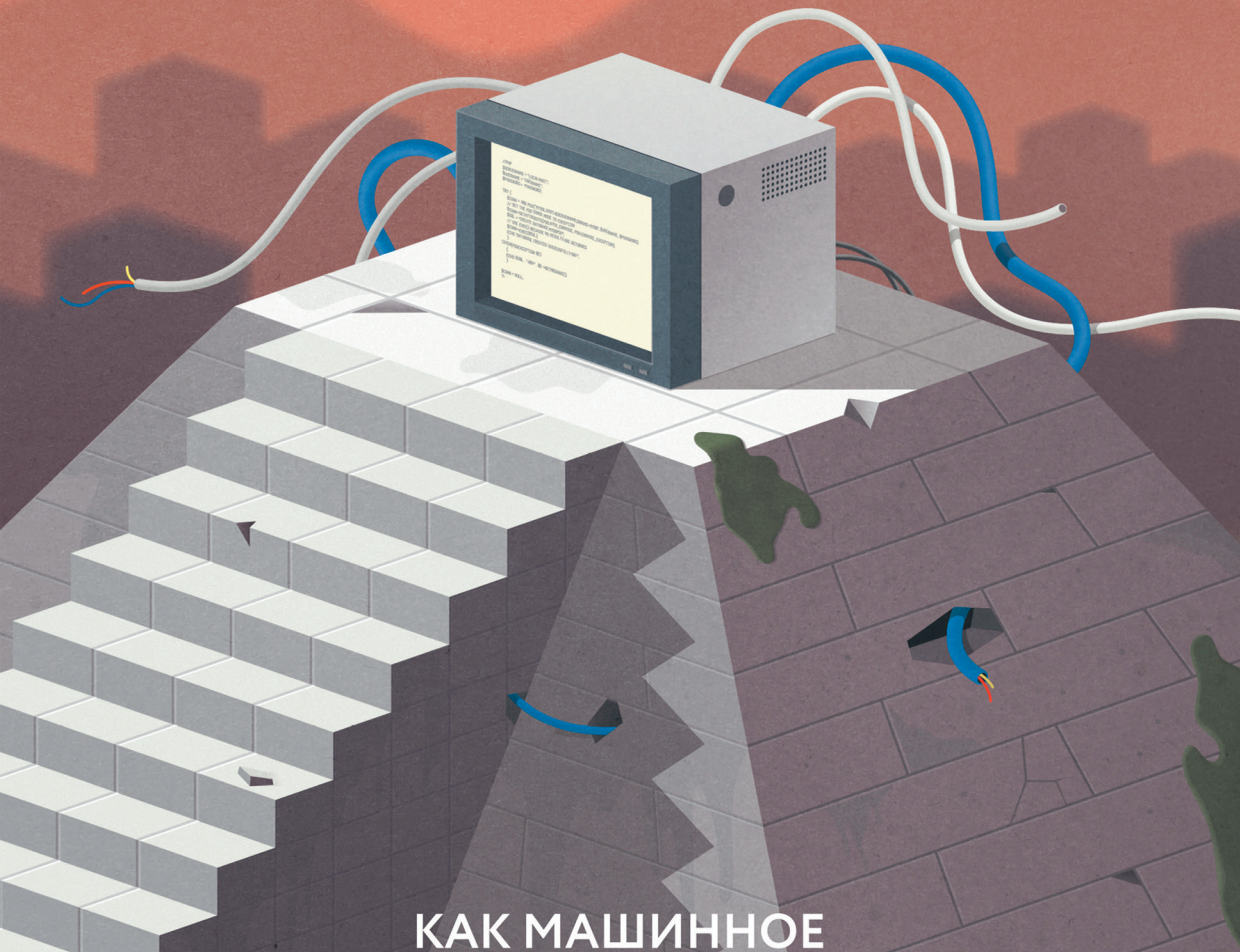
Педро Домингос

ВЕРХОВНЫЙ АЛГОРИТМ



КАК МАШИННОЕ ОБУЧЕНИЕ ИЗМЕНИТ НАШ МИР

ОГЛАВЛЕНИЕ

ГЛАВА 1

РЕВОЛЮЦИЯ МАШИННОГО ОБУЧЕНИЯ

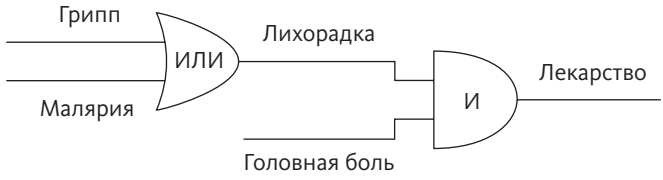
Мы живем в эпоху алгоритмов. Всего поколение-другое назад слово «алгоритм» у большинства людей вызвало бы лишь непонимание. Сегодня алгоритмы проникли во все уголки нашей цивилизации. Они вшиты в ткань повседневной жизни и нашли себе место не только в мобильных телефонах и ноутбуках, но и в автомобилях, квартирах, бытовой технике и игрушках. Так, банк — гигантское хитросплетение алгоритмов, а люди просто слегка регулируют настройки то тут, то там. Алгоритмы составляют расписание полетов, а затем ведут самолеты. Алгоритмы управляют производством, торговлей, снабжением, подсчитывают выручку и занимаются бухгалтерией. Если все алгоритмы вдруг перестанут работать, настанет конец света — такого, каким мы его знаем.

Алгоритм — определенная последовательность инструкций, диктующая компьютеру его действия. Компьютеры состоят из миллиардов крохотных переключателей — транзисторов, и алгоритмы включают и выключают эти транзисторы миллиарды раз в секунду.

Самый простой алгоритм — «нажми переключатель». Положение одного транзистора — одна единица информации: «один», если транзистор включен, и «ноль», если выключен. Единичка где-то в компьютерах банка информирует, превысили ли вы кредит. Еще одна единичка в недрах Управления социального обеспечения сообщает, живы вы или уже умерли.

Второй простейший алгоритм — «соедини два бита». Клод Шеннон, признанный отец теории информации, первым осознал, что включение и выключение транзисторов в ответ на действия других транзисторов — это,

в сущности, логический вывод. (Этой теме он посвятил свою дипломную работу в Массачусетском технологическом институте — самую важную дипломную работу в истории.) «Транзистор *A* включается, только если включены транзисторы *B* и *C*» — это крохотное логическое рассуждение. «*A* включается, когда включен либо *B*, либо *C*» — еще одна крупица логики. «*A* включается всегда, когда выключен *B*, и наоборот» — третья операция. Хотите верьте, хотите нет, любой алгоритм, как бы сложен он ни был, сводится всего к трем операциям: *И*, *ИЛИ* и НЕ. Используя для этих операций специальные символы, можно представить простые алгоритмы в виде диаграмм. Например, если у человека грипп или малярия и ему надо принять лекарство от температуры и головной боли, это можно выразить следующим образом:



