколько угодно.

Десятая — программное обеспечение. Идеи распространили фреймворки глубокого обучения. Первым поколением открытых инструментов были Caffe, Torch и Theano, и с ними написано много знаковых статей. Их сменили TensorFlow, которым часто пользуются через его более простую надстройку Keras, а также

CNTK, Caffe 2 и Apache MXNet. Третье поколение императивное: модель описывают на обычном Python, шаг за шагом, так же, как записали бы любое вычисление с библиотекой NumPy; начал это Chainer, а за ним последовали PyTorch, интерфейс Gluon от MXNet и JAX.

Опыт

Возьмите лист бумаги и выпишите рядом коэффициенты роста за пятьдесят лет из таблицы: данные — в десять миллиардов раз; память — в сто миллионов раз; вычисления — в десять миллиардов раз. Теперь представьте, что ваш дом вырос так же: комната стала в сто миллионов раз больше и заполнилась в десять миллиардов раз большим числом коробок. Держать в поле зрения каждую коробку вы бы не смогли; пришлось бы с умом выбирать, что держать при себе, а за остальным бегать на своих невероятно быстрых ногах. Это и есть ситуация, которую описывает книга, и глубокие нейронные сети подходят к ней лучше старых методов.

Запомним

Нейронные сети вернулись, потому что данных стало взрывообразно много — благодаря интернету, датчикам и дешёвому хранению, — а вычисления подешевели, прежде всего благодаря GPU, созданным для игр. За пятьдесят лет данные и вычисления выросли примерно в десять миллиардов раз, а память лишь примерно в сто миллионов, поэтому модели теперь меняют память на вычисления, что благоприятствует глубоким сетям. Важны были и новые идеи: дропаут против переобучения, внимание и трансформер, языковые модели, состязательные сети и диффузионные модели для порождения данных. Обучение, распределённое по многим GPU, с шагами в десятки тысяч изображений, сократило время обучения с дней до минут, а симуляторы питали обучение с подкреплением. Открытые фреймворки, от Theano и Caffe до TensorFlow и PyTorch, донесли эти идеи до всех.

Вопросы

1. Почему графический процессор, созданный для игр, так хорошо подошёл нейронным сетям?
 - a. У него больше памяти, чем у любого другого чипа
 - b. Он выполняет тысячи простых вычислений одновременно, а это и нужно для обучения
 - c. Он создан для хранения больших наборов данных

- d. Он быстрее выполняет код на Python
2. Память росла гораздо медленнее данных, а вычисления — быстро. Что это означало для моделей?
- a. Им приходилось хранить каждый пример
 - b. Им пришлось отказаться от GPU
 - c. Им нужно было экономно расходовать память, но можно было тратить гораздо больше вычислений на настройку параметров
 - d. Им нужно было стать меньше и проще, как линейные модели
3. Какую проблему моделей перевода решило внимание?
- a. Им приходилось втискивать всё предложение в одну фиксированную сводку перед переводом
 - b. Они не могли работать на GPU
 - c. Им были нужны размеченные данные
 - d. Они переобучались на обучающей выборке
4. Когда в порождающей состязательной сети генератор достаточно хорош?
- a. Когда он точно копирует обучающие примеры
 - b. Когда дискриминатор замечает каждую подделку
 - c. Когда он записывает формулу вероятности
 - d. Когда дискриминатор больше не может отличить его подделки от настоящих примеров

Ответы

1. **b)** GPU выполняет тысячи простых вычислений одновременно, а это как раз то, что нужно для обучения нейронной сети.
2. **c)** Модели должны расходовать память экономно, но могут позволить себе гораздо больше вычислений на настройку параметров, и это сместило золотую середину к глубоким сетям.
3. **a)** Старые модели втискивали всё предложение в одну сводку фиксированного размера; внимание держит обучаемый указатель и по ходу дела оглядывается на нужную часть.
4. **d)** Генератор настраивают против дискриминатора, который пытается отличить подделку от настоящего, пока дискриминатор больше не может заметить разницы.

