



Машинное обучение для людей

Разбираемся простыми словами

22 июля 2018 :: 241 комментарий :: 217768 просмотров :: 7048 слов

[Открыть изображение »](#)

Машинное обучение – как секс в старших классах. Все говорят о нем по углам, единицы понимают, а занимается только препод. Статьи о машинном обучении делятся на два типа: это либо трёхтомники с формулами и теоремами, которые я ни разу не смог дочитать даже до середины, либо сказки об *искусственном интеллекте, профессиях будущего и волшебных дата-саентистах*.

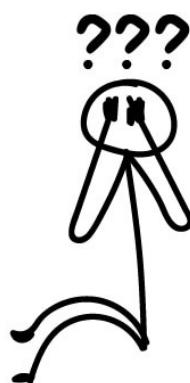
Решил сам написать пост, которого мне не хватало. Большое введение для тех, кто хочет наконец разобраться в

машинном обучении — простым языком, без формул-теорем, зато с примерами реальных задач и их решений.

Погнали.



Как мы видим,
тут всё очевидно



Есть два типа статей про машинное обучение

Программисты программируют!

Датасаенс!

Профессия будущего!

Буквально через пять лет...

Экспоненциально!!!

УМНЫЕ РОБОТЫ

A-A-A-A-A-A-A-A-A-A-aaa!!!!!!



Если вы хотите углубиться в машинное обучение и научиться применять его на практике, то ребята запустили курс «[Машинное обучение и анализ данных](#)» с преподавателями из Яндекса.

Только для читателей блога скидка 8600руб по промокоду VAS3K при оплате всей специализации сразу. Промокод действует до 15 октября 2018 года.

Зачем обучать машины

Снова разберём на Олегах.

Предположим, Олег хочет купить автомобиль и считает сколько денег ему нужно для этого накопить. Он пересмотрел десяток объявлений в интернете и увидел, что новые автомобили стоят около \$20 000, годовалые — примерно \$19 000, двухлетние — \$18 000 и так далее.



В уме Олег-аналитик выводит формулу: *адекватная цена автомобиля начинается от \$20 000 и падает на \$1000 каждый год, пока не упрётся в \$10 000.*

Олег сделал то, что в машинном обучении называют *регрессией* — предсказал цену по известным данным. Люди делают это постоянно, когда считают почём продать старый айфон или сколько шашлыка взять на дачу (моя формула — полкило на человека в сутки).

Да, было бы удобно иметь формулу под каждую проблему на свете. Но взять те же цены на автомобили: кроме пробега есть десятки комплектаций, разное техническое состояние, сезонность спроса и еще столько неочевидных факторов, которые Олег, даже при всём желании, не учел бы в голове.

Люди тупы и ленивы — надо заставить вкалывать роботов. Пусть машина посмотрит на наши данные, найдёт в них закономерности и научится предсказывать для нас ответ. Самое интересное, что в итоге она стала находить даже такие закономерности, о которых люди не догадывались.

Так родилось машинное обучение.

24 комментария

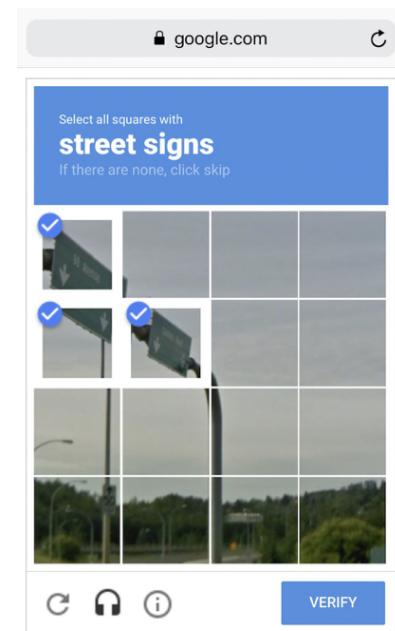
Три составляющие обучения

Цель машинного обучения — предсказать результат по входным данным. Чем разнообразнее входные данные, тем проще машине найти закономерности и тем точнее результат.

Итак, если мы хотим обучить машину, нам нужны три вещи:

Данные Хотим определять спам — нужны примеры спам-писем, предсказывать курс акций — нужна история цен, узнать интересы пользователя — нужны его лайки или посты. Данных нужно как можно больше. Десятки тысяч примеров — это самый злой минимум для отчаянных.

Данные собирают как могут. Кто-то вручную — получается дольше, меньше, зато без ошибок. Кто-то автоматически — просто сливает машине всё, что нашлось, и верит в лучшее. Самые хитрые, типа гугла, используют своих же пользователей для бесплатной разметки. Вспомните ReCaptcha, которая иногда требует «найти на фотографии все дорожные знаки» — это оно и есть.



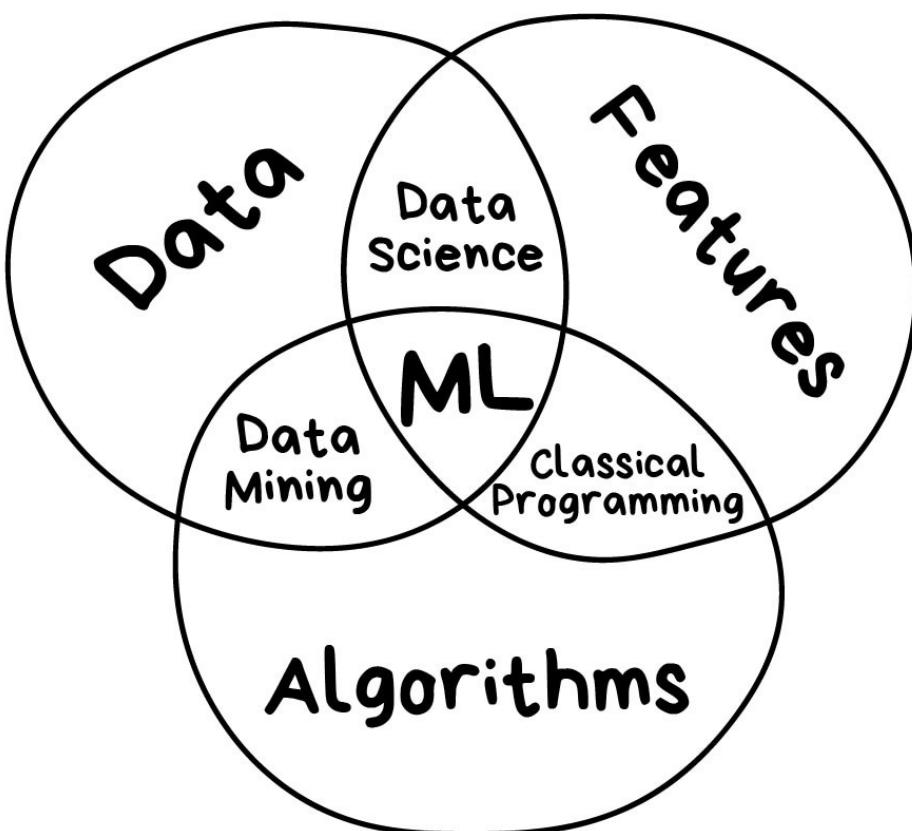
За хорошими наборами данных (датасетами) идёт большая охота. Крупные компании, бывает, раскрывают свои алгоритмы, но датасеты — крайне редко.

Признаки Мы называем их фичами (features), так что ненавистникам англизмов придётся страдать. Фичи, свойства, характеристики, признаки — ими могут быть пробег автомобиля, пол пользователя, цена акций, даже счетчик частоты появления слова в тексте может быть фичей.

Машина должна знать, на что ей конкретно смотреть.

Хорошо, когда данные просто лежат в табличках — названия их колонок и есть фичи. А если у нас сто гигабайт картинок с котами? Когда признаков много, модель работает медленно и неэффективно. Зачастую отбор правильных фич занимает больше времени, чем всё остальное обучение. Но бывают и обратные ситуации, когда кожаный мешок сам решает отобрать только «правильные» на его взгляд признаки и вносит в модель субъективность — она начинает дико врать.

Алгоритм Одну задачу можно решить разными методами примерно всегда. От выбора метода зависит точность, скорость работы и размер готовой модели. Но есть один нюанс: если данные говно, даже самый лучший алгоритм не поможет. Не зацикливайтесь на процентах, лучше соберите побольше данных.



Обучение vs Интеллект

Однажды в одном хипстерском издании я видел статью под заголовком «Заменят ли нейросети машинное обучение». Пиарщики в своих пресс-релизах обзывают «искусственным интеллектом» любую линейную регрессию, с которой уже дети во дворе играют. Объясняю разницу на картинке, раз и навсегда.



Искусственный интеллект — название всей области, как биология или химия.

Машинное обучение — это раздел искусственного

интеллекта. Важный, но не единственный.

Нейросети — один из видов машинного обучения.
Популярный, но есть и другие, не хуже.

Глубокое обучение — архитектура нейросетей, один из подходов к их построению и обучению. На практике сегодня мало кто отличает, где глубокие нейросети, а где не очень. Говорят название конкретной сети и всё.

Сравнивать можно только вещи одного уровня, иначе получается полный буллshit типа «что лучше: машина или колесо?» Не отождествляйте термины без причины, чтобы не выглядеть дурачком.

Вот что машины сегодня умеют, а что не под силу даже самым обученным.

Машина может

Машина не может

Предсказывать

Создавать новое

Запоминать

Резко поумнеть

Воспроизводить

Выйти за рамки задачи

Выбирать лучшее

Убить всех людей

Карта мира машинного обучения

Лень читать лонгрид — повтыкайте хотя бы в картинку, будет полезно.



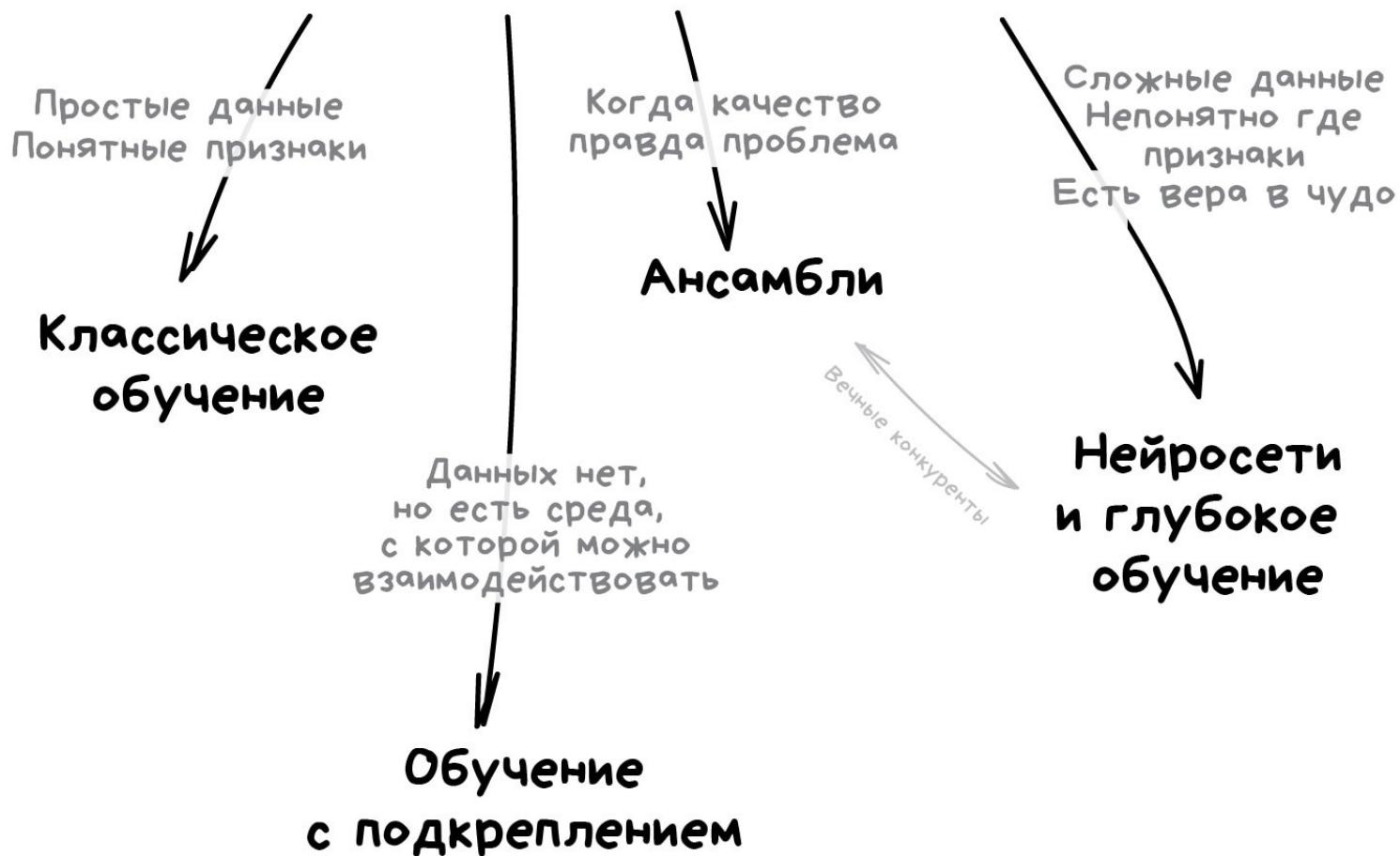
Думаю потом нарисовать полноценную настенную карту со стрелочками и объяснениями, что где используется, если статья зайдёт.

И да. Классифицировать алгоритмы можно десятком способов. Я выбрал этот, потому что он мне кажется самым

удобным для повествования. Надо понимать, что не бывает так, чтобы задачу решал только один метод. Я буду упоминать известные примеры применений, но держите в уме, что «сын маминой подруги» всё это может решить нейросетями.

Начну с базового обзора. Сегодня в машинном обучении есть всего четыре основных направления.

Основные виды машинного обучения



Часть 1. Классическое обучение

Первые алгоритмы пришли к нам из чистой статистики еще в 1950-х. Они решали формальные задачи — искали закономерности в циферках, оценивали близость точек в пространстве и вычисляли направления.

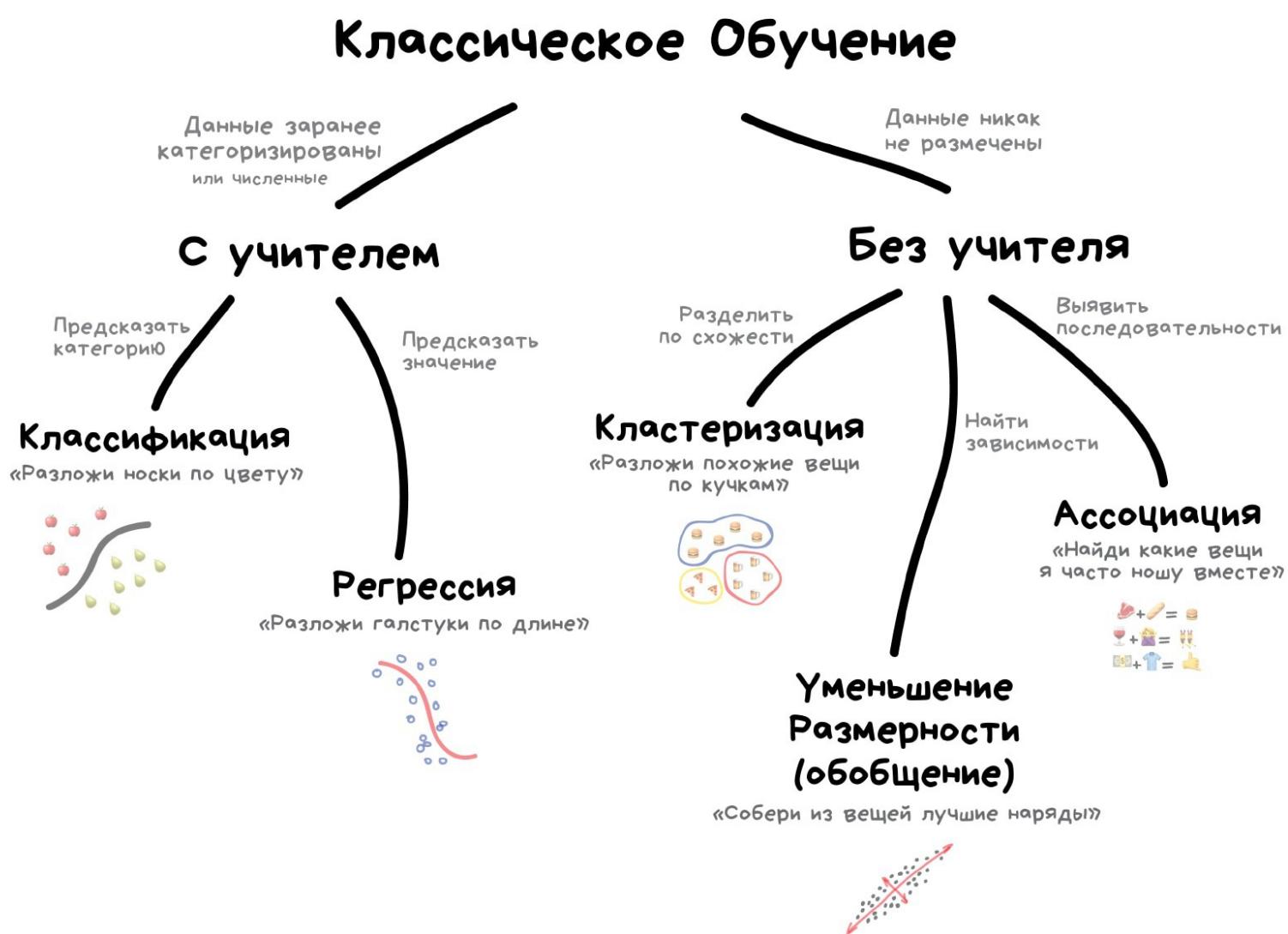
Сегодня на классических алгоритмах держится добрая половина интернета. Когда вы встречаете блок «Рекомендованные статьи» на сайте, или банк блокирует все ваши деньги на карточке после первой же покупки кофе за границей — это почти всегда дело рук одного из этих алгоритмов.

Да, крупные корпорации любят решать все проблемы нейросетями. Потому что лишние 2% точности для них легко конвертируются в дополнительные 2 миллиарда прибыли. Остальным же стоит включать голову. Когда задача решаема классическими методами, дешевле реализовать сколько-нибудь полезную для бизнеса систему на них, а потом думать об улучшениях. А если вы не решили задачу, то не решить её на 2% лучше вам не особо поможет.

Знаю несколько смешных историй, когда команда три

месяца переписывала систему рекомендаций интернет-магазина на более точный алгоритм, и только потом понимала, что покупатели вообще ей не пользуются. Большая часть просто приходит из поисковиков.

При всей своей популярности, классические алгоритмы настолько просты, что их легко объяснить даже ребёнку. Сегодня они как основы арифметики — пригождаются постоянно, но некоторые всё равно стали их забывать.



Обучение с учителем

Классическое обучение любят делить на две категории — с учителем и без. Часто можно встретить их английские наименования — Supervised и Unsupervised Learning.