Соединяя множество подобных операций, можно составлять очень сложные цепочки логических рассуждений. Люди часто думают, что вся суть компьютеров в вычислениях, но это не так. Сердце компьютеров — логика. Из логики в компьютере состоят и числа, и арифметика, и все остальное. Хотите сложить два числа? Есть комбинация транзисторов, которая это сделает. Хотите победить чемпиона в «Своей игре»? Для этого тоже найдется комбинация (естественно, она будет намного больше).

Однако строить новый компьютер для каждой новой задачи, которая нам придет в голову, было бы невероятно дорого, поэтому современный компьютер представляет собой большую совокупность транзисторов, способных решать много разных задач в зависимости от того, какие из них активны. Микеланджело говорил, что вся его работа — увидеть статую в глыбе мрамора и открыть ее миру, убрав лишнее. Аналогично алгоритмы «отсекают» избыточные транзисторы в компьютере, пока не обнажается нужная функция, будь то автопилот авиалайнера или новый мультфильм студии Pixar.

Алгоритм — не просто произвольный набор инструкций. Чтобы компьютер его выполнил, указания должны быть достаточно точными и однозначными.

Например, кулинарный рецепт — это не алгоритм, потому что не задает однозначного порядка действий и не объясняет, что делать на каждом этапе. Например, сколько именно сахара умещается в столовую ложку? Любой человек, который хоть раз пробовал готовить по незнакомому рецепту, знает, что может получиться и восхитительное блюдо, и не пойми что. А алгоритмы всегда дают идентичный результат. При этом, даже если указать в рецепте ровно 15 граммов сахара, это по-прежнему не решает проблему, потому что компьютер не знает ни что такое сахар, ни что такое грамм. Если бы мы захотели запрограммировать робота-повара для выпечки тортов, пришлось бы научить его узнавать сахар на видео, научить брать ложку и так далее (ученые все еще над этим работают). Компьютер должен знать, как выполнять алгоритм — вплоть до включения и выключения конкретных транзисторов, поэтому рецепт готовки очень далек от алгоритма.