Разобранные примеры

Общий мини-пакет Обучение распределено по 1024 GPU, каждый обрабатывает мини-пакет из 32 изображений. Сколько изображений обрабатывает один шаг обучения?

Каждый GPU обрабатывает 32 изображения за шаг.

Все 1024 GPU работают одновременно.

$1024 \times 32 = 32\,768$ изображений за шаг, около тридцати двух тысяч.

Ответ: 32 768 изображений, около 32 000

Задачи

1. По таблице: во сколько раз выросла память с одного килобайта в 1970 году до ста гигабайт в 2020-м и во сколько раз выросли данные со ста примеров до триллиона?

Подсказка: Гигабайт — это миллион килобайт; считайте степени десяти.

Ответ: Память: 100 ГБ — это 100 000 000 КБ, рост в 10^8 раз.

Данные: с 10^2 до 10^{12} , рост в 10^{10} раз. Данные выросли в сто раз сильнее, чем память.

2. Если 256 GPU обрабатывают каждый мини-пакет из 64 изображений, сколько изображений обрабатывается за один шаг?

Подсказка: Умножьте число GPU на размер мини-пакета.

Ответ: $256 \times 64 = 16\,384$ изображения за шаг.

Скрытые успехи и видимые

Напишите письмо, надпишите на конверте адрес от руки и отправьте его. Где-то в сортировочном центре машина читает этот адрес и отправляет письмо дальше, и делает это с 1990-х годов. Вы этого не видите, но это машинное обучение, и книга говорит, что это типично: технология десятилетиями тихо работает.

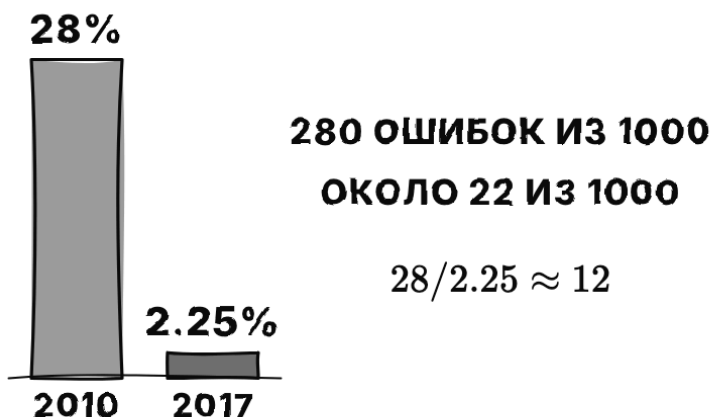
Но сначала авторы замечают, насколько легче стала сама работа. Исследователи систем теперь создают инструменты, а специалисты по статистическому моделированию строят на них сети. В 2014 году обучение простой модели под названием логистическая регрессия было серьёзной домашней задачей для новых аспирантов Университета Карнеги — Меллона. Сегодня на это уходит меньше десяти строк кода, и это по силам любому программисту.

Теперь о скрытых успехах. Именно из чтения рукописных адресов на почте взялось знаменитое собрание рукописных цифр MNIST. Такое же чтение обрабатывает чеки, зачисляемые в банк, а похожие модели оценивают, вернут ли заявители кредит. Платежи по картам автоматически проверяются на мошенничество, и эта проверка — основа платёжных систем вроде PayPal, Stripe, AliPay, WeChat, Apple, Visa и MasterCard. Шахматные программы сильны уже десятилетиями. А машинное обучение управляет поиском, рекомендациями, персонализацией и ранжированием по всему интернету. Короче говоря, говорит книга, машинное обучение повсюду, хотя в основном незаметно.

Каждый раз, когда вашу карту за границей блокируют, пока вы не подтвердите сообщение, модель мошенничества посмотрела на этот платёж и сочла его необычным.

Лишь недавно искусственный интеллект вышел на свет рампы, в основном благодаря решению задач, которые казались недостижимыми и с которыми люди сталкиваются напрямую; многие из этих достижений приписывают глубокому обучению. Первое и самое заметное — голосовой помощник. Siri, Alexa и ассистент Google довольно надёжно отвечают на устные просьбы — от включения света до записи на стрижку или разговора со службой поддержки. В их основе — распознавание речи, которое улучшилось настолько, что в некоторых применениях сравнялось по точности с человеком. Когда вы надиктовываете сообщение на шумной улице и оно выходит правильным, вы пользуетесь этим улучшением.