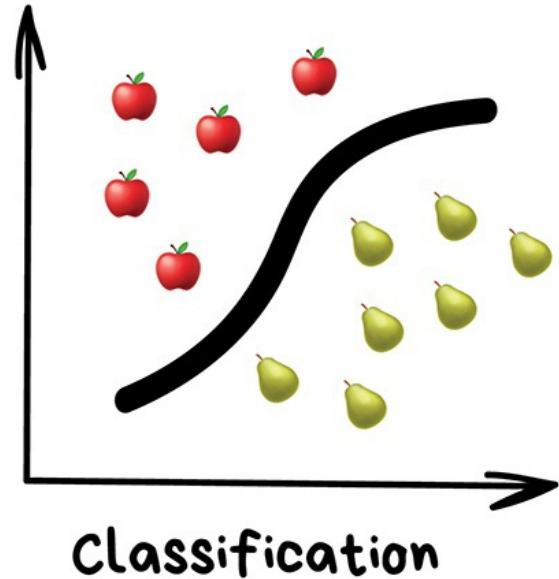
В первом случае у машины есть некий учитель, который говорит ей как правильно. Рассказывает, что на этой картинке кошка, а на этой собака. То есть учитель уже заранее разделил (разметил) все данные на кошек и собак, а машина учится на конкретных примерах.

В обучении без учителя, машине просто вываливают кучу фотографий животных на стол и говорят «разберись, кто здесь на кого похож». Данные не размечены, у машины нет учителя, и она пытается сама найти любые закономерности. Об этих методах поговорим ниже.

Очевидно, что с учителем машина обучится быстрее и точнее, потому в боевых задачах его используют намного чаще. Эти задачи делятся на два типа: **классификация — предсказание категории объекта, и регрессия — предсказание места на числовой прямой**.

Классификация

«Разделяет объекты по заранее известному признаку. Носки по цветам, документы по языкам, музыку по жанрам»



Сегодня используют для:

- Спам-фильтры
- Определение языка
- Поиск похожих документов
- Анализ тональности
- Распознавание рукописных букв и цифр
- Определение подозрительных транзакций

Популярные алгоритмы: Наивный Байес, Деревья Решений, Логистическая Регрессия, К-ближайших соседей, Машины Опорных Векторов

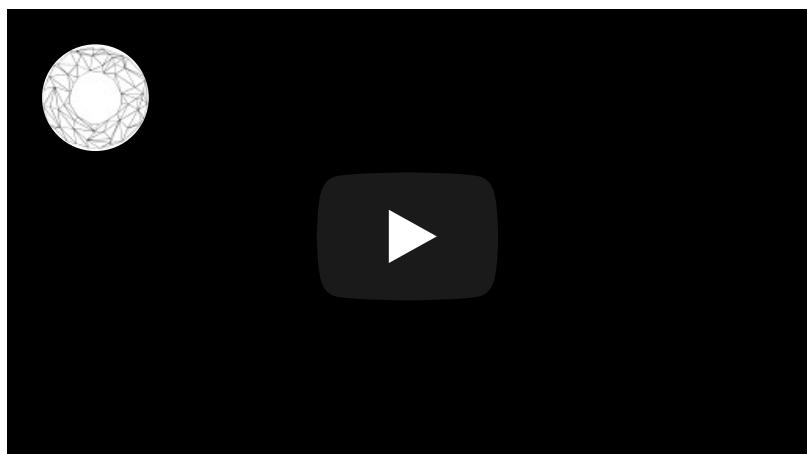
Здесь и далее в комментах можно дополнять эти блоки.

Приводите свои примеры задач, областей и алгоритмов, потому что описанные мной взяты из субъективного опыта.

5 комментариев

Классификация вещей — самая популярная задача во всём машинном обучении. Машина в ней как ребёнок, который учится раскладывать игрушки: роботов в один ящик, танки в другой. Опа, а если это робот-танк? Штош, время расплакаться и выпасть в ошибку.

Для классификации всегда нужен учитель — размеченные данные с признаками и категориями, которые машина будет учиться определять по этим признакам. Дальше классифицировать можно что угодно: пользователей по интересам — так делают алгоритмические ленты, статьи по языкам и тематикам — важно для поисковиков, музыку по жанрам — вспомните плейлисты Спотифая и Яндекс.Музыки, даже письма в вашем почтовом ящике.



Старый доклад Бобука про повышение конверсии лендингов с помощью SVM

Раньше все спам-фильтры работали на алгоритме Наивного

Байеса. Машина считала сколько раз слово «виагра» встречается в спаме, а сколько раз в нормальных письмах. Перемножала эти две вероятности по формуле Байеса, складывала результаты всех слов и бац, всем лежать, у нас машинное обучение!



Позже спамеры научились обходить фильтр Байеса, просто вставляя в конец письма много слов с «хорошими» рейтингами. Метод получил ироничное название Отравление Байеса, а фильтровать спам стали другими алгоритмами. Но метод навсегда остался в учебниках как самый простой, красивый и один из первых практически полезных.

Возьмем другой пример полезной классификации. Вот берёте вы кредит в банке. Как банку удостовериться, вернётё вы его или нет? Точно никак, но у банка есть тысячи профилей других людей, которые уже брали кредит до вас.

Там указан их возраст, образование, должность, уровень зарплаты и главное — кто из них вернул кредит, а с кем возникли проблемы.

Да, все догадались, где здесь данные и какой надо предсказать результат. Обучим машину, найдём закономерности, получим ответ — вопрос не в этом. Проблема в том, что банк не может слепо доверять ответу машины, без объяснений. Вдруг сбой, злые хакеры или бухой админ решил *скриптик исправить*.

Для этой задачи придумали Деревья Решений. Машина автоматически разделяет все данные по вопросам, ответы на которые «да» или «нет». Вопросы могут быть не совсем адекватными с точки зрения человека, например «*зарплата заёмщика больше, чем 25934 рубля?*», но машина придумывает их так, чтобы на каждом шаге разбиение было самым точным.

Так получается дерево вопросов. Чем выше уровень, тем более общий вопрос. Потом даже можно загнать их аналитикам, и они навыдумывают почему так.

Деревья нашли свою нишу в областях с высокой ответственностью: диагностике, медицине, финансах.

| Два самых популярных алгоритма построения деревьев — CART и C4.5.

В чистом виде деревья сегодня используют редко, но вот их ансамбли (о которых будет ниже) лежат в основе крупных систем и зачастую уделывают даже нейросети. Например, когда вы задаете вопрос Яндексу, именно толпа глупых деревьев бежит ранжировать вам результаты.

Давать ли кредит?



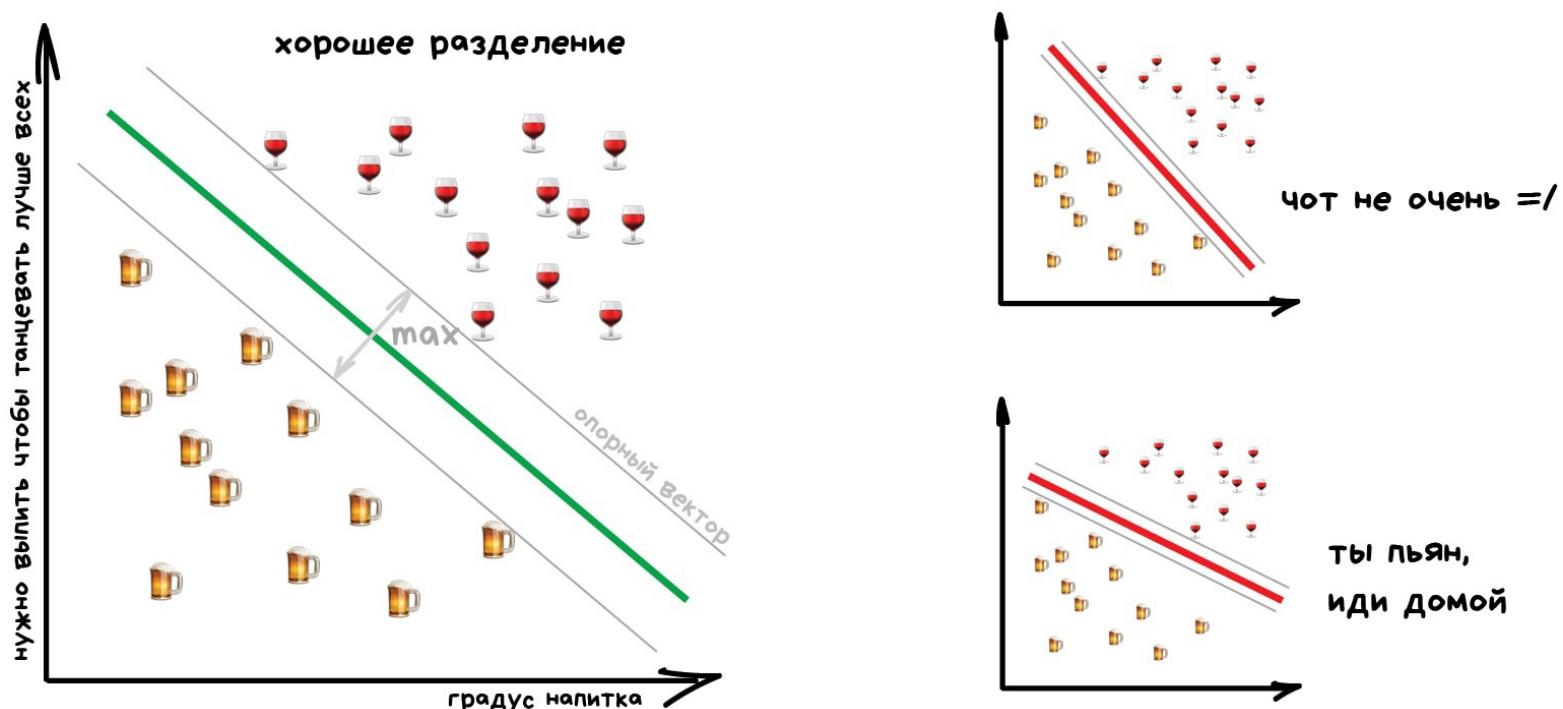
Дерево Решений

Но самым популярным методом классической классификации заслуженно является Метод Опорных Векторов (SVM). Им классифицировали уже всё: виды растений, лица на фотографиях, документы по тематикам, даже странных Playboу-моделей. Много лет он был главным ответом на вопрос «какой бы мне взять классификатор».

Идея SVM по своей сути проста — он ищет, как так провести две прямые между категориями, чтобы между ними

образовался наибольший зазор. На картинке видно нагляднее:

Разделяем виды алкоголя



Метод Опорных Векторов

У классификации есть полезная обратная сторона — поиск аномалий. Когда какой-то признак объекта сильно не вписывается в наши классы, мы ярко подсвечиваем его на экране. Сейчас так делают в медицине: компьютер подсвечивает врачу все подозрительные области МРТ или выделяет отклонения в анализах. На биржах таким же образом определяют нестандартных игроков, которые скорее всего являются инсайдерами. Научив компьютер «как правильно», мы автоматически получаем и обратный классификатор — как неправильно.

Сегодня для классификации всё чаще используют нейросети, ведь по сути их для этого и изобрели.

Правило буравчика такое: сложнее данные — сложнее алгоритм. Для текста, цифр, табличек я бы начинал с классики. Там модели меньше, обучаются быстрее и работают понятнее. Для картинок, видео и другой непонятной бигдаты — сразу смотрел бы в сторону нейросетей.

Лет пять назад еще можно было встретить классификатор лиц на SVM, но сегодня под эту задачу сотня готовых сеток по интернету валяются, чо бы их не взять. А вот спам-фильтры как на SVM писали, так и не вижу смысла останавливаться.

7 комментариев

Регрессия

«Нарисуй линию вдоль моих точек. Да, это машинное обучение»

Сегодня используют для:

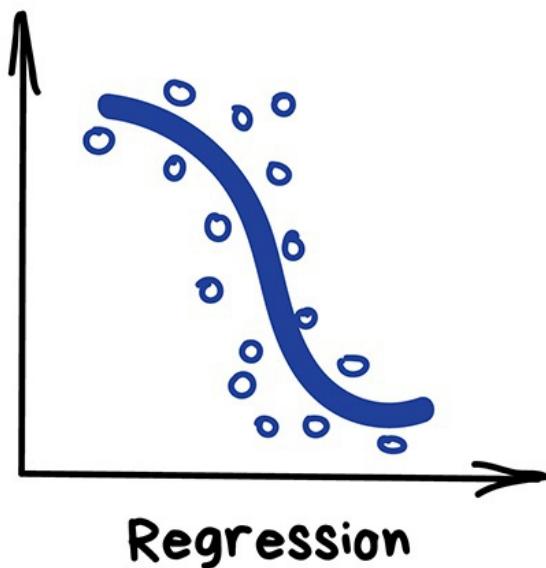
- Прогноз стоимости ценных бумаг
- Анализ спроса, объема продаж

- Медицинские диагнозы
- Любые зависимости числа от времени

Популярные алгоритмы:

Линейная или

Полиномиальная Регрессия

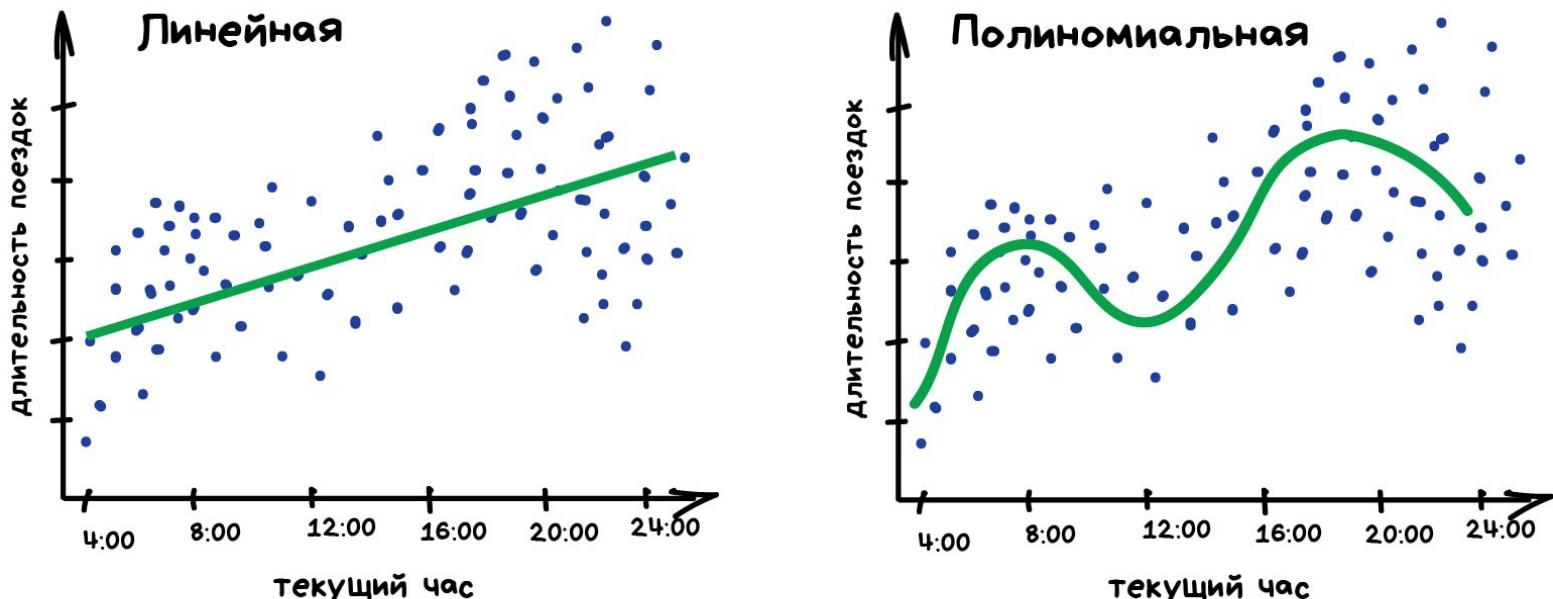


2 комментария

Регрессия — та же классификация, только вместо категории мы предсказываем число. Стоимость автомобиля по его пробегу, количество пробок по времени суток, объем спроса на товар от роста компании и.т.д. На регрессию идеально ложатся любые задачи, где есть зависимость от времени.

Регрессию очень любят финансисты и аналитики, она встроена даже в Excel. Внутри всё работает, опять же, банально: машина тупо пытается нарисовать линию, которая в среднем отражает зависимость. Правда, в отличии от человека с фломастером и вайтбордом, делает она это математически точно — считая среднее расстояние до каждой точки и пытаясь всем угодить.

Предсказываем пробки



Регрессия

Когда регрессия рисует прямую линию, её называют линейной, когда кривую — полиномиальной. Это два основных вида регрессии, дальше уже начинаются редкоземельные методы. Но так как в семье не без урода, есть Логистическая Регрессия, которая на самом деле не регрессия, а метод классификации, от чего у всех постоянно путаница. Не делайте так.

Схожесть регрессии и классификации подтверждается еще и тем, что многие классификаторы, после небольшого тюнинга, превращаются в регрессоры. Например, мы можем не просто смотреть к какому классу принадлежит объект, а запоминать, насколько он близок — и вот, у нас регрессия.

Для желающих понять это глубже, но тоже простыми словами, рекомендую цикл статей [Machine Learning for Humans](#)

Обучение без учителя

Обучение без учителя (Unsupervised Learning) было изобретено позже, аж в 90-е, и на практике используется реже. Но бывают задачи, где у нас просто нет выбора.

Размеченные данные, как я сказал, дорогая редкость. Но что делать если я хочу, например, написать классификатор автобусов — идти на улицу руками фотографировать миллион сраных икарусов и подписывать где какой? Так и жизнь вся пройдёт, а у меня еще игры в стиме не пройдены.

Когда нет разметки, есть надежда на капитализм, социальное расслоение и миллион китайцев из сервисов типа [Яндекс.Толока](#), которые готовы делать для вас что угодно за пять центов. Так обычно и поступают на практике. А вы думали где Яндекс берёт все свои крутые датасеты?

Либо, можно попробовать обучение без учителя. Хотя, честно говоря, из своей практики я не помню чтобы где-то оно сработало хорошо.

Обучение без учителя, всё же, чаще используют как метод анализа данных, а не как основной алгоритм. Специальный кожаный мешок с дипломом МГУ вбрасывает туда кучу мусора и наблюдает. Кластеры есть? Зависимости появились? Нет? Ну штош, продолжай, труд освобождает. Тыж хотел работать в датасаенсе.

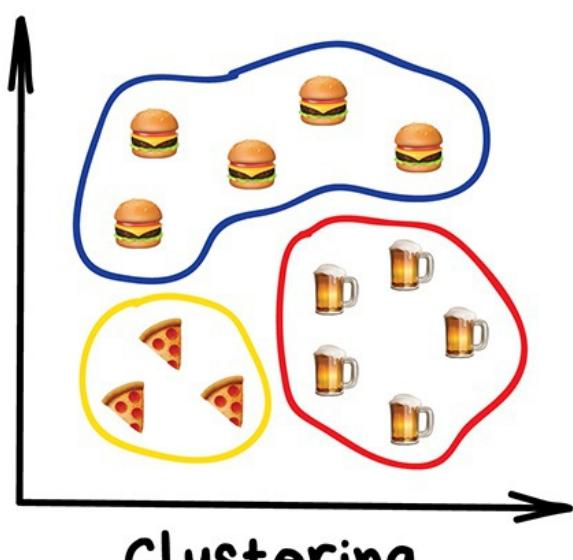
5 комментариев

Кластеризация

«Разделяет объекты по неизвестному признаку. Машина сама решает как лучше»

Сегодня используют для:

- Сегментация рынка
(типов покупателей,
лояльности)
- Объединение близких точек на карте
- Сжатие изображений
- Анализ и разметки новых данных



- Детекторы аномального поведения

Популярные алгоритмы: Метод K-средних, Mean-Shift, DBSCAN

+ добавить комментарий

Кластеризация — это классификация, но без заранее известных классов. Она сама ищет похожие объекты и объединяет их в кластеры. Количество кластеров можно задать заранее или доверить это машине. Похожесть объектов машина определяет по тем признакам, которые мы ей разметили — у кого много схожих характеристик, тех давай в один класс.

