Лекция **«Введение в машинное обучение»**

Санкт-Петербург
2019

Содержание

1 Введение **2**

2 Задачи машинного обучения **4**
2.1 Модельный пример и случайные величины 4
2.2 А на чем и как учиться? 5
2.3 А не искусственный ли интеллект? 7
2.4 Карта мира машинного обучения 8

3 Классификация алгоритмов машинного обучения **9**
3.1 Классическое обучение: обучение с учителем 9
3.1.1 Задача регрессии 11
3.1.2 Задача классификации 13
3.2 Классическое обучение: обучение без учителя 14
3.2.1 Кластеризация 15
3.2.2 Ассоциации 16
3.2.3 Уменьшение размерности 17
3.3 Обучение с подкреплением 18
3.4 Ансамблевые методы 19
3.5 Совсем немного о нейросетях и глубоком обучении 20

4 Чуть-чуть необходимой общей статистики **21**

1 Введение

Добрый день, уважаемые слушатели! Добро пожаловать на курс «Введение в машинное обучение». Данный курс состоит из 5 лекций, в каждой из которых мы расскажем и наглядно покажем современные подходы к статистической обработке данных и построению моделей в машинном обучении (МО). Для усвоения предлагаемого материала будет достаточно элементарного курса высшей математики. Все требующиеся факты и понятия мы будем вводить (или напоминать) по мере необходимости. Ну что, приступим?

Для начала давайте все-таки разберемся, а что же такое машинное обучение и при чем тут какая-то статистика? Ведь современные СМИ просто переполнены такими соблазнительными и манящими вбросами, как: искусственный интеллект, машинное обучение, восстание машин, программисты программируют умных роботов, дата сайнтист, профессии будущего, и куча всякого такого. И эти сказки – один из самых популярных и общедоступных типов статей по сабжу. Второй тип статей, и их меньшинство, – это многостраничные томики, расписанные теоремами, узорчатыми формулами с непонятными значками, графиками и много-много чем. И что мы получаем?

От СМИ мы получаем всяческие обещания прекрасного будущего, передовой науки, полной автоматизации и вообще, о боже, чуть ли не создания искусственного разумного человека. А от второго типа статей, который, вроде как, обеспечивает эти блестящие перспективы, мы бежим, как от огня, ведь там почти ничего не понять и не разобрать. В итоге, сами что-то попробовать сделать мы не можем, и верим в то, что кто-то, может быть, все же сделает это для нас, а обещания превратятся в реальность. Да и вообще, просто верим.

Идея этого курса – пролить свет на основные задачи и методы машинного обучения. Как мы покажем, многие задачи машинного обучения – это не что-то из области фантастики. Это задачи, с которыми сталкивается каждый из нас даже просто-напросто в быту. В то же время, способы решения этих задач, конечно, основаны на математике, которую мы постараемся изложить в максимально понятной и доступной форме.

В этой лекции мы, во-первых, договоримся о терминологии и ответим на такие вопросы:

1. А зачем вообще обучать машины?
2. Из чего состоит обучение?
3. На чем и как учатся машины?
4. Где граничат и чем отличаются искусственный интеллект и машинное обучение?

5. И многие другие...

Кроме того, мы расскажем об основных направлениях обучения: с учителем, без учителя, с подкреплением, а также приведем основные примеры и классы задач, решаемых в рамках этих подходов.

Начиная что-то изучать, неплохо бы хоть немного понимать, что раньше: курица или яйцо, ну или хотя бы откуда ноги растут. Посему, начнем с небольшого экскурса в историю. Идея машинного обучения сама по себе не нова. Так в 1950 уже были известны простейшие алгоритмы, в 60-ых разработаны байесовские методы, но в 70-ых наступила так называемая зима искусственного интеллекта. Вторую жизнь МО стало приобретать с 90-ых годов прошлого века. В последние же годы, как и многое другое, машинное обучение претерпевает некоторое переосмысление. Это связано в первую очередь с ростом производительности компьютеров, когда современный бюджетный смартфон по характеристикам превосходит профессиональные компьютеры 15-ти летней давности.

В масштабах науки, МО – это одно из направлений искусственного интеллекта. Машинное обучение использует статистические методы, чтобы дать компьютерам возможность «учиться», а основная идея заключается в использовании предыдущего опыта для принятия решений в будущем. Так что на традиционный провокационный вопрос: нужна ли программисту математика, мы можем смело ответить – в данной области просто необходима.

Машинное обучение часто синонимично называют статистическим обучением. Основная задача МО состоит в обнаружении и формализации различных закономерностей или в создании некоего правила, основываясь на предложенных примерах. Именно благодаря такому подходу можно обнаружить сложные закономерности в данных (если они, конечно, есть), которые не видны или неочевидны человеку. А имея достаточно большой набор исходных данных становится возможным научить машину вести себя похожим образом, как и моделируемый объект.

Для обработки этого огромного количества данных и построения моделей используется аппарат математической статистики. Чтобы понять с какого рода задачами нам придется сталкиваться, и как мы их будем описывать математическим языком, разберем конкретный пример.