ОШИБКА ТОП-5 IMAGENET



С 28% до 2,25%

Распознавание объектов на картинках прошло столь же долгий путь. Книга приводит числа с эталонного набора ImageNet, где модели даётся пять догадок на каждую картинку и ошибкой считается, только если ни одна из пяти не верна. В 2010 году у лучшей команды, из NEC Labs и Иллинойского университета, доля ошибок топ-5 составляла 28%. К 2017 году она снизилась до 2,25%. Из тысячи картинок это примерно $0,28 \times 1000 = 280$ ошибок в 2010 году и лишь около 22 в 2017-м. Ошибка уменьшилась в $28 / 2,25 \approx 12$ раз за семь лет. Похожие скачки произошли в распознавании птиц по пению и в диагностике рака кожи.

Игры давно служили мерилом человеческих способностей, и книга прослеживает их лестницу. Она начинается с TD-Gammon, программы для игры в нарды, которая училась одной из форм обучения с подкреплением. Шахматы гораздо сложнее нарды, и Deep Blue от IBM обыграл чемпиона мира Гарри Каспарова с помощью массовых параллельных вычислений, специального оборудования и хитрого перебора возможных ходов. Го ещё труднее, потому что число возможных позиций огромно; AlphaGo достигла уровня человека в 2015 году, соединив глубокое обучение с методом выборки вероятных продолжений, называемым поиском по дереву методом Монте-Карло. Покер добавил ещё одну трудность: игра велика и видна лишь частично, потому что карты других игроков не видны. Программа под названием Libratus тем не менее обыграла людей, используя тщательно устроенные стратегии.

ТРУДНЕЕ



ПОКЕР



СКРЫТО

ГО

ШАХМАТЫ

НАРДЫ

Лестница игр

Опыт

Подумайте, как вы играете в крестики-нолики с ребёнком. Вы видите всё поле, поэтому можете продумать каждый ход: это как шахматы или го. А теперь подумайте о карточной игре, где рука соперника скрыта; вам приходится догадываться по тому, как он ставит или ходит. Именно эта недостающая информация сделала покер особым испытанием для компьютеров.

Беспилотные автомобили — ещё один признак прогресса. Полной автономности пока нет, но такие компании, как Tesla,

NVIDIA и Waymo, продают машины, которые могут частично ездить сами. Трудность в том, что хорошее вождение требует восприятия, рассуждения и следования правилам — всего в одной системе. Пока что, говорит книга, глубокое обучение в основном отвечает за зрение; остальное тщательно настраивают инженеры.



ИИ УЖЕ РЕШАЕТ: КРЕДИТ, ЗАЛОГ

Вождению нужно больше, чем зрение

И это лишь малая выборка. Робототехника, логистика, изучение живых клеток, физика частиц и астрономия обязаны некоторыми недавними успехами машинному обучению, которое становится повседневным инструментом инженеров и учёных.

С успехом приходят и опасения. Популярные статьи часто спрашивают, не грядёт ли «апокалипсис ИИ» или «сингулярность» — момент, когда машины обретут сознание и начнут решать за нас, без своих программистов. Ответ книги начинается с трезвого наблюдения: ИИ уже напрямую влияет на жизнь людей. Дадут ли вам кредит, часто решается автоматически; автопилоты выполняют большую часть полёта; а решения об освобождении обвиняемого под залог используют статистические данные. И, если о приятном, мы можем попросить Alexa включить кофеварку.

Запомним

Инструменты сделали машинное обучение гораздо проще: модель, которая в 2014 году была домашней задачей аспиранта, теперь занимает меньше десяти строк кода. Десятилетиями оно работало незаметно: читало адреса и чеки, оценивало кредитоспособность, ловило мошенников и ранжировало результаты поиска. Глубокое обучение принесло видимые успехи: голосовых помощников, распознавание речи на уровне, близком к человеческому, в некоторых применениях и распознавание изображений, у которого ошибка на ImageNet упала с 28% до 2,25%. Игры от нарда до шахмат, го и покера пали одна за другой, а беспилотные автомобили используют глубокое обучение в основном для зрения. ИИ уже влияет на реальные решения о людях, например о кредитах и залоге.