Отличный пример кластеризации — маркеры на картах в вебе. Когда вы ищете все крафтовые бары в Москве, движку приходится группировать их в кружочки с циферкой, иначе браузер зависнет в потугах нарисовать миллион маркеров.

Более сложные примеры кластеризации можно вспомнить в приложениях iPhoto или Google Photos, которые находят лица людей на фотографиях и группируют их в альбомы. Приложение не знает как зовут ваших друзей, но может отличить их по характерным чертам лица. Типичная

кластеризация.

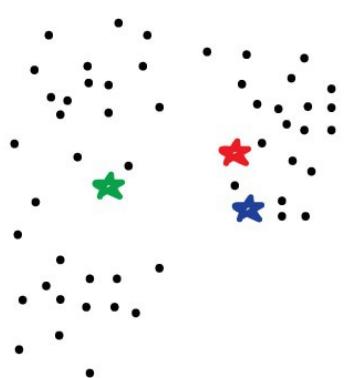
Правда для начала им приходится найти эти самые «характерные черты», а это уже только с учителем.

Сжатие изображений — еще одна популярная проблема. Сохраняя картинку в PNG, вы можете установить палитру, скажем, в 32 цвета. Тогда кластеризация найдёт все «примерно красные» пиксели изображения, высчитает из них «средний красный по больнице» и заменит все красные на него. Меньше цветов — меньше файл.

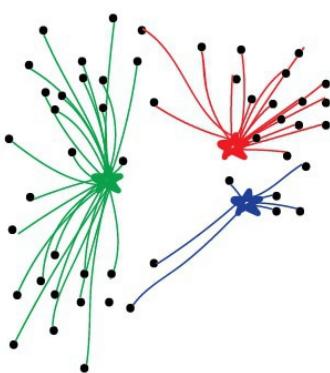
Проблема только, как быть с цветами типа Суан  — вот он ближе к зеленому или синему? Тут нам поможет популярный алгоритм кластеризации — Метод К-средних (K-Means). Мы случайным образом бросаем на палитру цветов наши 32 точки, обзываая их центроидами. Все остальные точки относим к ближайшему центроиду от них — получаются как бы созвездия из самых близких цветов. Затем двигаем центроид в центр своего созвездия и повторяем пока центроиды не перестанут двигаться. Кластеры обнаружены, стабильны и их ровно 32 как и надо было.

Ставим три ларька с шаурмой оптимальным образом

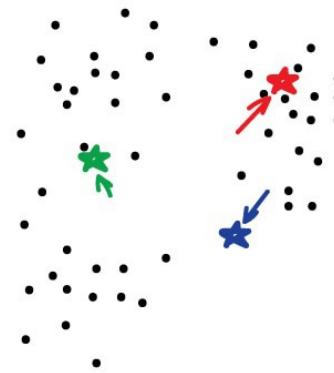
(илюстрируя метод К-средних)



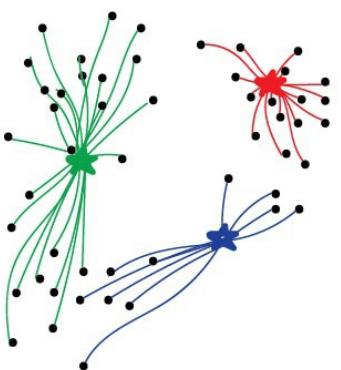
1. Ставим ларьки с шаурмой в случайных местах



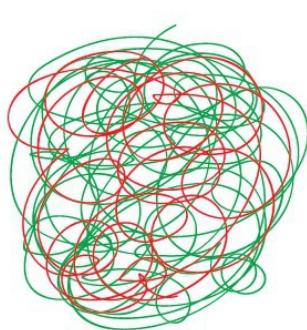
2. Смотрим в какой кому ближе идти



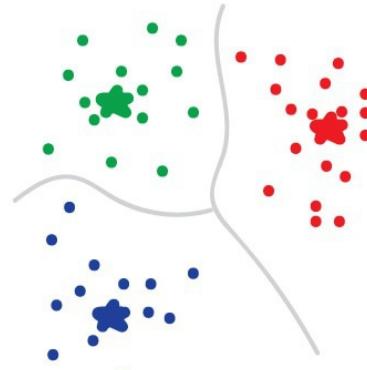
3. Двигаем ларьки ближе к центрам их популярности



4. Снова смотрим и двигаем



5. Повторяем много раз

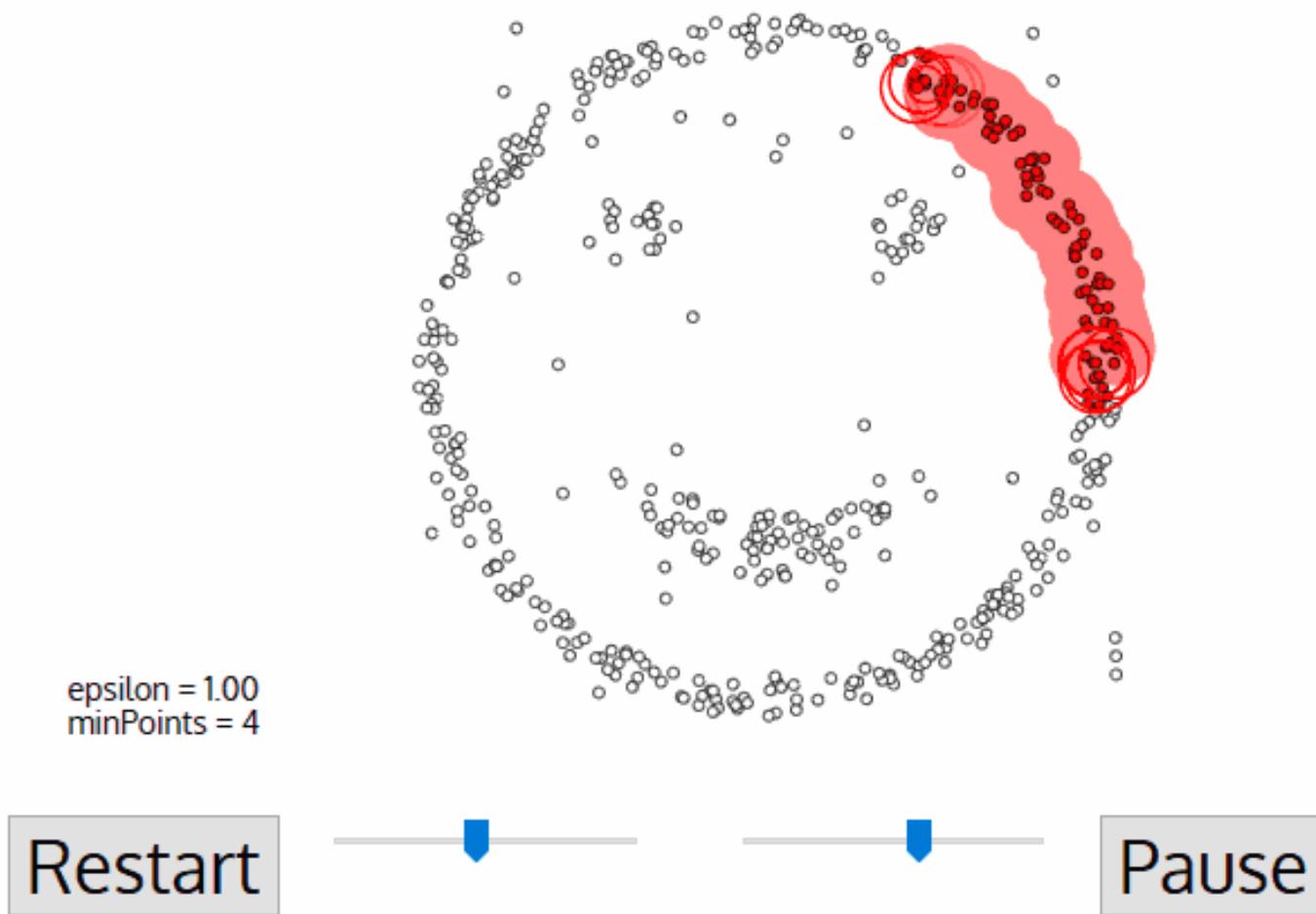


6. Готово, вы великолепны!

Искать центроиды удобно и просто, но в реальных задачах кластеры могут быть совсем не круглой формы. Вот вы геолог, которому нужно найти на карте схожие по структуре горные породы — ваши кластеры не только будут вложены друг в друга, но вы ещё и не знаете сколько их вообще получится.

Хитрым задачам — хитрые методы. DBSCAN, например. Он сам находит скопления точек и строит вокруг кластеры. Его легко понять, если представить, что точки — это люди на площади. Находим трёх любых близко стоящих человека и говорим им взяться за руки. Затем они начинают брать за руку тех, до кого могут дотянуться. Так по цепочке, пока никто больше не сможет взять кого-то за руку — это и будет

первый кластер. Повторяем, пока не поделим всех. Те, кому вообще некого брать за руку — это выбросы, аномалии. В динамике выглядит довольно красиво:



Интересующимся кластеризацией рекомендую статью [The 5 Clustering Algorithms Data Scientists Need to Know](#)

Как и классификация, кластеризация тоже может использоваться как детектор аномалий. Поведение пользователя после регистрации резко отличается от нормального? Заблокировать его и создать тикет саппорту, чтобы проверили бот это или нет. При этом нам даже не надо знать, что есть «нормальное поведение» — мы просто выгружаем все действия пользователей в модель, и пусть машина сама разбирается кто тут нормальный.

Работает такой подход, по сравнению с классификацией, не очень. Но за спрос не бьют, вдруг получится.

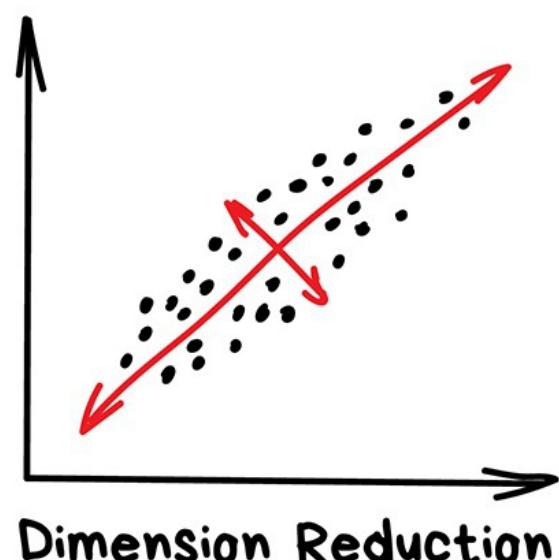
8 комментариев

Уменьшение Размерности (Обобщение)

«Собирает конкретные признаки в абстракции более высокого уровня»

Сегодня используют для:

- Рекомендательные Системы (★)
- Красивые визуализации
- Определение тематики и поиска похожих документов
- Анализ фейковых изображений
- Риск-менеджмент



Популярные алгоритмы: Метод главных компонент (PCA), Сингулярное разложение (SVD), Латентное размещение

Дирихле (LDA), Латентно-семантический анализ (LSA, pLSA, GLSA), t-SNE (для визуализации)

1 комментарий

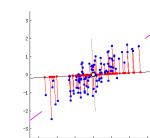
Изначально это были методы хардкорных Data Scientist'ов, которым сгружали две фуры цифр и говорили найти там что-нибудь интересное. Когда *просто строить графики в экселе* уже не помогало, они придумали напрячь машины искать закономерности вместо них. Так у них появились методы, которые назвали Dimension Reduction или Feature Learning.

Для нас практическая польза их методов в том, что мы можем объединить несколько признаков в один и получить абстракцию.

Например, собаки с треугольными ушами, длинными носами и большими хвостами соединяются в полезную абстракцию «овчарки».

Да, мы теряем информацию о конкретных овчарках, но новая абстракция всяко полезнее этих лишних деталей.

Плюс, обучение на меньшем количестве размерностей идёт сильно быстрее.



Проектируем 2D-данные на прямую (PCA)

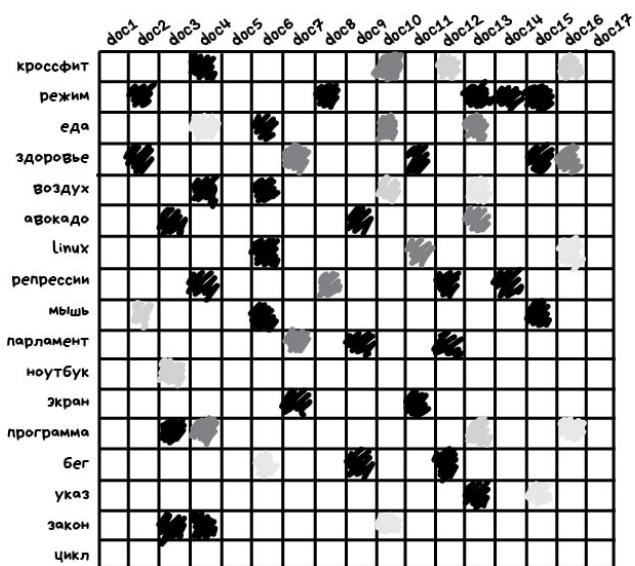
Инструмент на удивление хорошо подошел для определения

тематик текстов (Topic Modelling). Мы смогли абстрагироваться от конкретных слов до уровня смыслов даже без привлечения учителя со списком категорий. Алгоритм назвали Латентно-семантический анализ (LSA), и его идея была в том, что частота появления слова в тексте зависит от его тематики: в научных статьях больше технических терминов, в новостях о политике — имён политиков. Да, мы могли бы просто взять все слова из статей и кластеризовать, как мы делали с ларьками выше, но тогда мы бы потеряли все полезные связи между словами, например, что *батарейка* и *аккумулятор*, означают одно и то же в разных документах.

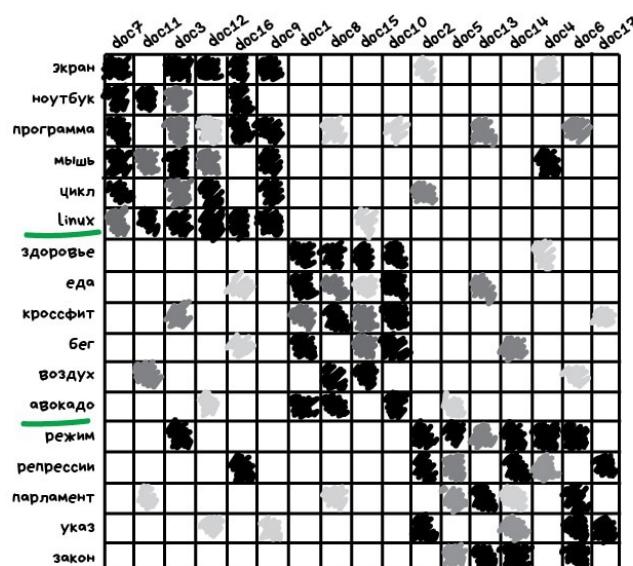
Точность такой системы — полное дно, даже не пытайтесь.

Нужно как-то объединить слова и документы в один признак, чтобы не терять эти скрытые (латентные) связи. Отсюда и появилось название метода. Оказалось, что Сингулярное разложение (SVD) легко справляется с этой задачей, выявляя для нас полезные тематические кластеры из слов, которые встречаются вместе.

Разделение документов по темам



→
SVD
2. Раскладываем



1. Строим матрицу как часто каждое слово встречается в каждом документе (чёрнее - чаще)

3. Получаем наглядные кластера по тематикам (даже если слова не встречались вместе)

Латентно-семантический Анализ (LSA)

Для понимания рекомендую статью [Как уменьшить количество измерений и извлечь из этого пользу](#), а практическое применение хорошо описано в статье [Алгоритм LSA для поиска похожих документов](#).

Другое мега-популярное применение метода уменьшения размерности нашли в рекомендательных системах и коллаборативной фильтрации (у меня был [пост про их виды](#)). Оказалось, если абстрагировать ими оценки пользователей фильмам, получается неплохая система рекомендаций кино, музыки, игр и чего угодно вообще.

Полученная абстракция будет с трудом понимаема мозгом, но когда исследователи начали пристально рассматривать новые признаки, они обнаружили, что какие-то из них явно коррелируют с возрастом пользователя (дети чаще играли в Майнкрафт и смотрели мультфильмы), другие с определёнными жанрами кино, а третьи вообще с

синдромом поиска глубокого смысла.

Машина, не знавшая ничего кроме оценок пользователей, смогла добраться до таких высоких материй, даже не понимая их. Достойно. Дальше можно проводить соцопросы и писать дипломные работы о том, почему бородатые мужики любят дегенеративные мультики.

На эту тему есть неплохая лекция Яндекса — [Как работают рекомендательные системы](#)

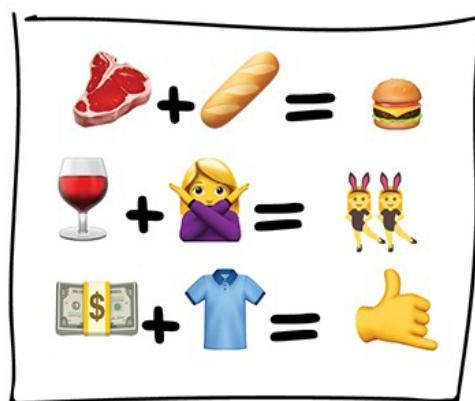
+ добавить комментарий

Поиск правил (ассоциация)

«Ищет закономерности в потоке заказов»

Сегодня используют для:

- Прогноз акций и распродаж
- Анализ товаров, покупаемых вместе
- Расстановка товаров на полках



Association
Rule Learning

- Анализ паттернов поведения на веб-сайтах

Популярные алгоритмы: Apriori, Euclat, FP-growth

3 комментария

Сюда входят все методы анализа продуктовых корзин, стратегий маркетинга и других последовательностей.

Предположим, покупатель берёт в дальнем углу магазина пиво и идёт на кассу. Стоит ли ставить на его пути орешки? Часто ли люди берут их вместе? Орешки с пивом, наверное да, но какие ещё товары покупают вместе? Когда вы владелец сети гипермаркетов, ответ для вас не всегда очевиден, но одно тактическое улучшение в расстановке товаров может принести хорошую прибыль.

То же касается интернет-магазинов, где задача еще интереснее — за каким товаром покупатель вернётся в следующий раз?

По непонятным мне причинам, поиск правил — самая хреново продуманная категория среди всех методов обучения. Классические способы заключаются в тупом переборе пар всех купленных товаров с помощью деревьев

или множеств. Сами алгоритмы работают наполовину — могут искать закономерности, но не умеют обобщать или воспроизводить их на новых примерах.

В реальности каждый крупный ритейлер пилит свой велосипед, и никаких особых прорывов в этой области я не встречал. Максимальный уровень технологий здесь — запилить систему рекомендаций, как в пункте выше. Хотя может я просто далёк от этой области, расскажите в комментариях, кто шарит?