С другой стороны, вот вам алгоритм игры в крестики-нолики.

Если вы или ваш противник поставили две отметки на одной линии, ставьте отметку в оставшейся на этой линии клетке.

Если такой ход невозможен, но есть ход, который создаст две линии по две отметки, — делайте его.

Если такой ход невозможен, но центральная клетка свободна, ставьте отметку в ней.

Если такой ход невозможен, но противник поставил отметку в углу, ставьте отметку в противоположном углу.

Если такой ход невозможен, но одна из угловых клеток свободна, ставьте отметку в ней.

Если такой ход невозможен, ставьте отметку в любой пустой клетке.

У этого алгоритма есть одно приятное свойство: он беспроигрышный! Конечно, ему не хватает многих деталей — как доска отображается в памяти компьютера и как это отображение меняется после каждого хода. Например, каждой клетке могут соответствовать два бита: 00 — если клетка пуста, 01 — если в ней поставили нолик и 10 — если крестик. Тем не менее предложенный алгоритм достаточно точен и однозначен, и любой грамотный программист сможет его дописать. Еще полезно не конкретизировать алгоритмы вплоть до отдельных транзисторов, а пользоваться уже существующими алгоритмами как кирпичиками. Их огромное количество, так что есть из чего выбирать.

Алгоритмы предъявляют строгие требования: часто говорят, что по-настоящему понимаешь что-то только тогда, когда можешь выразить это в виде алгоритма (как заметил Ричард Фейнман[1], «я не понимаю того, чего не могу создать»). Уравнения — хлеб насущный физиков и инженеров — на самом деле всего лишь особая разновидность алгоритмов. Например, второй закон Ньютона, который считают самым важным в мире уравнением, гласит, что для вычисления действующей на тело суммарной силы надо массу этого тела умножить на его ускорение. Он также подразумевает, что ускорение — это сила, разделенная на массу, но выведение этого следствия тоже алгоритм. Если теорию в любой научной дисциплине не получается выразить в виде алгоритма, она недостаточно строгая, не говоря уже о том, что ее решение нельзя компьютеризировать, а это всерьез ограничивает сферу ее применения. Ученые строят теории, инженеры изобретают устройства, а специалисты в области информатики создают алгоритмы, которые представляют собой и теории, и устройства одновременно.

Написать алгоритм непросто: есть очень много ловушек, и ни в чем нельзя быть уверенным. Интуитивные предположения вполне могут оказаться ошибочными, и тогда придется искать другой подход. Затем алгоритм надо выразить на понятном компьютеру языке, например Java или Python, и с этого момента алгоритм начнет называться программой. Потом программу надо отладить: найти все до единой ошибки и исправить их, пока компьютер не начнет выполнять ее без запинки. Но когда у вас наконец появится программа, которая умеет делает то, что вам нужно, вы получите все козыри. Компьютер станет послушно выполнять ваши задания миллионы раз со сверхвысокой скоростью. Созданной вами программой сможет пользоваться любой человек в мире. Она даже сделает вас миллиардером, если решенная проблема достаточно важна. Программист — человек, пишущий алгоритмы и кодирующий их, — маленький бог, создающий вселенные по своему желанию. Можно даже сказать, что сам Господь тоже был программистом, ведь в Книге Бытия он творил с помощью слов, а не руками. Речения стали мирами. Сегодня, сидя в кресле перед ноутбуком, вы тоже можете почувствовать себя богом: нарисуйте в воображении Вселенную и сделайте ее реальной. Законы физики соблюдать необязательно.

---

[1] Ричард Филлипс Фейнман (Richard Phillips Feynman, 1918–1988) — американский физик, лауреат Нобелевской премии, один из создателей современной квантовой электродинамики, внес существенный вклад в квантовую механику и квантовую теорию поля, его имя носит метод диаграмм Фейнмана.

Со временем информатики начинают опираться на уже проделанную работу и придумывают алгоритмы для все новых процессов. Одни алгоритмы соединяются с другими, чтобы использовать результаты третьих, производя, в свою очередь, еще больше алгоритмов. Каждую секунду миллиарды раз переключаются миллиарды транзисторов в миллиардах компьютеров. Алгоритмы образуют экосистему нового типа — непрерывно растущую и сопоставимую по богатству лишь с самой жизнью.