Вопросы

1. Почему в лекции сказано, что машинное обучение повсюду, хотя в основном незаметно?
 - a. Потому что оно работает только в лабораториях
 - b. Потому что десятилетиями оно тихо читает почту, проверяет платежи и ранжирует результаты поиска
 - c. Потому что оно стало возможным только в 2017 году
 - d. Потому что люди отказываются им пользоваться
2. Когда в доле ошибок топ-5 на ImageNet ответ модели считается неверным?
 - a. Когда неверна её первая догадка
 - b. Когда неверна любая из её пяти догадок
 - c. Когда ни одна из её пяти догадок не верна
 - d. Когда она думает дольше пяти секунд
3. Чем покер оказался испытанием, отличным от шахмат и го?
 - a. Игра видна лишь частично, потому что карты других игроков скрыты
 - b. В нём меньше возможных позиций, чем в шахматах
 - c. В него не могут играть компьютеры
 - d. В нём нет правил
4. За какую часть беспилотного вождения, по словам лекции, сегодня в основном отвечает глубокое обучение?
 - a. За соблюдение правил дорожного движения

- b. За планирование маршрута
- c. За рассуждения о других водителях
- d. За зрение, то есть восприятие

Ответы

1. **b)** Сортировка почты, чтение чеков, оценка кредитоспособности, проверка на мошенничество, поиск и рекомендации — всё это машинное обучение, которого никто не видит.
2. **c)** Модели даётся пять догадок на каждую картинку, и ошибкой считается, только если ни одна из пяти не верна.
3. **a)** Покер велик и виден лишь частично, потому что карты других игроков не видны.
4. **d)** Хорошему вождению нужны восприятие, рассуждение и правила; глубокое обучение в основном отвечает за зрение, а остальное настраивают инженеры.

Разобранные примеры

Как далеко продвинулось распознавание изображений

Доля ошибок топ-5 на ImageNet была 28% в 2010 году и 2,25% в 2017-м. Сколько из 1000 картинок модель не распознала в каждом году и во сколько раз уменьшилась ошибка?

2010: $0,28 \times 1000 = 280$ ошибок.

2017: $0,0225 \times 1000 = 22,5$, то есть около 22 ошибок.

Отношение: $28 \div 2,25 \approx 12,4$, так что за семь лет ошибка уменьшилась примерно в двенадцать раз.

Ответ: Около 280 ошибок в 2010 году, около 22 в 2017-м; примерно в двенадцать раз меньше

Задачи

1. Доля ошибок топ-5 у модели на 2000 картинках составляет 5%. Сколько картинок она не распознаёт?

Подсказка: Возьмите 5 процентов от 2000.

Ответ: $0,05 \times 2000 = 100$ картинок — тех, для которых ни одна из её пяти догадок не верна.

2. Назовите три способа, которыми, по словам лекции, ИИ уже напрямую влияет на жизнь людей.

Подсказка: Подумайте о деньгах, полётах и судах.

Ответ: Кредит часто решается автоматически, автопилоты выполняют большую часть полёта, а решения об освобождении под залог используют статистические данные.

Чем ИИ не является и что делает обучение глубоким

Спросите чат-бота, не собирается ли он захватить мир, и он может дать вдумчивый, даже жутковатый ответ. Звучит так, будто говорит кто-то со своим собственным разумом. Вердикт книги об этом впечатлении спокоен: мы далеки от машины, которая могла бы сознательно манипулировать людьми, создавшими её.