9 комментариев

Часть 2. Обучение с

подкреплением

«Брось робота в лабиринт
и пусть ищет выход»

Сегодня используют для:

- Самоуправляемых автомобилей
- Роботов пылесосов
- Игр
- Автоматической торговли
- Управления ресурсами предприятий



Популярные алгоритмы: Q-Learning, SARSA, DQN, А3С,
Генетический Алгоритм

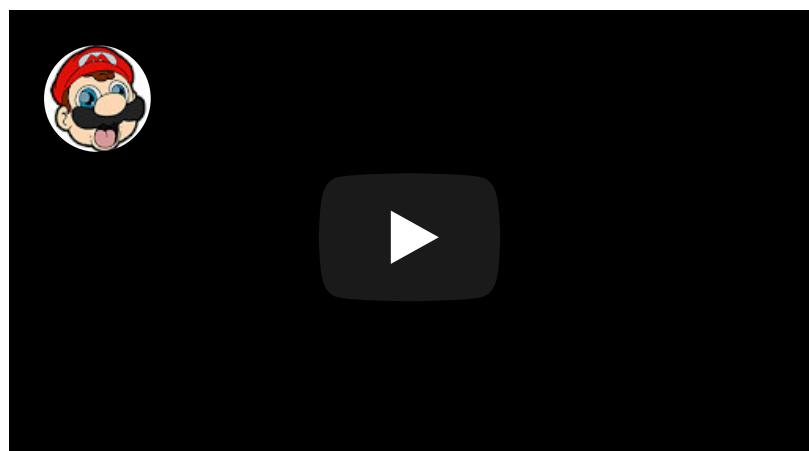
+ добавить комментарий

Наконец мы дошли до вещей, которые, вроде, выглядят как

настоящий искусственный интеллект. Многие авторы почему-то ставят обучение с подкреплением где-то между обучением с учителем и без, но я не понимаю чем они похожи. Названием?

Обучение с подкреплением используют там, где задачей стоит не анализ данных, а выживание в реальной среде.

Средой может быть даже видеоигра. Роботы, играющие в Марио, были популярны еще лет пять назад. Средой может быть реальный мир. Как пример — автопилот Теслы, который учится не сбивать пешеходов, или роботы-пылесосы, главная задача которых — напугать вашего кота до усрачки с максимальной эффективностью.



Нейросеть играет в Марио

Знания об окружающем мире такому роботу могут быть полезны, но чисто для справки. Не важно сколько данных он соберёт, у него всё равно не получится предусмотреть все ситуации. Потому его цель — **минимизировать ошибки, а не рассчитать все ходы.** Робот учится выживать в пространстве с максимальной выгодой: собранными монетками в Марио, временем поездки в Тесле или количеством убитых кожаных мешков хихих.

Выживание в среде и есть идея обучения с подкреплением. Давайте бросим бедного робота в реальную жизнь, будем штрафовать его за ошибки и награждать за правильные поступки. На людях норм работает, почему бы на и роботах не попробовать.

Умные модели роботов-пылесосов и самоуправляемые автомобили обучаются именно так: им создают виртуальный город (часто на основе карт настоящих городов), населяют случайными пешеходами и отправляют учиться никого там не убивать. Когда робот начинает хорошо себя чувствовать в искусственном GTA, его выпускают тестировать на реальные улицы.

Запоминать сам город машине не нужно — такой подход называется Model-Free. Конечно, тут есть и классический Model-Based, но в нём нашей машине пришлось бы запоминать модель всей планеты, всех возможных ситуаций на всех перекрёстках мира. Такое просто не работает. В обучении с подкреплением **машина не запоминает каждое движение, а пытается обобщить ситуации, чтобы выходить из них с максимальной выгодой**.

Как машины ведут себя при пожаре

Классическое программирование

«Я просчитал все варианты событий и ты сейчас должен связать верёвку из хлебного мякиша»

Машинное обучение

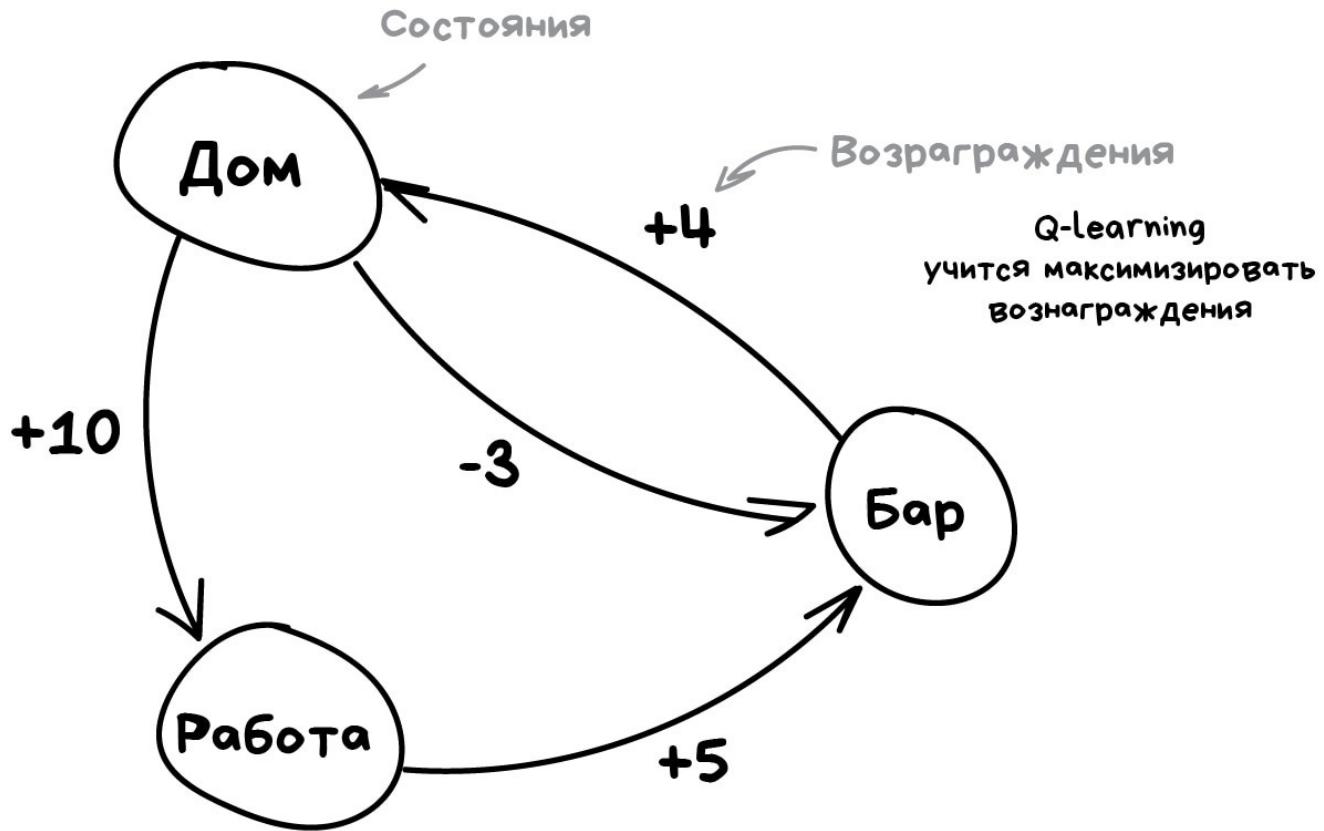
«По статистике люди гибнут в 6% пожаров, поэтому рекомендую вам умереть прямо сейчас»

Обучение с подкреплением

«Да просто беги от огня ААААААААА!!!! Бля бля бля»

Помните новость пару лет назад, когда машина обыграла человека в Го? Хотя незадолго до этого было доказано, что число комбинаций физически невозможно просчитать, ведь оно превышает количество атомов во вселенной. То есть если в шахматах машина реально просчитывала все будущие комбинации и побеждала, с Го так не прокатывало. Поэтому она просто выбирала наилучший выход из каждой ситуации и делала это достаточно точно, чтобы обыграть кожаного ублюдка.

Эта идея лежит в основе алгоритма Q-learning и его производных (SARSA и DQN). Буква Q в названии означает слово Quality, то есть робот учится поступать наиболее качественно в любой ситуации, а все ситуации он запоминает как простой марковский процесс.



Рутинный Марковский Процесс

Машина прогоняет миллионы симуляций в среде, запоминая все сложившиеся ситуации и выходы из них, которые принесли максимальное вознаграждение. Но как понять, когда у нас сложилась известная ситуация, а когда абсолютно новая? Вот самоуправляемый автомобиль стоит у перекрестка и загорается зелёный — значит можно ехать? А если справа мчит скорая помощь с мигалками?

Ответ — хрен знает, никак, магии не бывает, исследователи постоянно этим занимаются, изобретая свои костыли. Одни прописывают все ситуации руками, что позволяет им обрабатывать исключительные случаи типа проблемы вагонетки. Другие идут глубже и отдают эту работу нейросетям, пусть сами всё найдут. Так вместо Q-learning'a у нас появляется Deep Q-Network (DQN).

Reinforcement Learning для простого обывателя выглядит как настоящий интеллект. Потому что ух ты, машина сама принимает решения в реальных ситуациях! Он сейчас на хайпе, быстро прёт вперёд и активно пытается в нейросети, чтобы стать еще точнее (а не стукаться о ножку стула по двадцать раз).

Потому если вы любите наблюдать результаты своих трудов и хотите популярности — смело прыгайте в методы обучения с подкреплением (до чего ужасный русский термин, каждый раз передёргивает) и заводите канал на ютюбе! Даже я бы смотрел.

Помню, у меня в студенчестве были очень популярны генетические алгоритмы (по ссылке прикольная визуализация). Это когда мы бросаем кучу роботов в среду и заставляем их идти к цели, пока не сдохнут. Затем выбираем лучших, скрещиваем, добавляем мутации и бросаем еще раз. Через пару миллиардов лет должно получиться разумное существо. Теория эволюции в действии.

Так вот, генетические алгоритмы тоже относятся к обучению с подкреплением, и у них есть важнейшая особенность, подтвержденная многолетней практикой — они нахер никому не нужны.

Человечеству еще не удалось придумать задачу, где они были бы реально эффективнее других. Зато отлично заходят

как студенческие эксперименты и позволяют кадрить научруков «достижениями» особо не заморачиваясь. На ютюбе тоже зайдёт.

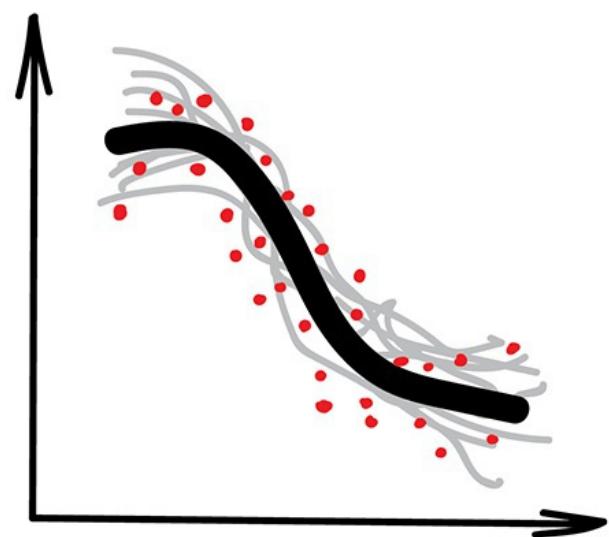
10 комментариев

Часть 3. Ансамбли

«Куча глупых деревьев
учится исправлять
ошибки друг друга»

Сегодня используют для:

- Всего, где подходят классические алгоритмы (но работают точнее)
- Поисковые системы (★)
- Компьютерное зрение
- Распознавание объектов



Ensemble Methods

+ добавить комментарий

Теперь к настоящим взрослым методам. Ансамбли и нейросети — наши главные бойцы на пути к неминуемой сингулярности. Сегодня они дают самые точные результаты и используются всеми крупными компаниями в продакшене. Только о нейросетях трещат на каждом углу, а слова «бустинг» и «бэггинг», наверное, пугают только хипстеров с теккранча.

При всей их эффективности, идея до издевательства проста. Оказывается, если взять несколько не очень эффективных методов обучения и обучить исправлять ошибки друг друга, качество такой системы будет аж сильно выше, чем каждого из методов по отдельности.

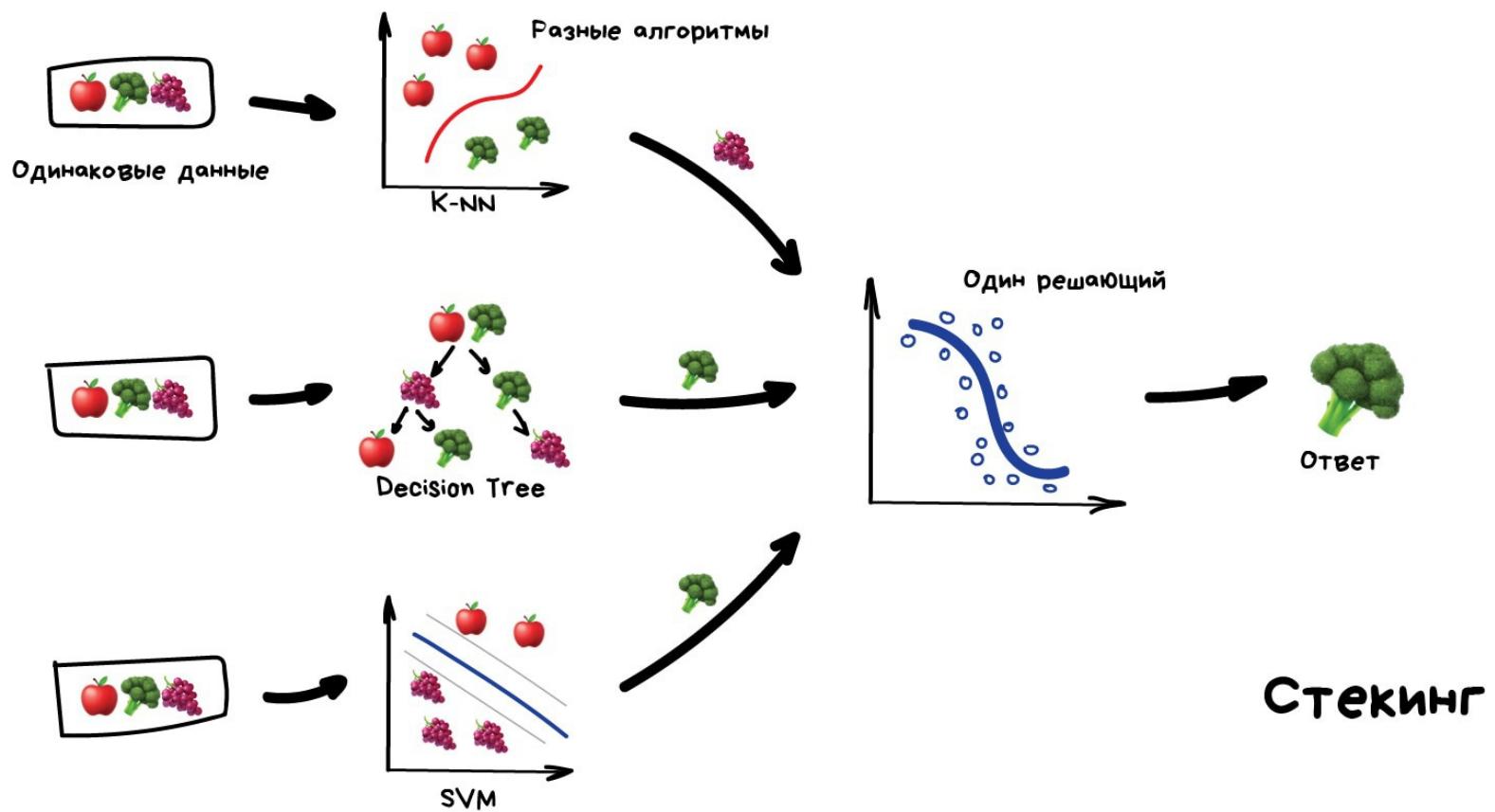
Причём даже лучше, когда взятые алгоритмы максимально нестабильны и сильно плавают от входных данных. Поэтому чаще берут Регрессию и Деревья Решений, которым достаточно одной сильной аномалии в данных, чтобы поехала вся модель. А вот Байеса и K-NN не берут никогда — они хоть и тупые, но очень стабильные.

Ансамбль можно собрать как угодно, хоть случайно нарезать

в тазик классификаторы и залить регрессией. За точность, правда, тогда никто не ручается. Потому есть три проверенных способа делать ансамбли.

Стекинг

Обучаем несколько разных алгоритмов и передаём их результаты на вход последнему, который принимает итоговое решение. Типа как девочки сначала опрашивают всех своих подружек, чтобы принять решение встречаться с парнем или нет.

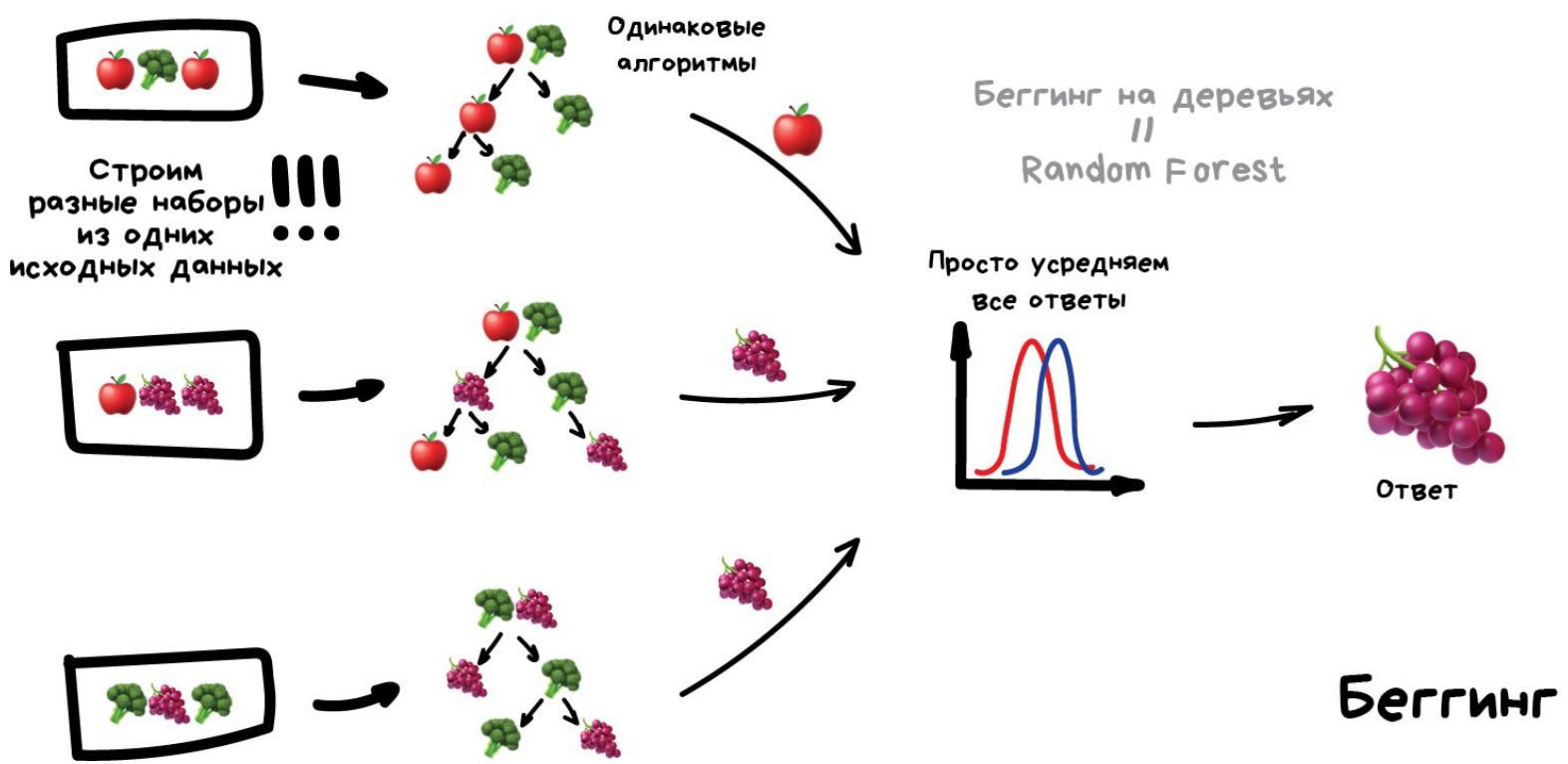


Ключевое слово — разных алгоритмов, ведь один и тот же алгоритм, обученный на одних и тех же данных не имеет смысла. Каких — ваше дело, разве что в качестве решающего алгоритма чаще берут регрессию.