Однако, как это всегда бывает, в райском саду обитает змей — Монстр Сложности. У него, как у лернейской гидры, много голов. Одна из них — пространственная: количество битов информации, которое алгоритм должен хранить в памяти компьютера. Если алгоритму требуется больше памяти, чем есть в наличии, он бесполезен, и его приходится отбрасывать. У пространственной сложности есть злая сестрица: временна́я сложность. Сколько будет длиться выполнение алгоритма, то есть сколько раз нужно использовать транзисторы, прежде чем алгоритм даст желаемый результат? Если мы не можем столько ждать, алгоритм снова оказывается бесполезным. Но самая пугающая голова Монстра Сложности — сложность человеческая. Когда алгоритм становится слишком запутанным и непонятным для нашего скромного разума, а взаимодействия между его элементами — слишком многочисленными и обширными, в него начинают вкрадываться ошибки. Человек не в состоянии их отыскать и исправить, поэтому алгоритм не делает то, что от него требуется. Даже если каким-то образом заставить его работать, он окажется неоправданно сложным для пользователя, будет плохо взаимодействовать с другими алгоритмами и порождать все больше проблем.

Специалисты-информатики сражаются с Монстром Сложности каждый день. Когда они проигрывают, сложность прорывается в нашу жизнь. Вы, наверное, и сами замечали, как много было проиграно битв. Тем не менее башня алгоритмов продолжает расти, хотя строить ее все труднее: каждое новое поколение алгоритмов приходится возводить на вершине предшественников, их сложность суммируется. Башня растет и растет, алгоритмы опутывают весь мир, но конструкция становится все более шаткой — как карточный домик, который только и ждет толчка. Мизерная ошибка в алгоритме — и ракета, стоившая миллиард долларов, взрывается на старте, или миллионы людей остаются без электричества. Непредвиденное взаимодействие алгоритмов — и рушится фондовый рынок.

Если программисты — маленькие боги, то Монстр Сложности — его величество Сатана. И мало-помалу он выигрывает войну.

Должен быть способ лучше.

Познакомимся с обучающимся алгоритмом

У любого алгоритма есть вход и выход: данные поступают в компьютер, алгоритм делает с ними то, что должен, и выдает результат. Машинное обучение переворачивает все задом наперед: имея в своем распоряжении данные и желаемый результат, оно выдает алгоритм, превращающий одно в другое. Обучающиеся алгоритмы — те, что создают другие алгоритмы, обученные на основе данных. С помощью машинного обучения компьютеры пишут себе программы, и нам не надо этим заниматься.

Здорово, правда?

Компьютеры сами пишут для себя программы. Эта мысль потрясает настолько, что даже страшно: если компьютеры начнут программировать сами себя, сможем ли мы их контролировать? Оказывается — и мы в этом убедимся, — людям вполне по силам с ними совладать. Но есть и другое возражение — все это слишком хорошо, чтобы быть правдой. Разве для написания алгоритмов не нужны ум, творческая жилка, умение решать проблемы — все те качества, которых у компьютеров просто нет? Чем машинное обучение отличается от магии? Все это правда: сегодня мы умеем писать много программ, которым компьютер научиться не может. Но еще удивительнее то, что и компьютеры могут научиться программам, которые не в состоянии написать человек. Мы умеем водить машину или читать написанный от руки текст, но эти навыки у нас подсознательные: рассказать компьютеру, как это делать, не получится. Однако если дать обучающемуся алгоритму достаточное количество примеров каждого из этих действий, он с легкостью во всем разберется и без нашей помощи, и тогда можно будет развязать ему руки. Именно так машины научились читать почтовые индексы, и именно поэтому на дорогах скоро появятся автомобили без водителей.

Мощь машинного обучения, наверное, лучше всего показать, сравнив технологию с сельским хозяйством. В индустриальном обществе товары делают на заводах, а это значит, что инженерам надо точно определить, как именно их собирать, как изготавливать все элементы и так далее, вплоть до сырья. Это требует больших усилий. Самые сложные устройства, которые

человеку удалось изобрести, — компьютеры, и их разработка, производство и написание для них программ требуют колоссального труда. Но есть другой, намного более древний способ получить некоторые необходимые нам вещи: предоставить их изготовление самой природе. Посадить семечко, полить его, добавить удобрений, а потом сорвать спелый плод. Может ли технология выглядеть примерно так же? Может! Именно это сулит нам машинное обучение. Обучающиеся алгоритмы — как семена, почва — это данные, а обученные программы — это наша жатва. Эксперт по машинному обучению похож на крестьянина, сеющего, поливающего и удобряющего землю. Он присматривает за здоровьем растущего урожая, но в целом не вмешивается.