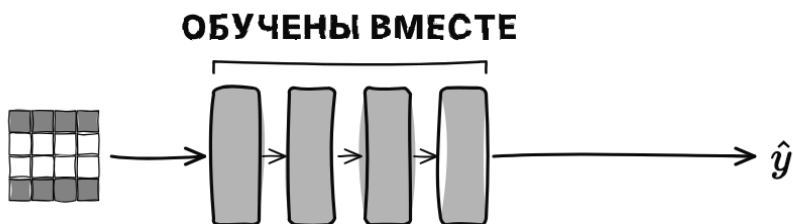
Авторы называют две причины. Во-первых, системы ИИ проектируют, обучают и развёртывают под определённые цели. Их поведение может создавать иллюзию общего интеллекта, но под ним — смесь правил, эвристик и статистических моделей. Во-вторых, сейчас просто нет инструментов для общего искусственного интеллекта — систем, которые могли бы совершенствовать себя, рассуждать о себе и перестраивать собственное устройство, пытаясь решить любую задачу. Жутковатый ответ чат-бота — это, как мы слышали раньше, текст, предсказанный по другому тексту, а не план.

Настоящая забота, говорит книга, — то, как ИИ используется в повседневной жизни. Многие рутинные работы, которые сейчас делают люди, могут быть и будут автоматизированы.

Сельскохозяйственные роботы, вероятно, удешевят органическое земледелие, но они же возьмут на себя уборку урожая. Этот этап промышленной революции может сильно ударить по большим слоям общества, потому что во многих странах простой ручной труд даёт очень много рабочих мест. А статистические модели,

применённые небрежно, могут по-разному обращаться с людьми в зависимости от расы, пола или возраста, и это поднимает справедливые вопросы, когда такие модели автоматически принимают важные решения. Эти алгоритмы нужно применять осторожно, и авторы считают это гораздо более насущным, чем любой злой сверхинтеллект, уничтожающий человечество.

Теперь книга переходит к тому, что такое глубокое обучение на самом деле. Глубокое обучение — это часть машинного обучения, построенная на нейронных сетях со многими слоями. Оно глубокое именно в этом смысле: его модели выучивают много слоёв преобразований, одно за другим. Можно возразить, что слои есть в любом машинном обучении, начиная с шагов, подготавливающих признаки. Глубокое обучение отличается тем, что каждый слой выучивается, и все они выучиваются вместе, по данным.



СКВОЗНОЕ

Ручные признаки и сквозное обучение

Задачи, которые мы встречали, — звуковые волны, необработанные пиксели, предложения любой длины для перевода — как раз те, где глубокое обучение блистает, а старые методы буксовали. Многослойные модели могут работать прямо с необработанными низкоуровневыми данными от органов чувств так, как прежние инструменты не могли. Общая нить, говорит книга, — сквозное обучение. Вместо того чтобы собирать систему из частей, каждая из которых настроена отдельно, вы строите всю систему и настраиваете её целиком вместе.

Самый ясный пример — компьютерное зрение. Годами учёные разделяли две работы: конструирование признаков, то есть ручное решение, как превратить картинку в числа, и построение

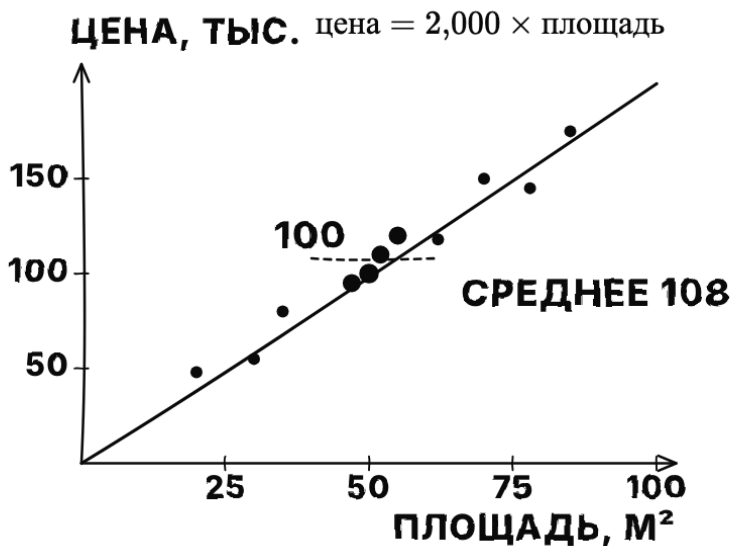
модели, которая учится на этих числах. Ручные инструменты вроде детектора границ Кэнни и метода SIFT Лоу господствовали больше десятилетия. Самой хитрой частью работы было изобрести такие преобразования, чтобы простая неглубокая модель могла справиться. Но человеческая изобретательность может перепробовать не так уж много по сравнению с алгоритмом, который последовательно и автоматически проверяет миллионы вариантов. Когда пришло глубокое обучение, ручные экстракторы признаков уступили место автоматически настраиваемым фильтрам, и точность выросла. Фильтры для лица в приложении видеозвонков находят ваши глаза и рот с помощью признаков, которые никто не рисовал вручную; сеть выучила их на примерах.