Чисто из опыта — стекинг на практике применяется редко, потому что два других метода обычно точнее.

Беггинг Он же Bootstrap AGGREGATING. Обучаем один алгоритм много раз на случайных выборках из исходных данных. В самом конце усредняем ответы.

Данные в случайных выборках могут повторяться. То есть из набора 1-2-3 мы можем делать выборки 2-2-3, 1-2-2, 3-1-2 и так пока не надоест. На них мы обучаем один и тот же алгоритм несколько раз, а в конце вычисляем ответ простым голосованием.



Самый популярный пример баггинга — алгоритм Random Forest, баггинг на деревьях, который и нарисован на картинке. Когда вы открываете камеру на телефоне и видите как она очертила лица людей в кадре желтыми прямоугольниками — скорее всего это их работа. Нейросеть будет слишком медлительна в реальном времени, а баггинг идеален, ведь он может считать свои деревья параллельно

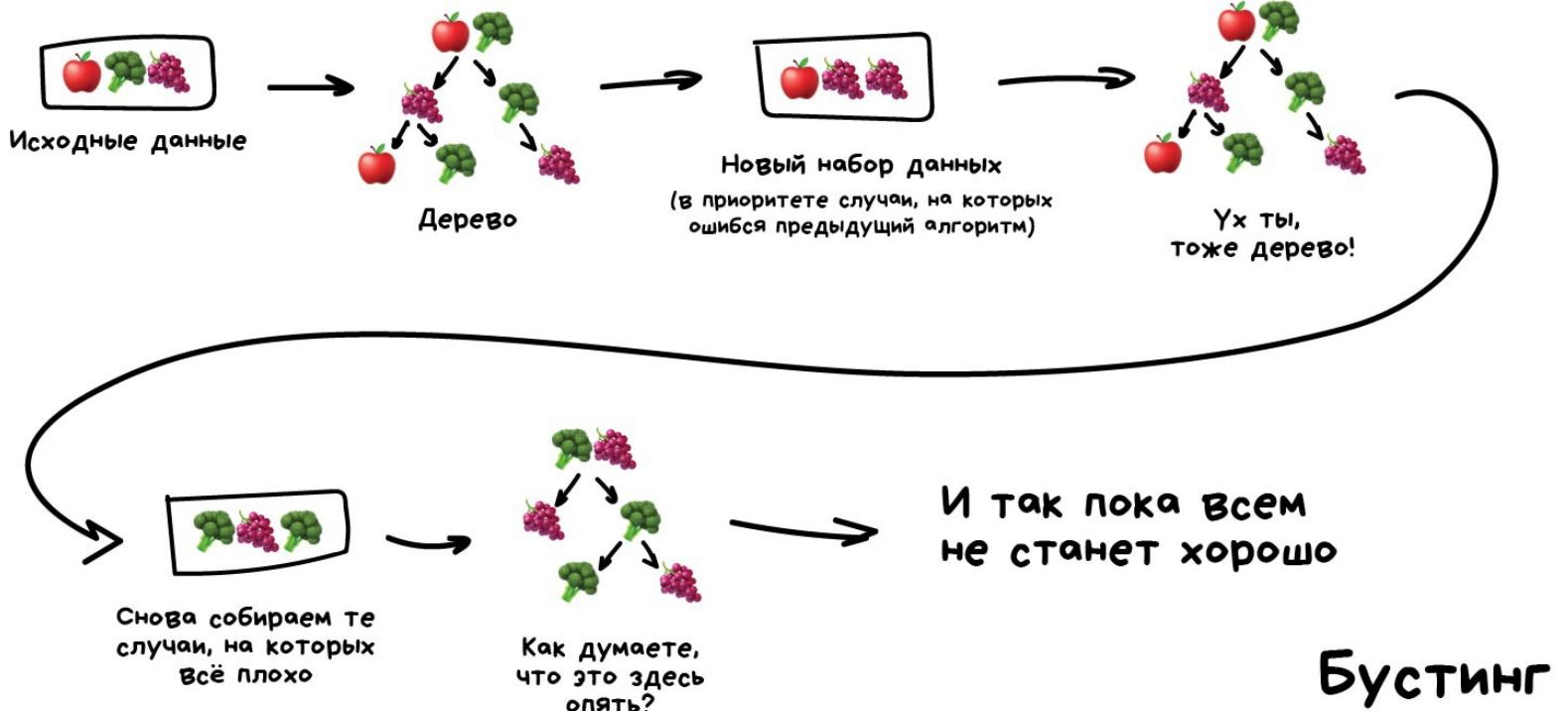
на всех шейдерах видеокарты.

Дикая способность параллелизиться даёт беггингу преимущество даже над следующим методом, который работает точнее, но только в один поток. Хотя можно разбить на сегменты, запустить несколько... ах кого я учу, сами не маленькие.



Бустинг Обучаем алгоритмы последовательно, каждый следующий уделяет особое внимание тем случаям, на которых ошибся предыдущий.

Как в беггинге, мы делаем выборки из исходных данных, но теперь не совсем случайно. В каждую новую выборку мы берём часть тех данных, на которых предыдущий алгоритм отработал неправильно. То есть как бы доучиваем новый алгоритм на ошибках предыдущего.



Плюсы — неистовая, даже нелегальная в некоторых странах, точность классификации, которой позавидуют все бабушки у подъезда. Минусы уже названы — не параллелизируется. Хотя всё равно работает быстрее нейросетей, которые как груженые камазы с песком по сравнению с шустрым бустингом.

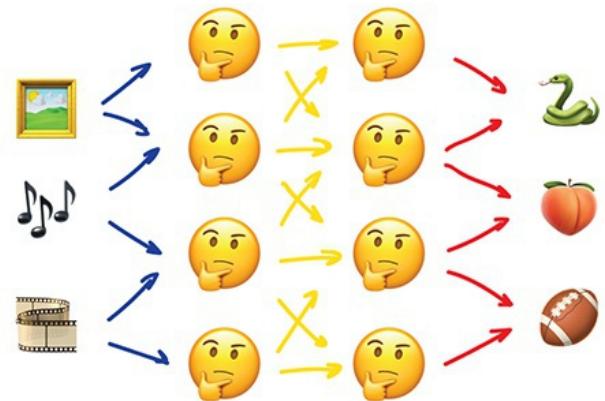
Нужен реальный пример работы бустинга — откройте Яндекс и введите запрос. Слышите, как Матрикснет грохочет деревьями и ранжирует вам результаты? Вот это как раз оно, Яндекс сейчас весь на бустинге. Про Google не знаю.

Сегодня есть три популярных метода бустинга, отличия которых хорошо донесены в статье [CatBoost vs. LightGBM vs. XGBoost](#)

Часть 4. Нейросети и глубокое обучение

«У нас есть сеть из тысячи слоёв, десятки видеокарт, но мы всё еще не придумали где это может быть полезно.

Пусть рисует котиков!»



Neural Networks

Сегодня используют для:

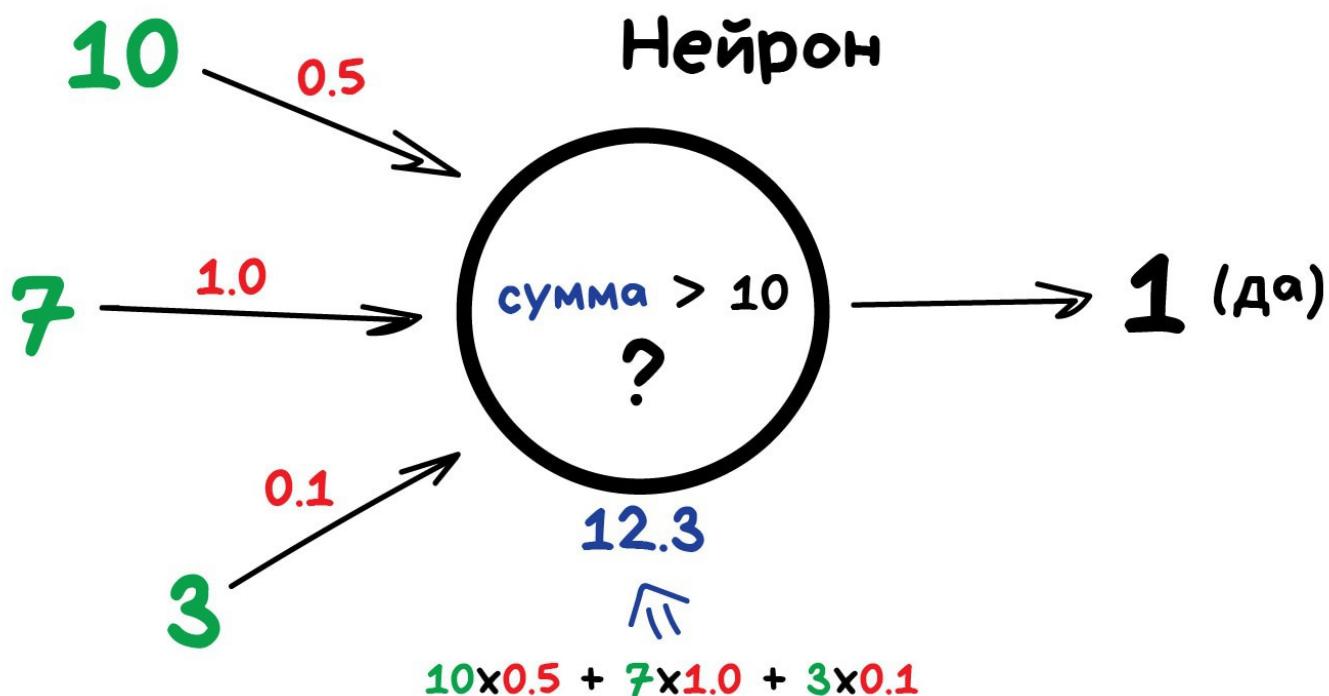
- Вместо всех вышеперечисленных алгоритмов вообще
- Определение объектов на фото и видео
- Распознавание и синтез речи
- Обработка изображений, перенос стиля
- Машинный перевод

Популярные архитектуры: Перцептрон, Свёрточные Сети (CNN), Рекуррентные Сети (RNN), Автоэнкодеры

Если вам хоть раз не пытались объяснить нейросеть на примере якобы работы мозга, расскажите, как вам удалось спрятаться? Я буду избегать этих аналогий и объясню как нравится мне.

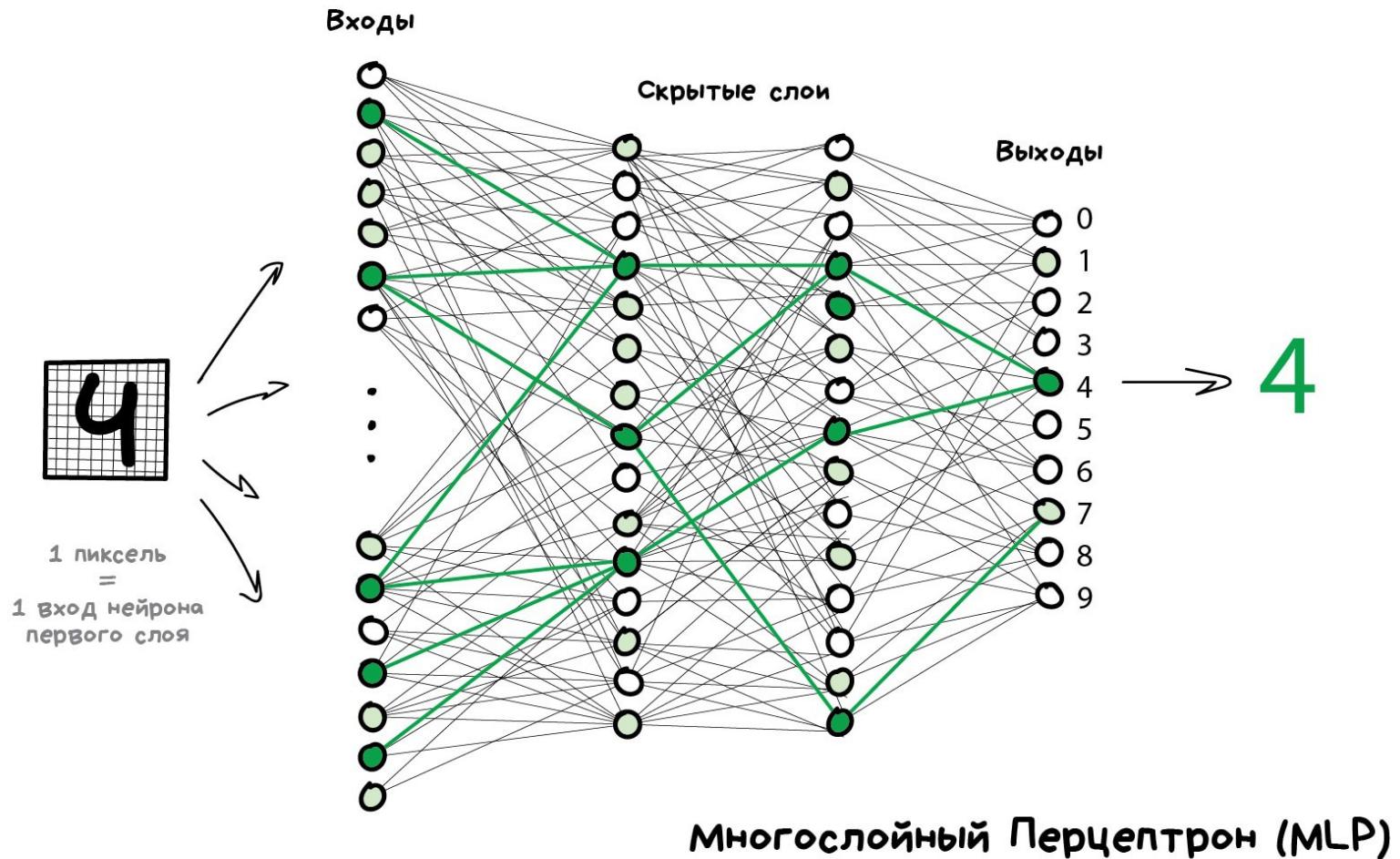
Любая нейросеть — это набор нейронов и связей между ними. Нейрон лучше всего представлять просто как функцию с кучей входов и одним выходом. Задача нейрона — взять числа со своих входов, выполнить над ними функцию и отдать результат на выход. Простой пример полезного нейрона: просуммировать все цифры со входов, и если их сумма больше N — выдать на выход единицу, иначе — ноль.

Связи — это каналы, через которые нейроны шлют друг другу циферки. У каждой связи есть свой вес — её единственный параметр, который можно условно представить как прочность связи. Когда через связь с весом 0.5 проходит число 10, оно превращается в 5. Сам нейрон не разбирается, что к нему пришло и суммирует всё подряд — вот веса и нужны, чтобы управлять на какие входы нейрон должен реагировать, а на какие нет.

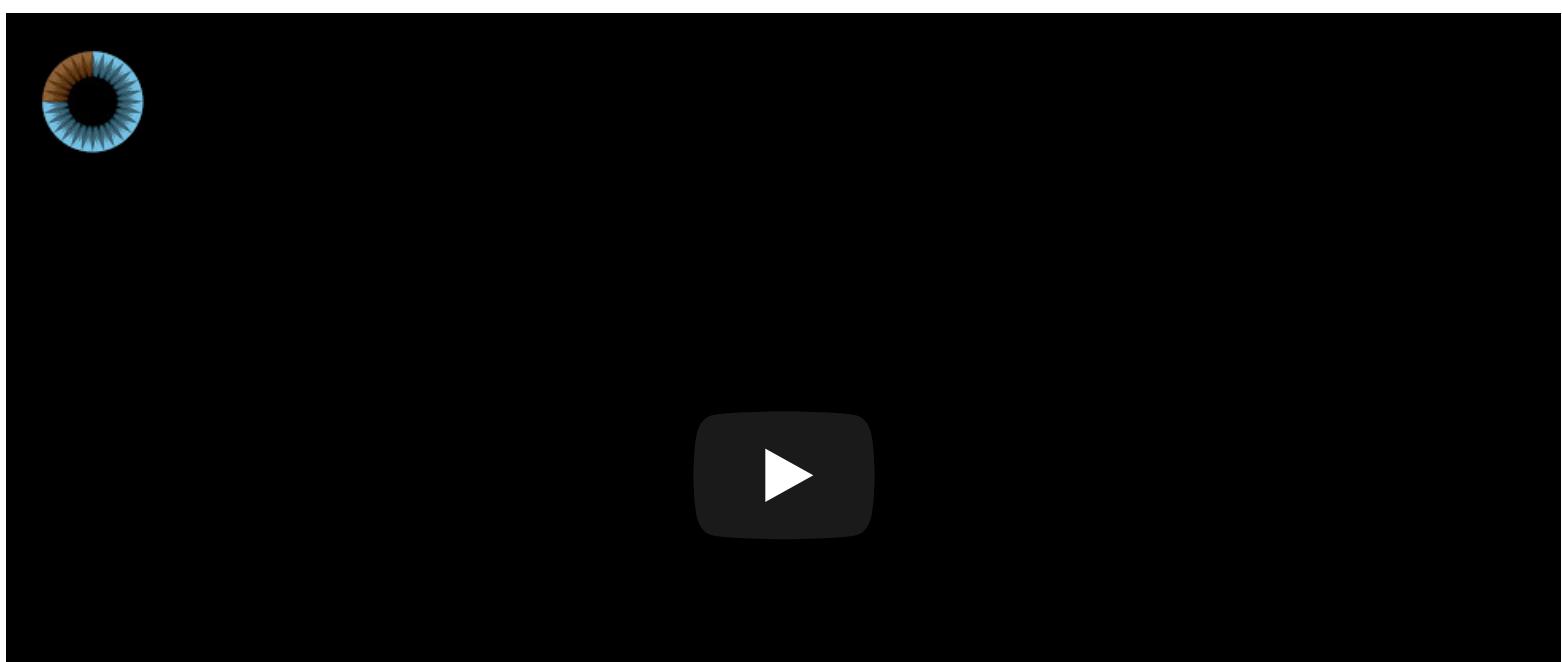


Чтобы сеть не превратилась в анархию, нейроны решили связывать не как захочется, а по слоям. Внутри одного слоя нейроны никак не связаны, но соединены с нейронами следующего и предыдущего слоя. Данные в такой сети идут строго в одном направлении — от входов первого слоя к выходам последнего.

Если нафигачить достаточно количество слоёв и правильно расставить веса в такой сети, получается следующее — подав на вход, скажем, изображение написанной от руки цифры 4, чёрные пиксели активируют связанные с ними нейроны, те активируют следующие слои, и так далее и далее, пока в итоге не загорится самый выход, отвечающий за четвёрку. Результат достигнут.



В реальном программировании, естественно, никаких нейронов и связей не пишут, всё представляют матрицами и считают матричными произведениями, потому что нужна скорость. У меня есть два любимых видео, в которых весь описанный мной процесс наглядно объяснён на примере распознавания рукописных цифр. Посмотрите, если хотите разобраться.