Если посмотреть на машинное обучение под этим углом, сразу бросаются в глаза два момента. Во-первых, чем больше у нас данных, тем больше мы можем узнать. Нет данных? Тогда и учиться нечему. Большой объем информации? Огромное поле для обучения. Вот почему машинное обучение заявляет о себе везде, где появляются экспоненциально растущие горы данных. Если бы в магазине продавали машинное обучение быстрого приготовления, на коробке было бы написано: «Просто добавь данных».

Второе наблюдение заключается в том, что машинное обучение — это меч-кладенец, которым можно обезглавить Монстра Сложности. Если дать обучающей программе длиной всего пару сотен строк достаточно данных, она не только с легкостью сгенерирует программу из миллионов строк кода, но и сможет делать это вновь и вновь для разных проблем. Уменьшение сложности для программиста просто феноменальное. Конечно, как и гидра, Монстр Сложности будет отращивать все новые и новые головы, но они окажутся меньше и вырастут не сразу, так что у нас все равно будет большое преимущество.

Машинное обучение можно представить себе как вывернутое наизнанку программирование, точно так же как квадратный корень противоположен возведению во вторую степень, а интегрирование обратно дифференцированию. Если можно спросить, квадрат какого числа равен 16 или производной какой функции является $x + 1$, уместен и вопрос: «Какой алгоритм даст такой результат?» Вскоре мы увидим, как превратить оба наблюдения в конкретные обучающиеся алгоритмы.

Некоторые обучающиеся алгоритмы добывают знания, а некоторые — навыки. «Все люди смертны» — это знание. Езда на велосипеде — навык. В машинном обучении знание часто предстает в форме статистических

моделей, потому что знание как таковое — это во многом статистика: смертны все люди, но только четыре процента людей американцы. Навыки зачастую представляют собой наборы процедур: если дорога сворачивает влево, поверни руль влево. Если перед тобой выскочил олень, дави на тормоз. (К сожалению, на момент написания этой книги беспилотная машина Google все еще путает оленей с полиэтиленовыми пакетами.) Часто процедура довольно проста, хотя заложенное в ней знание сложно. Спам надо отправить в корзину, однако сначала придется научиться отличать его от обычных писем. Если разобраться, какая позиция на шахматной доске удачна, станет ясно, какой сделать ход (тот, что приведет к лучшей позиции).

Машинное обучение принимает много разных форм и скрывается под разными именами: распознавание паттернов, статистическое моделирование, добыча данных, выявление знаний, предсказательная аналитика, наука о данных, адаптивные и самоорганизующиеся системы и так далее. Все они находят свое применение и имеют разные ассоциации. Некоторые живут долго, а некоторые не очень. Все это многообразие я буду называть просто — *машинное обучение*.

Машинное обучение иногда путают с искусственным интеллектом. С формальной точки зрения это действительно подраздел науки об искусственном интеллекте, однако он очень разросся и оказался настолько успешным, что затмил гордого родителя. Цель искусственного интеллекта — научить компьютеры делать то, что люди пока делают лучше, а умение учиться — наверное, самый важный из этих навыков, без которого компьютерам никогда не угнаться за человеком. Остальное приложится.

Если представить обработку данных в виде экосистемы, обучающиеся алгоритмы будут в ней суперхищниками. Базы данных, поисковые роботы, индексаторы и так далее — это травоядные, мирно пасущиеся на бескрайних лугах данных. Статистические алгоритмы, оперативная аналитическая обработка и так далее — просто хищники. Без травоядных не обойтись, потому что без них все остальное бы умерло, однако у суперхищника жизнь интереснее. Поисковый робот, как корова, пасется в интернете — поле мирового масштаба, а каждая страница в нем — травинка. Робот пощипывает травку, копии страниц оседают на его жестком диске. Затем индексатор создает список страниц, где встречается каждое слово, во многом как предметный указатель в конце книги. Базы данных похожи на слонов: они большие, тяжелые и никогда ни о чем не забывают. Среди этих степенных животных

носятся статистические и аналитические алгоритмы, которые сжимают, выбирают и превращают данные в информацию. Обучающиеся алгоритмы поглощают эту информацию, переваривают ее и дают нам знание.