Значит, глубокое обучение заменяет сразу две вещи: неглубокую модель в конце старого конвейера и трудоёмкое конструирование признаков в его начале. Поскольку оно избавляет от столь большой части подготовки, специфичной для каждой области, оно ещё и разрушило стены. У зрения, распознавания речи, обработки языка и медицинской информатики раньше были свои отдельные наборы инструментов; глубокое обучение даёт им один общий набор.

Опыт

Попробуйте описать письменными правилами, как найти край кофейной кружки на фотографии: «там, где тёмное встречается со светлым», например. Затем представьте белую кружку на белом столе или кружку в тени — и посмотрите, как ваше правило сломается. Инженер 2000-х годов годами оттачивал такие правила; глубокая сеть выучивает свои по тысячам фотографий кружек. Это и есть размен, который описывает книга: долой ручные признаки, да здравствуют выученные слои.

Книга называет и второй сдвиг: от параметрических моделей к непараметрическим. Параметрическая модель заранее задаёт форму ответа, оставляя заполнить несколько чисел, как прямая с её наклоном и начальным значением. Когда данных мало, такие упрощающие предположения — единственный способ получить полезную модель. Когда данных много, от предположений можно отказаться, и модель может следовать самим данным. Вот придуманный пример. Чтобы оценить квартиру, правило-прямая может сказать: $\text{цена} = 2000 \times \text{площадь}$, так что квартира в 50 квадратных метров стоит 100 000. Непараметрический подход вместо этого найдёт три проданные квартиры, больше всего похожие на эту, и усреднит их цены, скажем 95, 110 и 120 тысяч, что даёт $(95 + 110 + 120) / 3 \approx 108$ тысяч. При малом объёме данных прямая надёжнее; при данных о продажах целого города соседи учитывают то, чего прямая никогда бы не учла, например тихую улицу или вид из окна.



Прямая и соседи

Авторы сравнивают это с физикой середины прошлого века. Когда появились компьютеры, физикам больше не нужно было вручную решать упрощённые формулы поведения электронов; они могли численно моделировать исходные уравнения. Модели стали гораздо точнее, но часто труднее для понимания. Глубокое обучение платит ту же цену: гибкие модели лучше подходят к данным, но их труднее объяснить.

Запомним

Мы далеки от ИИ, обладающего сознанием: сегодняшние системы создаются под конкретные цели, а инструментов для машин, которые сами себя улучшают и перестраивают, пока нет. Насущные заботы — автоматизация рутинного труда и небрежные модели, несправедливо обращающиеся с людьми. Глубокое обучение — это много слоёв преобразований, все выученные вместе по данным и обучаемые сквозным образом. Оно заменяет и ручное конструирование признаков, и неглубокую модель и даёт зрению, речи и языку один общий набор инструментов. При обилии данных гибкие непараметрические модели заменяют упрощающие предположения, выигрывая в точности ценой некоторой потери понятности.