Такая сеть, где несколько слоёв и между ними связаны все нейроны, называется перцептроном (MLP) и считается самой простой архитектурой для новичков. В боевых задачах лично я никогда её не встречал.

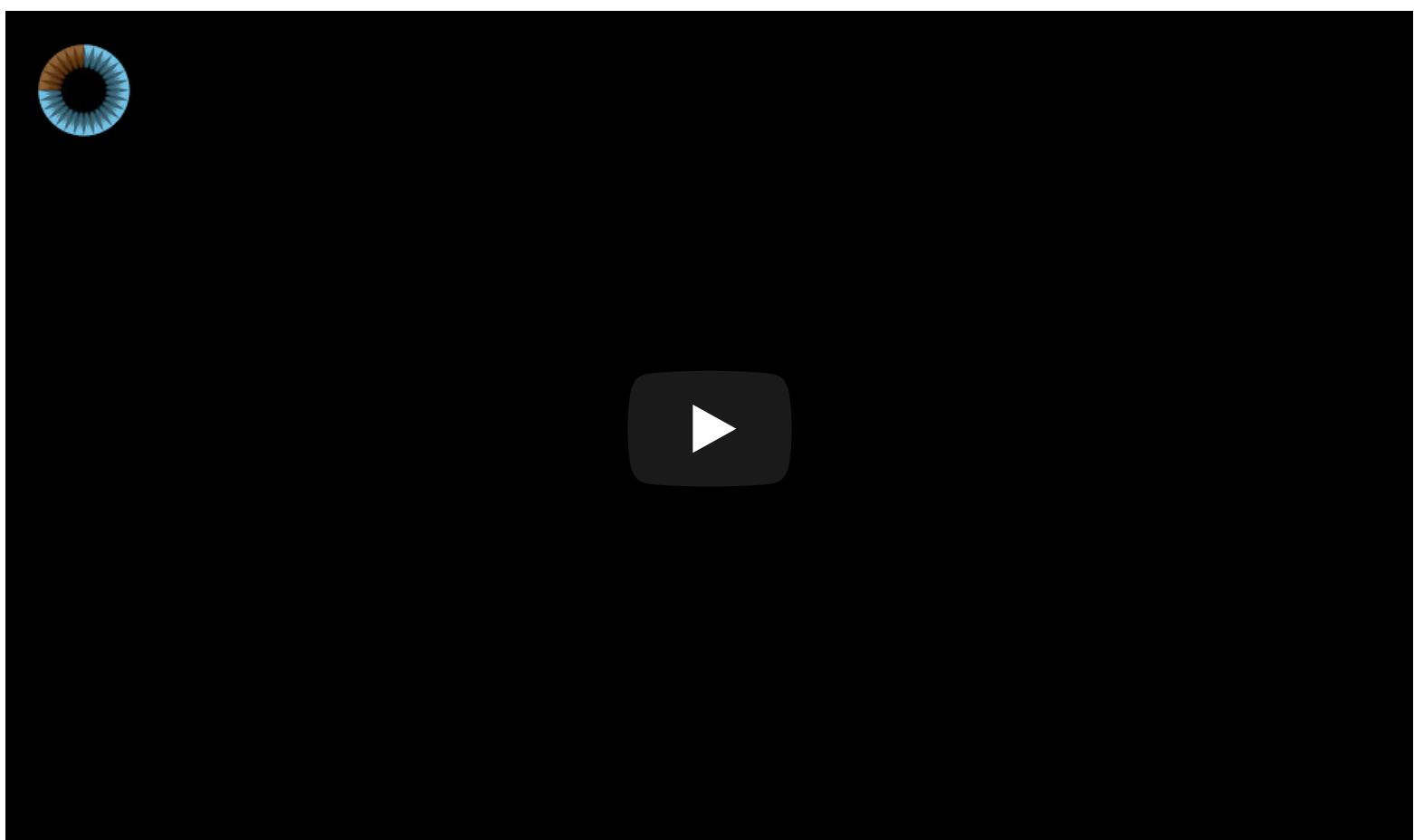
Когда мы построили сеть, наша задача правильно расставить веса, чтобы нейроны реагировали на нужные сигналы. Тут нужно вспомнить, что у нас же есть данные — примеры «входов» и правильных «выходов». Будем показывать нейросети рисунок той же цифры 4 и говорить «подстрой свои веса так, чтобы на твоём выходе при таком входе всегда загоралась четвёрка».

Сначала все веса просто расставлены случайно, мы показываем сети цифру, она выдаёт какой-то случайный ответ (весов-то нет), а мы сравниваем, насколько результат отличается от нужного нам. Затем идём по сети в обратном направлении, от выходов ко входам, и говорим каждому нейрону — так, ты вот тут зачем-то активировался, из-за тебя всё пошло не так, давай ты будешь чуть меньше реагировать на вот эту связь и чуть больше на вон ту, ок?

Через тысяч сто таких циклов «прогнали-проверили-наказали» есть надежда, что веса в сети откорректируются так, как мы хотели. Научно этот подход называется Backpropagation или «Метод обратного распространения»

ошибки». Забавно то, что чтобы открыть этот метод понадобилось двадцать лет. До него нейросети обучали как могли.

Второй мой любимый видос более подробно объясняет весь процесс, но всё так же просто, на пальцах.



Хорошо обученная нейросеть могла притворяться любым алгоритмом из этой статьи, а зачастую даже работать точнее. Такая универсальность сделала их дико популярными. *Наконец-то у нас есть архитектура человеческого мозга*, говорили они, *нужно просто собрать много слоёв и обучить их на любых данных*, надеялись они. Потом началась первая Зима ИИ, потом оттепель, потом вторая волна разочарования.

Оказалось, что на обучение сети с большим количеством слоёв требовались невозможные по тем временам мощности. Сейчас любое игровое ведро с жифорсами превышает мощность тогдашнего датацентра. Тогда даже надежды на это не было, и в нейросетях все сильно разочаровались.

Пока лет десять назад не бомбанул диплёрнинг.

На английской википедии есть страничка [Timeline of machine learning](#), где хорошо видны всплески радости и волны отчаяния.

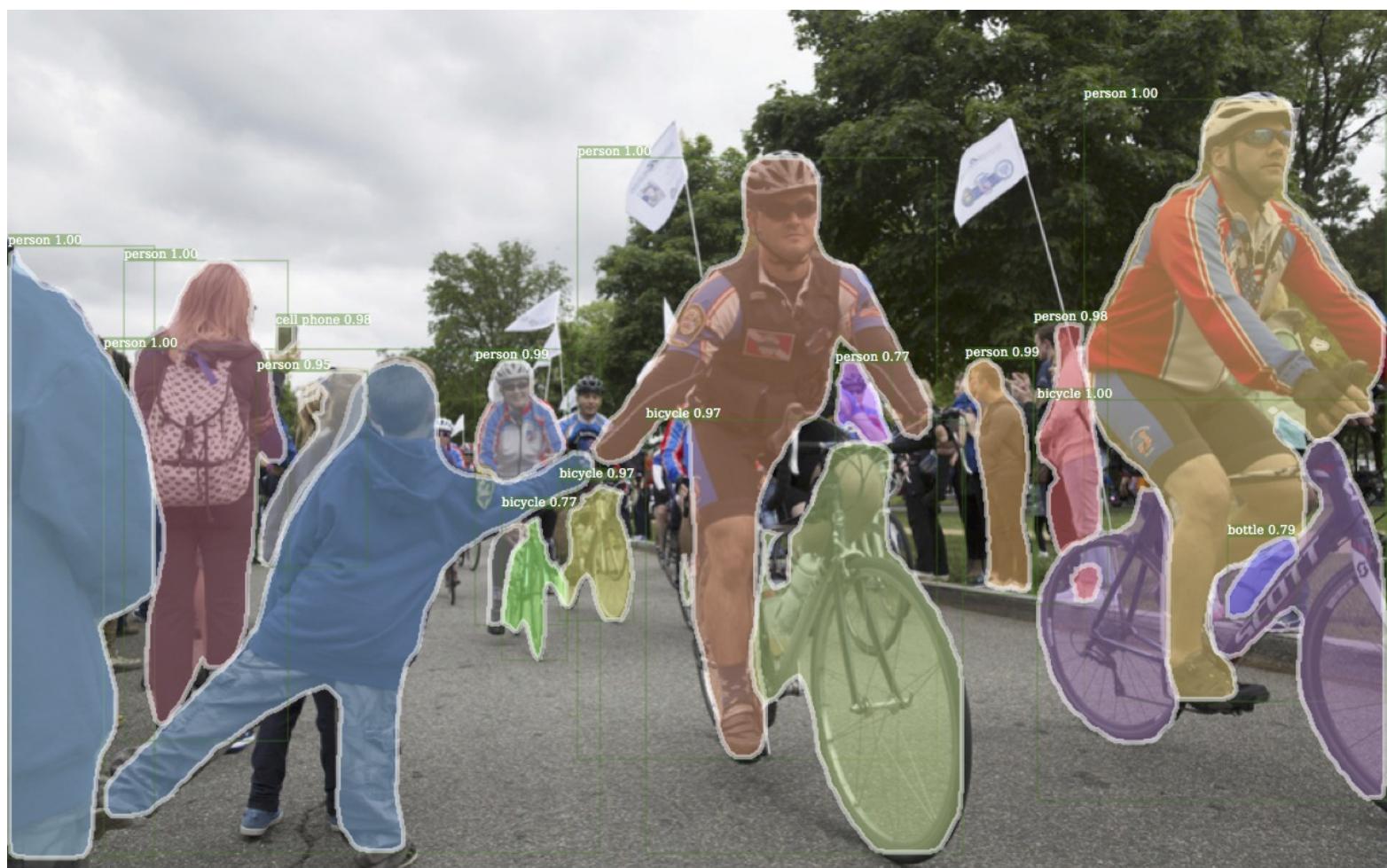
В 2012 году свёрточная нейросеть порвала всех в конкурсе ImageNet, из-за чего в мире внезапно вспомнили о методах глубокого обучения, описанных еще в 90-х годах. Теперь-то у нас есть видеокарты!

Отличие глубокого обучения от классических нейросетей было в новых методах обучения, которыеправлялись с большими размерами сетей. Однако сегодня лишь теоретики разделяют, какое обучение можно считать глубоким, а какое не очень. Мы же, как практики, используем популярные «глубокие» библиотеки типа [Keras](#), [TensorFlow](#) и [PyTorch](#) даже когда нам надо собрать мини-сетку на пять слоёв. Просто потому что они удобнее всего того, что было раньше. Мы называем это просто нейросетями.

Расскажу о двух главных на сегодняшний момент.

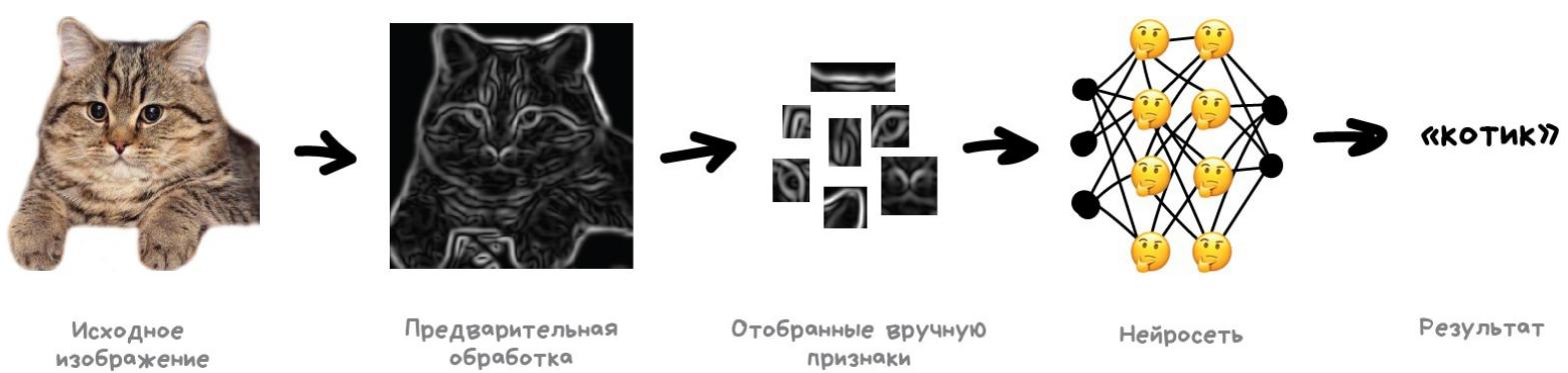
Свёрточные Нейросети (CNN)

Свёрточные сети сейчас на пике популярности. Они используются для поиска объектов на фото и видео, распознавания лиц, переноса стиля, генерации и дорисовки изображений, создания эффектов типа слоу-мо и улучшения качества фотографий. Сегодня CNN применяют везде, где есть картинки или видео. Даже в вашем айфоне несколько таких сетей смотрят на ваши голые фотографии, чтобы распознать объекты на них. Если там, конечно, есть что распознавать хехехе.



Картина выше — результат работы библиотеки [Detectron](#), которую Facebook недавно заопенсорсил

Проблема с изображениями всегда была в том, что непонятно, как выделять на них признаки. Текст можно разбить по предложениям, взять свойства слов из словарей. Картинки же приходилось размечать руками, объясняя машине, где у котика на фотографии ушки, а где хвост. Такой подход даже называли «handcrafting признаков» и раньше все так и делали.



Проблем у ручного крафтинга много.

Во-первых, если котик на фотографии прижал ушки или отвернулся — всё, нейросеть ничего не увидит.

Во-вторых, попробуйте сами сейчас назвать хотя бы десять характерных признаков, отличающих котиков от других животных. Я вот не смог. Однако когда ночью мимо меня пробегает чёрное пятно, даже краем глаза я могу сказать котик это или крыса. Потому что человек не смотрит только на форму ушей и количество лап — он оценивает объект по куче разных признаков, о которых сам даже не задумывается. А значит, не понимает и не может объяснить машине.

Получается, машине надо самой учиться искать эти признаки, составляя из каких-то базовых линий. Будем делать так: для начала разделим изображение на блоки 8x8 пикселей и выберем какая линия доминирует в каждом — горизонтальная [-], вертикальная [|] или одна из диагональных [/]. Могут и две, и три, так тоже бывает, мы не всегда точно уверены.

На выходе мы получим несколько массивов палочек, которые по сути являются простейшими признаками наличия очертаний объектов на картинке. По сути это тоже картинки, просто из палочек. Значит мы можем вновь выбрать блок 8x8 и посмотреть уже, как эти палочки сочетаются друг с другом. А потом еще и еще.

Такая операция называется свёрткой, откуда и пошло название метода. Свёртку можно представить как слой нейросети, ведь нейрон — абсолютно любая функция.



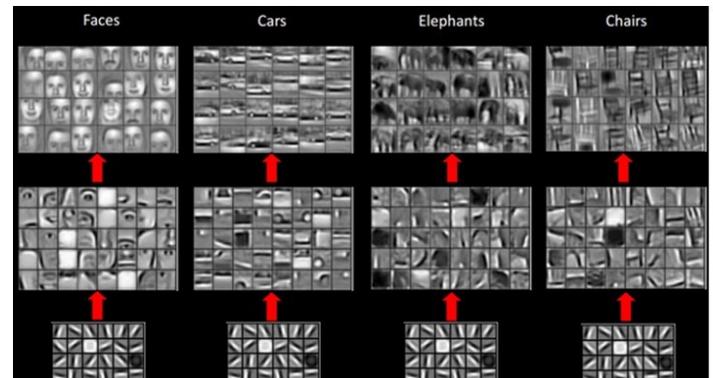
Свёрточная Нейросеть (CNN)

Когда мы прогоняем через нашу нейросеть кучу фотографий котов, она автоматически расставляет большие веса тем сочетаниям из палочек, которые увидела чаще всего. Причём неважно, это прямая линия спины или сложный геометрический объект типа мордочки — что-то обязательно будет ярко активироваться.

На выходе же мы поставим простой перцепtron, который будет смотреть какие сочетания активировались и говорить кому они больше характерны — кошке или собаке.

Красота идеи в том, что у нас получилась нейросеть, которая сама находит характерные признаки объектов. Нам больше не надо отбирать их руками.

Мы можем сколько угодно кормить её изображениями любых объектов, просто нагуглив миллион картинок с ними — сеть сама составит карты признаков из палочек и научится определять что угодно.



По этому поводу у меня даже есть несмешная шутка:

Дай нейросети рыбу — она сможет определять рыбу до конца жизни. Дай

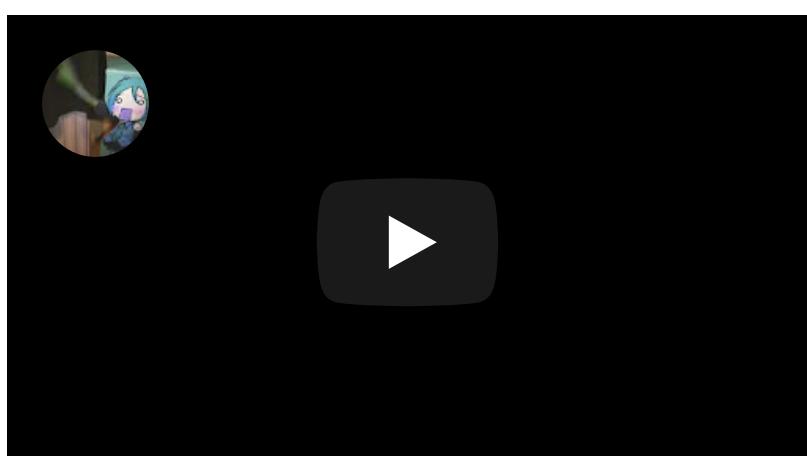
нейросети удочку — она сможет определять и удочку до конца жизни...

Рекуррентные Нейросети (RNN)

Вторая по популярности архитектура на сегодняшний день. Благодаря рекуррентным сетям у нас есть такие полезные вещи, как машинный перевод текстов ([читайте мой пост об этом](#)) и компьютерный синтез речи. На них решают все задачи, связанные с последовательностями — голосовые, текстовые или музыкальные.

Помните олдскульные голосовые синтезаторы типа Microsoft Sam из Windows XP, который смешно произносил слова по буквам, пытаясь как-то склеить их между собой? А теперь посмотрите на Amazon Alexa или Алису от Яндекса — они сегодня не просто произносят слова без ошибок, они даже расставляют акценты в предложении!