Эксперты по машинному обучению — элита, каста священников среди ученых-информатиков. Многие компьютерщики, особенно старшего поколения, понимают машинное обучение не так хорошо, как им хотелось бы. Дело в том, что компьютерные науки традиционно следовали в русле детерминизма, а в машинном обучении нужно мыслить в категориях статистики. Если какое-то правило, скажем, отмечать определенные письма как спам, срабатывает в 99, а не в 100 процентах случаев, это не значит, что в нем какая-то ошибка: может быть, это лучшее, что можно сделать, и даже такая точность очень полезна. Различия в стиле мышления во многом послужили причиной, по которой Microsoft оказалось намного сложнее нагнать Google, чем в свое время Netscape. В конце концов, браузер всего лишь стандартная программа, а вот поисковая система требует другого склада ума.

Еще одна причина, по которой эксперты по машинному обучению слывут сверхумниками, заключается в том, что в мире их намного меньше, чем надо, даже по меркам компьютерных наук. Тим О'Райли, гуру в области технологий, утверждает, что «специалист по обработке данных» — самая востребованная вакансия в Кремниевой долине. По оценке McKinsey Global Institute, в 2018 году в одних только Соединенных Штатах спрос на экспертов по машинному обучению будет превышать предложение на 140–190 тысяч человек. Кроме того, потребуется дополнительно полтора миллиона разбирающихся в данных управленцев. Поток программ, связанных с машинным обучением, оказался слишком внезапным и мощным — система образования просто не успевает за спросом, к тому же машинное обучение считается трудной специальностью, и учебники вполне могут вызвать неприятие математики. Однако сложность скорее мнимая: все важнейшие идеи машинного обучения можно выразить и без математики. Читая эту книгу, вы, может быть, даже поймаете себя на том, что изобретаете обучающиеся алгоритмы без всяких уравнений.

Промышленная революция автоматизировала ручной труд, информационная революция проделала то же с трудом умственным, а машинное обучение автоматизировало саму автоматизацию. Без него программирование стало бы узким горлом, сдерживающим прогресс. Если вы ленивый и не слишком сообразительный компьютерщик, машинное обучение для вас — идеальная специальность, потому что обучающиеся алгоритмы сделают всю работу

сами, а вам достанутся только лавры. С другой стороны, обучающиеся алгоритмы могут оставить нас без работы, и поделом.

Подняв автоматизацию на невиданные высоты, революция машинного обучения вызовет огромные изменения в экономике и обществе, как в свое время интернет, персональные компьютеры, автомобили и паровой двигатель. Одна из областей, где изменения уже очевидны, — бизнес.

Почему бизнес рад машинному обучению?

Почему Google стоит намного дороже Yahoo? Обе компании зарабатывают на показе рекламы в интернете, и у той, и у другой прекрасная посещаемость, обе проводят аукционы по продаже рекламы и используют машинное обучение, чтобы предсказать, с какой вероятностью пользователь на нее кликнет (чем выше вероятность, тем ценнее реклама). Дело, однако, в том, что обучающиеся алгоритмы у Google намного совершеннее, чем у Yahoo. Конечно, это не единственная весьма серьезная причина разницы в капитализации. Каждый предсказанный, но не сделанный клик — упущенная возможность для рекламодателя и потерянная прибыль для поисковика. Учитывая, что годовая выручка Google составляет 50 миллиардов долларов, улучшение прогнозирования всего на один процент потенциально означает еще полмиллиарда долларов в год на банковском счету. Неудивительно, что Google — большая поклонница машинного обучения, а Yahoo и другие конкуренты изо всех сил пытаются за ней угнаться.

Реклама в сети — всего лишь один из аспектов более широкого явления. На любом рынке производители и потребители перед тем, как заключить сделку, должны выйти друг на друга. До появления интернета основные препятствия между ними были физическими: книгу можно было купить только в книжном магазине поблизости, а полки там не безразмерные. Однако теперь, когда книги можно в любой момент скачать на «читалку», проблемой становится колоссальное число вариантов. Как тут искать, если на полках книжного магазина стоят миллионы томов? Это верно и для других информационных продуктов: видео, музыки, новостей, твитов, блогов, старых добрых сайтов. Это также касается продуктов и услуг, которые можно получить на расстоянии: обуви, цветов, гаджетов, гостиничных номеров, обучения, инвестиций и даже поисков работы и спутника жизни. Как найти друг друга? Это определяющая проблема информационной эры, и машинное обучение помогает ее решить.

издательство
МАНН, ИВАНОВ И ФЕРБЕР
Максимально полезные книги