Вопросы

1. Почему, согласно книге, мы далеки от ИИ, который мог бы сознательно манипулировать своими создателями?
 - a. Потому что системы ИИ выключают на ночь
 - b. Потому что ИИ создают под определённые цели, а инструментов для самосовершенствующегося общего интеллекта пока нет
 - c. Потому что чат-боты не умеют писать текст
 - d. Потому что это запрещают законы

2. Что книга считает более насущным, чем злой сверхинтеллект?
- a. Нехватку памяти у компьютеров
 - b. Жутковатые ответы чат-ботов
 - c. Автоматизацию рутинного труда и небрежные модели, несправедливо обращающиеся с людьми
 - d. То, что игры становятся слишком лёгкими для компьютеров
3. Что отличает глубокое обучение, если слои, можно сказать, есть в любом машинном обучении?
- a. Каждый слой выучивается, и все они выучиваются вместе по данным
 - b. У него ровно два слоя
 - c. Его первый слой всегда проектируют вручную
 - d. Оно не использует данных
4. Какую цену, как и физические симуляции, платят гибкие непараметрические модели?
- a. Им нужно меньше данных
 - b. Они менее точны
 - c. Их невозможно вычислить
 - d. Их часто труднее понять

Ответы

1. b) Системы ИИ проектируют под определённые цели, это смесь правил и статистических моделей, а инструментов для систем, которые сами себя улучшают и перестраивают, пока нет.

2. c) Настоящая забота — как используется ИИ: автоматизация рутинных работ и небрежно применённые статистические модели, по-разному обращающиеся с людьми в зависимости от расы, пола или возраста.

3. a) Глубокое обучение отличается тем, что каждый слой выучивается, и все они выучиваются вместе, по данным.

4. d) Численное моделирование уравнений сделало модели в физике гораздо точнее, но часто труднее для понимания; гибкие модели глубокого обучения платят ту же цену.

Разобранные примеры

Прямая против соседей Правило-прямая оценивает квартиру по формуле $\text{цена} = 2000 \times \text{площадь}$. Три самые похожие проданные соседние квартиры для квартиры в 50 квадратных метров ушли за 95, 110 и 120 тысяч. Сравните цену по прямой со средним по соседям.

Прямая: $2000 \times 50 = 100\,000$.

Соседи: $(95 + 110 + 120) \div 3 = 325 \div 3 \approx 108,3$ тысячи.

Прямая задаёт форму заранее; среднее по соседям следует самим данным.

При обилии данных о продажах соседи могут учесть то, что упускает прямая, например тихую улицу.

Ответ: Прямая: 100 000; соседи: около 108 000

Задачи

1. По тому же правилу-прямой, $\text{цена} = 2000 \times \text{площадь}$, сколько стоит квартира в 65 квадратных метров? Если три её ближайшие соседки продались за 120, 126 и 138 тысяч, что даёт непараметрический подход?

Подсказка: Для прямой умножьте; для соседей усредните.

Ответ: Прямая: $2000 \times 65 = 130\,000$. Соседи: $(120 + 126 + 138) \div 3 = 384 \div 3 = 128$ тысяч.

2. Своими словами объясните, что сквозное обучение заменило в компьютерном зрении.

Подсказка: Подумайте о двух отдельных работах, которые раньше делали учёные.

Ответ: Раньше учёные вручную проектировали признаки с помощью инструментов вроде детекторов границ и SIFT, а затем отдельно обучали на них неглубокую модель. Сквозное обучение строит всю систему и настраивает её целиком вместе, так что сеть выучивает и свои признаки, и итоговую модель одновременно.

Пробуй и делись

Вспомните спуск по склону холма из начала курса, но теперь поле полно впадин и гребней. Вы можете оказаться в маленькой впадине, а не в самой глубокой долине, и не сможете доказать, что нашли лучшее место. Книга говорит, что глубокое обучение с этим смирилось.



Впадины и гребни

Прежние работы стремились к задачам, где лучший ответ можно найти и доказать. Глубокое обучение принимает ответы хорошие, а не идеальные, работает с запутанными, невыпуклыми

ландшафтами потерь, полными впадин, и готово пробовать что-то раньше, чем кто-нибудь сможет доказать, что это работает. Этот новый эмпиризм вместе с потоком талантливых новичков быстро дал практические методы, хотя часто ценой переделки или повторного изобретения инструментов, существовавших десятилетиями.