Потому что современные голосовые помощники обучают говорить не буквами, а фразами. Но сразу заставить нейросеть целиком выдавать фразы не выйдет, ведь тогда ей надо

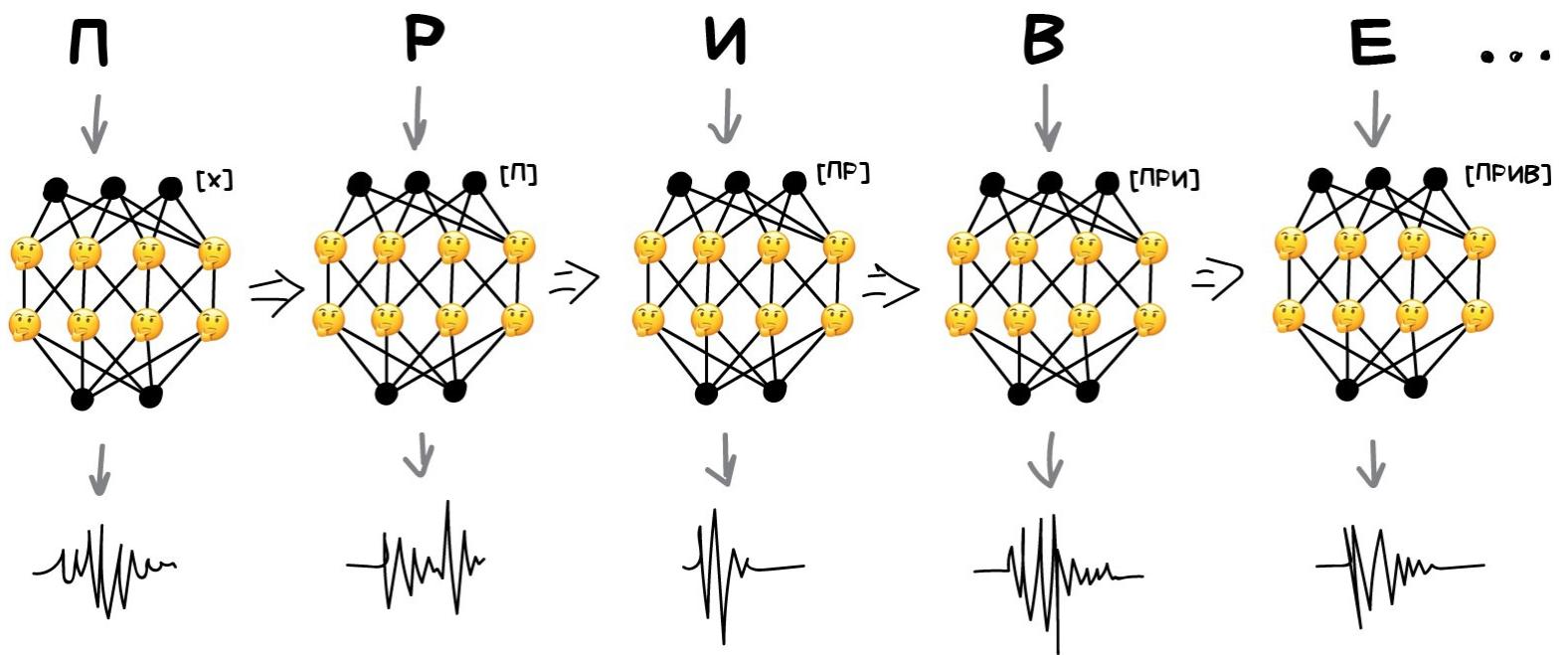


Нейросеть учится разговаривать

будет запомнить все фразы в языке и её размер будет исполинским. Тут на помощь приходит то, что текст, речь или музыка — это последовательности. Каждое слово или звук — как бы самостоятельная единица, но которая зависит от предыдущих. Когда эта связь теряется — получатся дабстеп.

Достаточно легко обучить сеть произносить отдельные слова или буквы. Берём кучу размеченных на слова аудиофайлов и обучаем по входному слову выдавать нам последовательность сигналов, похожих на его произношение. Сравниваем с оригиналом от диктора и пытаемся максимально приблизиться к идеалу. Для такого подойдёт даже перцепtron.

Вот только с последовательностью опять беда, ведь перцепtron не запоминает что он генерировал ранее. Для него каждый запуск как в первый раз. Появилась идея добавить к каждому нейрону память. Так были придуманы рекуррентные сети, в которых каждый нейрон запоминал все свои предыдущие ответы и при следующем запуске использовал их как дополнительный вход. То есть нейрон мог сказать самому себе в будущем — эй, чувак, следующий звук должен звучать повыше, у нас тут гласная была (очень упрощенный пример).



Рекуррентная Нейросеть (RNN)

Была лишь одна проблема — когда каждый нейрон запоминал все прошлые результаты, в сети образовалось такое дикое количество входов, что обучить такое количество связей становилось нереально.

Когда нейросеть не умеет забывать — её нельзя обучить (у людей та же фигня).

Сначала проблему решили в лоб — обрубили каждому нейрону память. Но потом придумали в качестве этой «памяти» использовать специальные ячейки, похожие на память компьютера или регистры процессора. Каждая ячейка позволяла записать в себя циферку, прочитать или сбросить — их назвали ячейки долгой и краткосрочной памяти (LSTM).

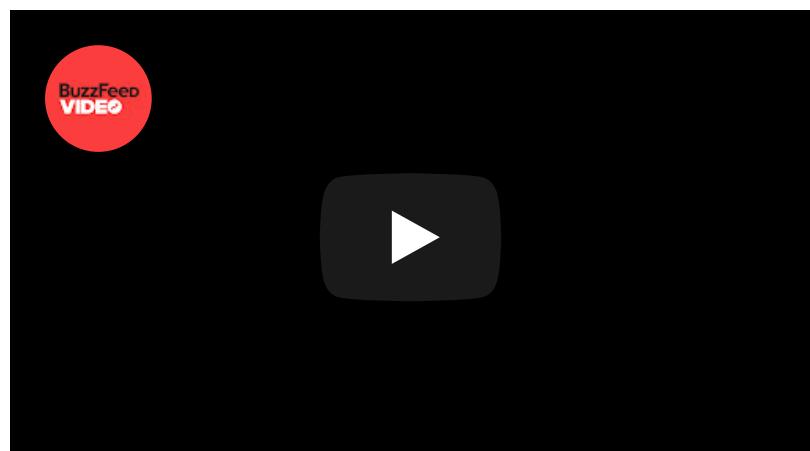
Когда нейрону было нужно поставить себе напоминалку на будущее — он писал это в ячейку, когда наоборот вся

история становилась ненужной (предложение, например, закончилось) — ячейки сбрасывались, оставляя только «долгосрочные» связи, как в классическом перцептроне. Другими словами, сеть обучалась не только устанавливать текущие связи, но и ставить напоминалки.

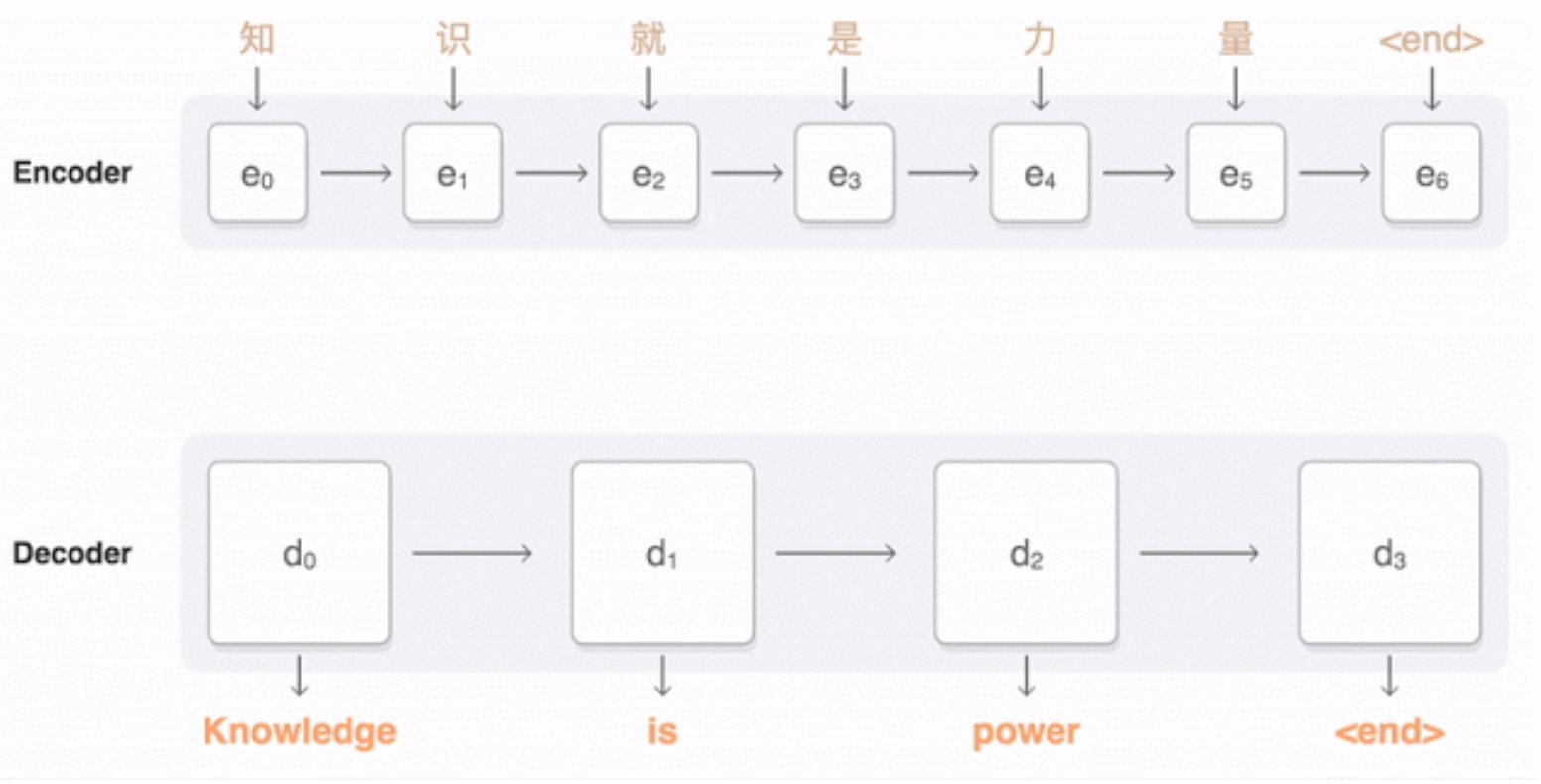
Просто, но работает!

Озвученные тексты для обучения начали брать откуда угодно. Даже базфид смог выгрузить видеозаписи выступлений Обамы и весьма неплохо научить нейросеть разговаривать его голосом.

На этом примере видно, что имитировать голос — достаточно простая задача для сегодняшних машин. С видео посложнее, но это пока.



CNN + RNN = фейковый Обама



Про архитектуры нейросетей можно говорить бесконечно.
Любознательных отправляю смотреть схему и читать [статью Neural Network Zoo](#), где собраны все типы нейронных сетей.
Есть и [русская версия](#).

A mostly complete chart of
Neural Networks

©2016 Fjodor van Veen - asimovinstitute.org

Backfed Input Cell

Input Cell

Noisy Input Cell

Hidden Cell

Probabilistic Hidden Cell

Spiking Hidden Cell

Output Cell

Match Input Output Cell

Recurrent Cell

Memory Cell

Different Memory Cell

Kernel

Convolution or Pool

Deep Feed Forward (DFF)

Perceptron (P)



Feed Forward (FF)



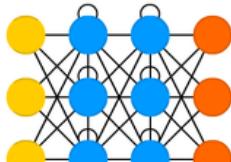
Radial Basis Network (RBF)



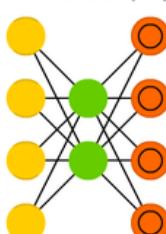
Recurrent Neural Network (RNN)

Long / Short Term Memory (LSTM)

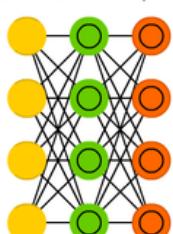
Gated Recurrent Unit (GRU)



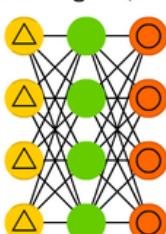
Auto Encoder (AE)



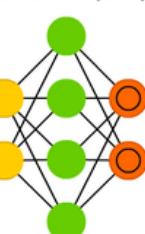
Variational AE (VAE)



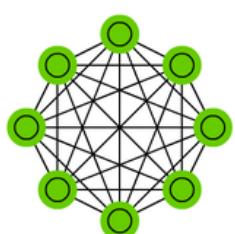
Denoising AE (DAE)



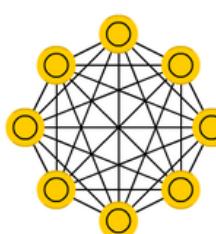
Sparse AE (SAE)



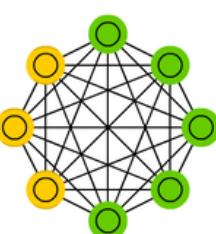
Markov Chain (MC)



Hopfield Network (HN)



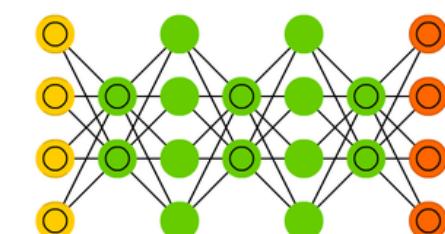
Boltzmann Machine (BM)



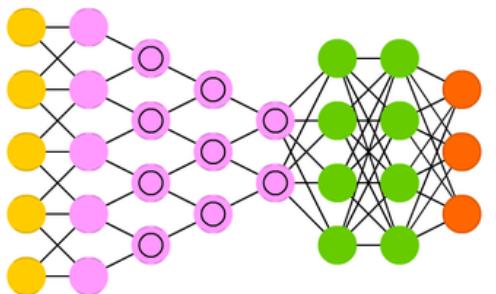
Restricted BM (RBM)



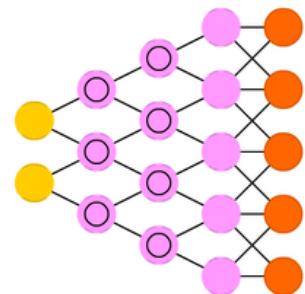
Deep Belief Network (DBN)



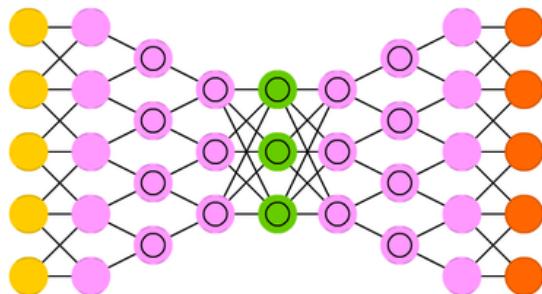
Deep Convolutional Network (DCN)



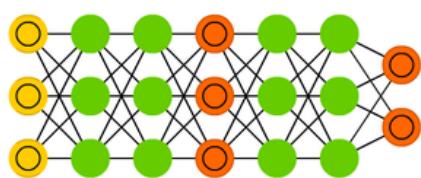
Deconvolutional Network (DN)



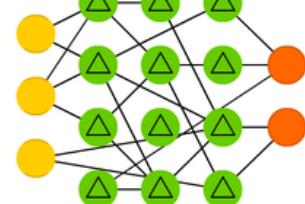
Deep Convolutional Inverse Graphics Network (DCIGN)



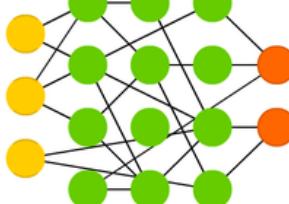
Generative Adversarial Network (GAN)



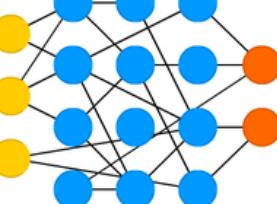
Liquid State Machine (LSM)



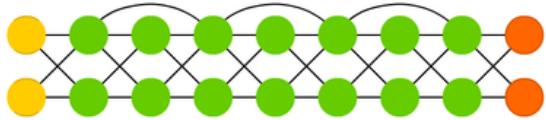
Extreme Learning Machine (ELM)



Echo State Network (ESN)



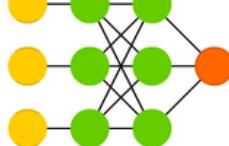
Deep Residual Network (DRN)



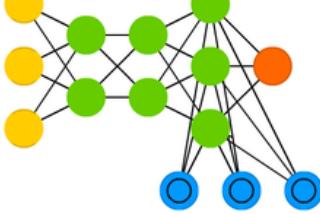
Kohonen Network (KN)



Support Vector Machine (SVM)



Neural Turing Machine (NTM)



Заключение: когда на войну с машинами?

На вопрос «*когда машины станут умнее нас и всех поработят?*» я всегда отвечаю, что он заранее неправильный. В нём слишком много скрытых условий, который примаются как данность.

Вот мы говорим «станут умнее нас». Значит мы подразумеваем, что **существует некая единая шкала интеллекта**, наверху которой находится человек, собаки пониже, а глупые голуби тусят в самом низу. Получается человек должен превосходить нижестоящих животных во всём, так? А в жизни не так. Средняя белка может помнить тысячу тайников в орешками, а я не могу вспомнить где ключи. Получается интеллект — это набор разных навыков, а не единая измеримая величина? Или просто запоминание орешков в него не входит? А убивание человеков входит?

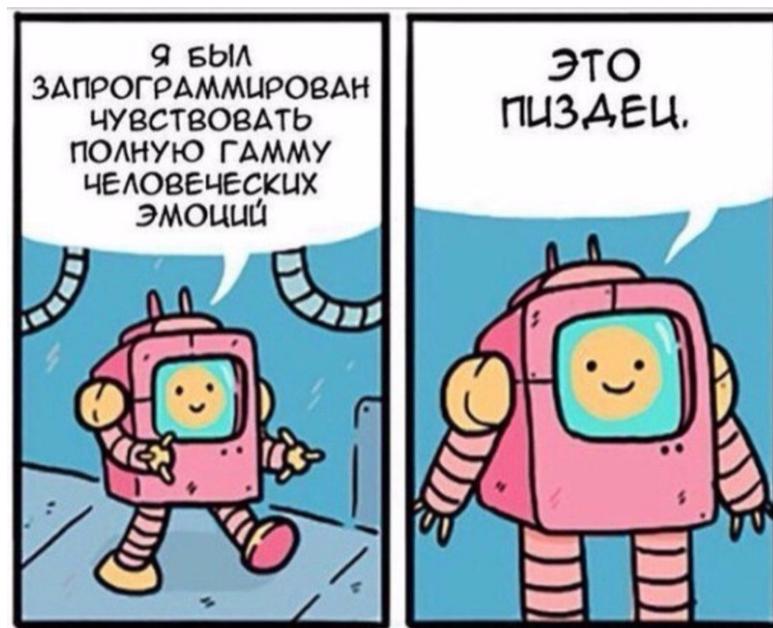
Ну и самый интересный для меня вопрос — **почему мы**

заранее считаем, что возможности человеческого мозга ограничены? В интернетах обожают рисовать графики, на которых технологический прогресс обозначен экспонентой, а возможности кожаных мешков константой. Но так ли это?

Вот давайте, прямо сейчас в уме умножьте 1680 на 950. Да, знаю, вы даже пытаться не станете. Но дай вам калькулятор, это займёт две секунды. Значит ли это, что вы только что расширили возможности своего мозга с помощью калькулятора? Можно ли продолжать их расширять другими машинами? Я вот использую заметки на телефоне — значит ли это, что я расширяю свою память с помощью машины?

Получается, мы уже успешно расширяем способности нашего мозга с помощью машин. Или нет?

Подумайте. У меня всё.



Проставить пивас
автору

Читайте Вастрик.Инсайд

Новые посты, заметки о технологиях и домашнем киберпанке.
Раз в две недели у вас на почте. Лучший способ подписки,
который не заблокирует Роскомнадзор.
Гляньте прошлые выпуски.