Сообщество к тому же гордится тем, что делится через границы университетов и компаний: библиотеки, модели и полностью обученные сети выпускаются открыто. В том же духе авторы делают блокноты своей книги свободными для использования и распространения, чтобы снизить барьеры для всех, кто хочет учиться. Многие модели, стоящие за приложениями в вашем телефоне, начинались с таких открыто опубликованных сетей.

Итог главы связывает нити воедино. Машинное обучение изучает, как компьютеры могут использовать опыт, обычно данные, чтобы лучше справляться с задачей; оно опирается на статистику, интеллектуальный анализ данных и оптимизацию и часто служит способом построения ИИ. Обучение представлений, одна из его ветвей, ищет хорошие способы автоматически представлять данные. Глубокое обучение выучивает представления во многих слоях и потому заменяет и неглубокую модель, и ручное конструирование признаков. Его недавний прогресс обеспечили обилие данных от дешёвых датчиков и интернета, гораздо больший объём вычислений, в основном благодаря GPU, и эффективные фреймворки, с которыми легко строить и настраивать целые системы.

На этом введение заканчивается: от правил, которые никогда не учатся, через многие виды задач обучения — к составляющим,

благодаря которым глубокое обучение заработало.

Запомним

Глубокое обучение принимает хорошие, неидеальные решения, работает с запутанными ландшафтами потерь и пробует идеи, прежде чем их доказать. Его сообщество открыто делится инструментами, моделями и обученными сетями, и эта книга бесплатна в том же духе. Машинное обучение использует опыт, чтобы становиться лучше; глубокое обучение выучивает многослойные представления и заменяет ручные признаки. Данные, GPU и хорошие фреймворки — три силы, стоящие за его недавним прогрессом.

Вопросы

1. Чем отношение глубокого обучения к оптимизации отличается от прежних работ?
 - a. Оно принимает только решения, доказанно лучшие
 - b. Оно принимает хорошие, но неидеальные решения и пробует что-то, прежде чем доказать
 - c. Оно избегает невыпуклых задач
 - d. Оно никогда не использует градиентный спуск
2. Что может случиться при спуске по ландшафту потерь, полному впадин и гребней?

- a. Вы всегда дойдёте до самой глубокой долины
 - b. Вы вообще не сможете сдвинуться
 - c. Потери перестанут зависеть от параметров
 - d. Вы можете оказаться в маленькой впадине, а не в самой глубокой долине
3. Какие три силы, стоящие за недавним прогрессом глубокого обучения, называет итог?
- a. Обилие данных, больше вычислений благодаря GPU и эффективные фреймворки
 - b. Ручные признаки, малые наборы данных и доказательства
 - c. Правила, базы данных и веб-браузеры
 - d. Шахматы, го и покер

Ответы

1. **b)** Прежние работы стремились к доказуемо лучшим ответам; глубокое обучение принимает ответы хорошие, а не идеальные, работает с запутанными ландшафтами и пробует идеи, прежде чем их доказать.
2. **d)** На поле, полном впадин, можно остановиться в маленькой впадине и не суметь доказать, что нашли лучшее место.
3. **a)** Недавний прогресс обеспечили обилие данных, гораздо больший объём вычислений, в основном благодаря GPU, и эффективные фреймворки, с которыми легко строить и настраивать целые системы.

Задачи

1. Своими словами объясните, что такое обучение представлений и как глубокое обучение развивает его дальше.

Подсказка: Подумайте о том, что конструирование признаков делало вручную.

Ответ: Обучение представлений ищет хорошие способы автоматически представлять данные, вместо того чтобы люди проектировали признаки вручную. Глубокое обучение выучивает такие представления во многих слоях, каждый из которых опирается на предыдущий, и это позволяет ему заменить и ручное конструирование признаков, и неглубокую модель в конце.

2. Почему открытая публикация обученных сетей, в духе, который описывает лекция, может ускорить прогресс?

Подсказка: Подумайте, кто может строить на основе общей модели.

Ответ: Когда библиотеки, модели и обученные сети выпускаются открыто, любой в университете или компании может начать с них, а не с нуля; это снижает порог входа и позволяет идеям быстро распространяться и улучшаться.