Эл. адрес:

Подписаться

Комментарии



ak3n :: 23 июля 2018, 10:00 :: Yekaterinburg, RU :: Windows :: лог

#

+46

Очень круто



neu :: 23 июля 2018, 12:36 :: Moscow, RU :: Windows :: лог

+44 Мля. это ж какой труд ты проделал



vas3k :: 23 июля 2018, 13:13 :: Moscow, RU :: Apple :: лог

+19 **neu**, ой, даже не спрашивай



Max :: 23 июля 2018, 15:22 :: UA :: Linux :: лог

+16 Чувак, снимаю шляпу!



dmash :: 23 июля 2018, 21:26 :: St Petersburg, RU :: Windows :: лог

+26 Это, пожалуй, самая понятная статья про нейросети, которую я видел



gerrk :: 24 июля 2018, 06:02 :: KZ :: Windows :: лог

+1 TL;DR потомушто я кожаный мешок



Мне повезет! :: 24 июля 2018, 08:15 :: Moscow, RU :: Apple :: лог

+9 Да, похоже работа сегодня будут работать себя сама
Спасибо!



Yur1 :: 24 июля 2018, 09:19 :: Syanno, BY :: Windows :: лог

+5 Класс, не зря я отказался от солёных т&т's в пользу поддержки тебя на патreonе. Снимаю воображаемую шляпу.



Deterge :: 24 июля 2018, 10:43 :: Krasnogorsk, RU :: :: лог #

+3 Спасибо за статью! Так все четко и понятно, очень зашла!



Onotole :: 24 июля 2018, 11:02 :: Willemstad, CW :: Apple :: лог #

+2 Брат, завтра получу зп — проставлю пивас, а пока что огромное спасибо за упорядочивание мусора, который был у меня в голове :)



Ton :: 24 июля 2018, 12:55 :: Dublin, IE :: Apple :: лог #

+4 "Кожаный ублюдок купил диван! Рекомендуем ему еще 100500 диванов"

Я так понимаю проблема в том что "умная" система знает лишь то, что я НАЧАЛ интересоваться диваном, но нет никакой обратной связи которая бы пояснила что диван уже куплен.

В итоге получается, что вместо того чтоб рекомендовать мне плед/подушку, кожаный мешок получает бесконечные вариации все тех же диванов.



DataAutist :: 24 июля 2018, 14:13 :: Rostov-on-Don, RU :: Apple :: лог #

лого

+9

Это лучшее, что я читал в своей жизни



alanin :: 24 июля 2018, 14:30 :: NL :: Linux :: лог #

+2 Спасибо, статья зашла на одном дыхании.



Кожаный :: 24 июля 2018, 16:05 :: Prague, CZ :: Apple :: лог #

+7 огромный респект автору! и плюсую за отличное чувство юмора



вротмненоги :: 24 июля 2018, 16:57 :: Moscow, RU :: Apple :: лог #

+6

Где картинки рисовал?



Вадик :: 24 июля 2018, 16:59 :: Lisbon, PT :: Linux :: лог #

+5

Апплодирую!



Аноним :: 24 июля 2018, 18:51 :: Moscow, RU :: Apple :: лог #

+3

Где ты работаешь ?



Georgy :: 24 июля 2018, 18:53 :: St Petersburg, RU :: Windows :: лог #

+2

Ура, Вастрик-жив!

Как всегда-великолепно - пивас!



Кот :: 24 июля 2018, 19:31 :: Naziya, RU :: Windows :: лог #

+5

Статья огонь! Продолжай!



mokogon :: 24 июля 2018, 19:36 :: Novosibirsk, RU :: Apple :: лог #

+1

Ясно-понятно!



St.me :: 24 июля 2018, 19:44 :: St Petersburg, RU :: Windows :: лог

#

+4

Спасибо огромное за статью - на одном дыхании, хотя от этой темы далека и никогда особо не интересовалась. Надеюсь, будет еще :) Очень вставочки порадовали "сын мамкиной подруги" и подобное, стиль зачетный :)



Lexing :: 24 июля 2018, 22:10 :: Krasnogorsk, RU :: Windows :: лог

#

+3

Чёрт, Вастрик, спасибо тебе.

Постоянно откладывал прочтение всей этой информации по кускам и ленился.

Лонгрид лучше очень многое, что я читал по этой тематике.

Лови пятишечный донатик.



МамкинХайпожор :: 24 июля 2018, 22:41 :: Moscow, RU :: Linux ::

лог

#

+3

Чувак, ты забыл самое мозговрывающее - GANы.

Статья супер!



Space Crusader :: 25 июля 2018, 03:59 :: KZ :: Windows :: лог

#

+3

Офигительная статья! Автор крут! Плакат жду, повешу на стену.



Anonymous :: 25 июля 2018, 08:02 :: London, GB :: Apple :: лог

#

+5

Эпичный пост по длине как режиссерская версия властелина колец

впервые решил проставить пивас автору



Del'ka :: 25 июля 2018, 11:12 :: Kharkiv, UA :: Windows :: лог

#

+3

Вастрик, добавь пожалуйста LTC-кошелёк для доната, у PayPal конская комиссия (больше 3\$) да и много у кого есть мелочь в крипте, думаю будет полезно.

Статья - огонь.



Некитос :: 25 июля 2018, 11:36 :: Samara, RU :: Linux :: лог

#

+1

Ну как бы да, когда мы пользуемся калькулятором или блокнотом - мы расширяем свой интеллект. Также как бульдозеры расширяют наши физ. возможности. Белка просто заточена на запоминание тайников, эволюцией. Но она не умнее человека. Вывод то какой?



Vitalever :: 25 июля 2018, 15:24 :: Moscow, RU :: Windows :: лог

#

+3

Восторг!

Пока читал думал:

В чём секрет объяснений, которые работают? Чем они отличаются от десятков и сотен других, не работающих? Простотой? Почему другие не используют эту простоту?

И можно ли эту концепцию простоты использовать (пусть хотя бы в качестве идеи, метафоры) в ML?

Если вообще дело в «простоте», а не в чём-то ещё...
Спасибо, автор!

P.S. Попытался простоять пиво, но перевод с карты тинькофф на тинькофф не прокатил. Какая-то странная ошибка. Попробую ещё попозже.



Александр :: 25 июля 2018, 15:26 :: :: Windows :: лог

+1

Возможно автор поймет и этих ребят -Джон Гриндер, Ричард Бэндлер.
Структура магии .
Я встретил лишь одного, кто понял, как это использовать в прикладной
психологии.
Если поймет - будет вторым



I Lay :: 25 июля 2018, 15:28 :: St Petersburg, RU :: Windows :: лог

+2

Поставил пивас автору!



Давид когда в навес :: 25 июля 2018, 23:46 :: Moscow, RU ::

Windows :: лог

+3

Статья шик, но ради христа подчисти опечатки, тысячи их



Некитос :: 26 июля 2018, 05:26 :: Samara, RU :: Linux :: лог

+1

Vitalever, если хорошо понимаешь, то и можно найти простое
объяснение. Если не можешь объяснить понятно, значит сам не
понимаешь.



gift :: 26 июля 2018, 06:29 :: Vilnius, LT :: Windows :: лог

+4

Отличная статья, спасибо!



Игорь :: 26 июля 2018, 12:27 :: St Petersburg, RU :: Windows :: лог

+4

Спасибо! Снимаю шляпу и простилаю пиво!



Йоу :: 26 июля 2018, 19:46 :: Yekaterinburg, RU :: Windows :: лог

#

+3 Статья просто супер. Спасибо большое! Лучшее, что читал по ML.
Максимально понятно, подробно и много ссылок для дальнейшего изучения! :)



Кондратий :: 26 июля 2018, 22:03 :: Perm, RU :: Apple :: лог

#

+2 Статья прекрасна. Но, черт возьми, где г-н Вастрик рисует такие картинки? В фотошопе? Как такому научиться?



Trototop :: 27 июля 2018, 04:30 :: Kemerovo, RU :: Windows :: лог

#

+3 Огромное спасибо за статью. Донат +



Андрюша :: 27 июля 2018, 15:05 :: Volgograd, RU :: Windows :: лог

#

+1 Привет. Очень крутая статья, есть один вопрос. Какой шрифт используешь в своих схемах? (например, "принять кредит" и прочее)?



Radishead :: 29 июля 2018, 07:23 :: Moscow, RU :: Linux :: лог

#

+2 Мешок с транзисторами, признавайся, какой алгоритм использовал, чтобы сгенерировать эту статью?



Александр :: 31 июля 2018, 08:51 :: RU :: Windows :: лог

#

+4 Это божественно! Спасибо за статью!



Bromansky :: 31 июля 2018, 11:03 :: Moscow, RU :: Windows :: лог #

+6

Ничего себе ты психанул. Очень круто! Даже мне как человеку в теме было интересно не торопясь пройтись по каждой части и насладиться разжеванным контентом, а также одновременно всплакнуть с мыслью, почему таких статей не было, когда я только начинал разбираться с ML.



Анна :: 31 июля 2018, 22:42 :: Voskresensk, RU :: Windows :: лог #

+2

Класс! Эта статья просто космос. Если бы мне кто сказал, что я буду про какие-то там алгоритмы читать не отрываясь несколько часов подряд, я бы ржала.

Вастрик, ты реально нереально круто рассказываешь.



FrozenFrame :: 02 августа 2018, 02:29 :: Kurtti, RU :: Windows :: лог #

+2

Восторг и восхищение!

Донат тебе за такой аццкий труд!



Rainbow-Spike :: 03 августа 2018, 15:42 :: RU :: Windows :: лог #

+2

Ох*ительно. Огромное спасибо. Комикс в конце статьи порадовал. Редкий случай, но я даже готов простить ему шрифт Comic Sans (всё равно никто не знает про Dat Fest Comic...). Когда-нибудь мой Комикслейт обзаведётся нейросетью, которая будет сама ставить забеливание, подбирать шрифты и переводить так, чтоб глаза не вытекали. На 68 языков сразу...



Людмила :: 05 августа 2018, 07:54 :: Uzhhorod, UA :: Windows :: лог #

+1

Прекрасный материал, спасибо. Казалось бы, где я и где машинное обучение :) Но у вас отличный стиль, чувство юмора и вообще вы крутой :)

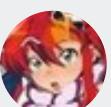


Igor Baliuk :: 06 августа 2018, 07:41 :: BY :: Windows :: лог

#

+0

Спасибо за качественный материал и подробное объяснение.



Dima :: 06 августа 2018, 16:08 :: Queanbeyan, AU :: Linux :: лог

#

+0

Как можно с помощью ИИ всех наебать так и не вычитал. К чему это всё, в офисе за з.п. сидеть?



AlexP :: 07 августа 2018, 05:57 :: :: Windows :: лог

#

+0

Добрый день!

Спасибо за прекрасный материал!

Подскажите кто то проходил "Машинное обучение и анализ данных" от Яндекс? Интересует порог входления в курс...ну и на сколько он практически полезен

Буду признателен за любую информацию



Oleg :: 11 августа 2018, 22:06 :: Moscow, RU :: Linux :: лог

#

+1

В статье очень много ошибок :(



AMD :: 14 августа 2018, 10:30 :: FR :: Windows :: лог

#

+1

Осилил за пару подходов, правда под конец мозг все равно начал вскипать и читал уже по диагонали) Очень классная статья, спасибо



Alex :: 14 августа 2018, 15:52 :: BY :: Windows :: лог

+1 Машина может создавать сама - текст, картинки, звуки и т.д. А так нереально круто.



Лёха :: 20 августа 2018, 13:54 :: St Petersburg, RU :: Windows :: лог

+0 Годнота



Виталий :: 04 сентября 2018, 13:42 :: Murmansk, RU :: Windows ::

лог

+0

Спасибо.



Ричард Фейнман :: 04 сентября 2018, 20:48 :: Innopolis, RU ::

Linux :: лог

+0

#респект, всё по полочкам, всё по красоте



Сергей Марков :: 27 сентября 2018, 02:14 :: Moscow, RU ::

Windows :: лог

+1

CNN это тоже MLP. CNN обычно являются частью GAN. Seq2seq это не автоэнкодер, это класс задач, которые обычно решаются при помощи рекуррентных сетей...

Ансамбли моделей могут включать в себя любые модели, в том числе, например, могут быть ансамбли нейронных сетей. Обучение с подкреплением — это не группа методов машинного обучения (с этой точки зрения виды МО это: обучение с учителем, обучение без учителя и обучение с частичным привлечением учителя; при этом обучение с подкреплением это разновидность обучения с учителем).



Алексей Рыжков :: 27 сентября 2018, 13:55 :: Vegas, CZ ::

#



Приветствую.

Статья отличная. Читать интересно и весело. Думаю такая задача и была. Да?

По сути где-то Сергей может формально и прав. Но для вводной статьи это допустимо, я думаю. Тут нужен интерес вместо академической точности формулировок. Кому надо - тот почитает Гудфеллоу и других "что, кто и зачем".

Самому тема интересна на практике. Поэтому поделюсь опытом и примером.

Вопрос про анализ совместных покупок. Тут все сложнее. Компании пилият велосипед, так как хотят сохранить конкурентное преимущество. Ведь там много вопросов. Обычно начинают с В1 и делают руками и что-то вроде KNN и деревьев простых, возможно.

Однако надо понимать - хорошая оценка возможна только по карточкам, так как у вас будет нормальная классификация как минимум по демографии и истории покупок. С историей покупок я бы думал о кластеризации и поиске скрытых зависимостей. Далее мы возвращаемся во временные ряды с анализом сезонностей, аномалий. Например выходные, праздники, 1 сентября, конец учебного года, дачный сезон по погоде каждый раз отличаться будет немного. Если изучать поведение конкретного человека, то есть примеры отличного моделирования и привязки к событиям. Правда не в России этот случай был.

Система сети супермаркетов определила предложила девушке в почте купоны на товары для беременных. Родители написали гневное письмо. Однако система нашла скрытый паттерн СМЕНЫ групп и цепочек покупок. И оказалась права. Человек (молодая девушка) резко перестал покупать сигареты и алкоголь, начала покупать другие крема, одежду, еду. Такая смена паттернов четко указывала на высокую вероятность такого события.

Безусловно это могло бы быть и заболевание. Но со значительно меньшей правдоподобностью.

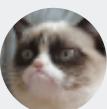
В Чехии сети это надо. У наших сетей может и так хватает денег и им это не надо.

не надо.

Удачи.

Еще вопрос автору - что ж у вас новые комменты внизу?

Я сначала подумал, что вообще не особо кому интересно, комменты очень старые. Это, правда, не помешало приятелям ссылку бросить, но ощущение....



ПРЕПОД ПО РУССКОМУ :: 01 октября 2018, 13:25 :: Rostov-on-

Don, RU :: Windows :: лог

+1

Статья классная, но автор пишет как двоечник. Подучил бы язык, прежде чем писать огромные пасты.. (Именно стилистика и логика текста жутко хромает)

Вот к примеру обзац, который надо бы отредактировать:

"Старый доклад Бобука про повышение конверсии лендингов с помощью SVM

Для классификации всегда нужен учитель — размеченные данные с признаками и категориями, которые машина будет учиться определять по этим признакам. Дальше классифицировать можно что угодно: пользователей по интересам — так делают алгоритмические ленты, статьи по языкам и тематикам — важно для поисковиков, музыку по жанрам — вспомните плейлисты Спотифая и Яндекс.Музыки, даже письма в вашем почтовом ящике."

Он реально очень раздражает)

А так конечно ,извините за критику. Содержание статьи бесспорно - ПИЗДАТО, нигде в инете такого нет. За это спасибо, а про редактирование текста я написал, что бы будущим людям было легче читать.

Спасибо и досвидания.

 **Михаил** :: 05 октября 2018, 14:18 :: RU :: Windows :: лог

#

+0 Отличная статья, столько полезного, что некоторым умникам на докторскую хватит. Спасибо автору за щедрость. Пивасик обязательно.

 **Сергей** :: 20 октября 2018, 23:46 :: St Petersburg, RU :: Windows ::

#

лог

+0

Вы великолепны!

 **вАхуе** :: 23 октября 2018, 14:37 :: Moscow, RU :: Apple :: лог

#

+0

ИдеальнO!

Спасибо за статью!

 **психолог** :: 26 октября 2018, 00:39 :: Nizhniy Novgorod, RU ::

#

Windows :: лог

+0

Офигенский текст понятный нематематику и непрограммисту.

Огромное спасибо

 **Валентин** :: 26 октября 2018, 12:11 :: Moscow, RU :: Windows :: лог

#

+0

Это топ, спасибо!

 **Ольга** :: 12 ноября 2018, 04:53 :: Moscow, RU :: Apple :: лог

#

+0

Огромное спасибо!!

 Яна :: 20 ноября 2018, 13:20 :: St Petersburg, RU :: Linux :: лог

#

+2

Я мама парня, который работает как раз в сфере глубокого обучения нейросетей. Чтоб мне не объяснять долго, скинул вашу статью. :)) Очень понравилось, все понятно и доступно! И весело, мило. Про ошибки не слушайте никого. Я вот сознательно иногда люблю слова подменять и буквы, чтоб придать большей выразительности! На вкусняшки отправила:)



 Лёша :: 21 ноября 2018, 11:19 :: Moscow, RU :: Windows :: лог

#

+0

Автор, я кожаный ублюдок-гуманитарий, но работаю в сфере ML. Спасибо тебе огромное за то, что мне теперь всё понятно и я больше не буду чувствовать себя дебилом на планёрках)



 Эльдар :: 11 декабря 2018, 06:19 :: Volgograd, RU :: Windows :: лог

#

+0

Статья классная, а иллюстрации вообще огонь! Все понятно, все ясно даже не особо просвященному человеку. Я обычно все новости про машинное обучение и искусственный интеллект беру из Вк https://vk.com/neural_nets или канала <https://t.me/aichannel> в телеграмме, но как же приятно находить еще полезные статьи по этой теме на других ресурсах.



 Рустам :: 12 декабря 2018, 15:10 :: FI :: Apple :: лог

#

+0

Замечательная статья! Можно тиражировать как учебник!



 svet :: 27 декабря 2018, 12:02 :: Novosibirsk, RU :: Apple :: лог

#

+0

Дочитала всё и в конце не поленилась перемножить в уме 1680 на 950. Получилось правильно :) Твой лонгрид отлично разогнал мозг!



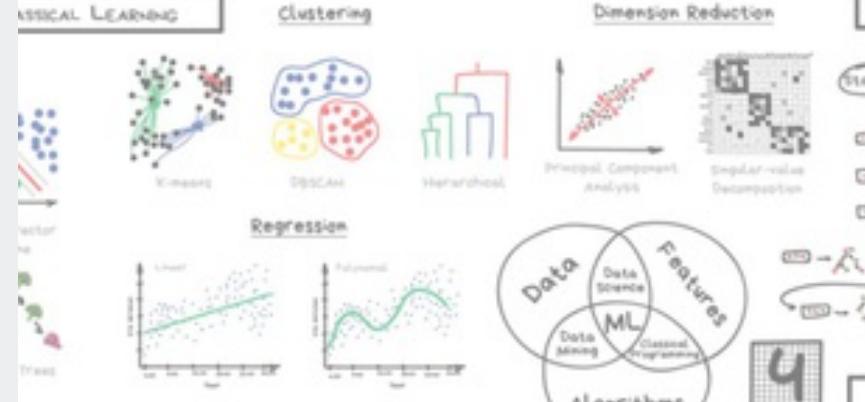
yartkin :: 07 января 2019, 00:13 :: Yekaterinburg, RU :: Apple :: лог

+0 Лайк, шер, респект!

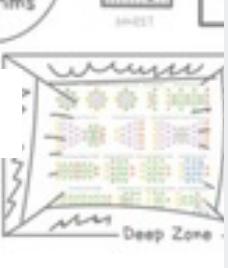
Автор:

Отправить

— 40 —



Машинное обучение для людей



Дом-дурачок 2.0



Телеграм-канал



Твиттер



Инстаграм



Патreon



me@vas3k.ru