2 Задачи машинного обучения

2.1 Модельный пример и случайные величины

Давайте предположим, что мы хотим купить мобильный телефон, и прикидываем, какую сумму нужно накопить для его покупки. Посмотрев кучу объявлений в интернете, мы делаем вывод, что новый телефон подходящей марки с интересующими нас функциями стоит около 1000 долларов. При этом полугодовалый такой же телефон стоит уже 900 долларов, а годовалый – 800. Наверное, у каждого в голове напрашивается вывод: цена телефона каждые полгода падает на 100 долларов. Мы решили задачу, которую в машинном обучении называют задачей регрессии – задачу предсказания числа по некоторым входным данным.

Вообще, мы, люди, постоянно решаем в уме задачу регрессии. Мы оцениваем сколько стоит тот или иной автомобиль, взглянув на внешний вид, «внутренности» и узнав объем двигателя. Или задумываемся, а сколько продуктов нужно брать с собой на дачу на выходные, если собирается вся семья.

Ну что, ничего не смущает? Неужели есть формула, описывающая все на свете? Или все-таки во всех приводимых примерах есть очевидные изъяны? Конечно, есть. Ведь странно считать, что остаточная стоимость мобильного телефона зависит только от его возраста. А если у него, например, разбит экран, или не работает камера? Видимо, цена должна упасть еще больше, не так ли? А если телефон лежит в коробке, даже не распакованный, точно ли его цена упадет так же стремительно, как и цена телефона, который активно используют? Сомнительно, правда ведь? И, кстати, наша формула не учитывает никакие такие детали, а ведь их масса! Ну или, например, с автомобилями: ведь на реальную цену влияют наполнение конкретной комплектации, техническое состояние автомобиля, пробег, страна сборки, использования и многое-многое другое. И каков вывод?

А вывод таков: человек просто не может учесть все, ведь факторов, влияющих на то или иное явление – очень много. Есть и еще одна мысль, которая может перевернуть ваше сознание: иногда мы даже не знаем и не догадываемся о тех факторах, которые влияют на интересующее нас явление, а значит и тем более не можем их учесть. Тут-то на помощь и приходят машины. Конечно, с терминами типа «факторы» и «то или иное явление» далеко не уедешь, поэтому приведем еще один пример, на основе которого введем понятия, которыми и будем оперировать.

Предположим, что у нас есть данные о 100000 квартир в Санкт-Петербурге, причем мы знаем такие параметры, как: площадь квартиры, количество комнат, этаж, район, наличие парковки, расстояние до ближайшей станции метро и так далее. Кроме того, известна стоимость каждой кварти-

ры. Можем ли мы построить какую-то модель, которая будет предсказывать стоимость квартиры по заданным параметрам?

В рассматриваемом примере площадь квартиры, количество комнат, этаж и прочее – входные переменные. Их часто называют предикторами, независимыми переменными или просто переменными и обозначают $X_1, X_2, ..., X_p$. Конкретные их значения чаще обозначают маленькими буквами $x_1, x_2, ..., x_p$.

Величина на выходе – это цена квартиры, которая основывается на некоторых значениях предикторов. Ее часто называют откликом или зависимой переменной и обозначают Y, а ее значения маленькими буквами y.

Давайте поясним (как можно менее занудно) отличие больших и маленьких букв. Это отличие, кстати, мотивировано не только математикой. Ну, например, пусть X_1 – это площадь квартиры. И что вы понимаете из этого «например»? Площадь какой квартиры? Вашей, или квартиры друзей? Или сотни квартир из объявлений? Они же все отличаются! Значит, пока мы не говорим о конкретной квартире, о конкретном наблюдении, мы не можем сказать, чему в точности равно значение этой самой площади X_1. В математике X_1 принято называть случайной величиной. А вот при рассмотрении конкретного наблюдения, конкретной квартиры, мы уже можем сказать, что вот у этой-то квартиры площадь равна x_1. И это значение x_1 и есть значение случайной величины X_1 для конкретной квартиры. Понятно?

Аналогично мотивируется и ситуация с Y и y. Не зная конкретных значений случайных величин $X_1, X_2, ..., X_p$, конкретного значения отклика Y мы тоже не знаем, значит и Y – это случайная величина. Вот!

Ну хорошо, терминологию ввели, но что нам делать с этими ста тысячами объявлений с кучей параметров? Ни один человек не может охватить такой набор данных, выявить закономерности, отбросить лишнее. Конечно, как мы уже намекали, нужно заставить работать машины.

2.2 А на чем и как учиться?

Итак, для того, чтобы машина работала, то есть предсказывала результат по входным данным, ее нужно обучить. Чем разнообразнее обучающие данные, чем их больше, тем больше прецедентов наблюдает машина, а значит тем ей проще найти закономерности и, вроде как, тем точнее результат. Значит, во-первых, нам требуются данные.

Хотите предсказывать цену квартиры – отберите кучу объявлений о продаже. Хотите определять спам – дайте примеры спам-писем. Хотите предугадывать курс доллара – возьмите историю многолетней давности. Чем больше всяких разнообразных данных – тем лучше, ведь обучение основано на прецедентах, то есть на принципе: «если было так, а стало сяк, то, когда снова будет так, в результате будет сяк». Наивно? Возможно, но что вы еще хотите

сказать про будущее? Если данные точны, а условия одинаковы (кстати, это тоже данные), то и выхлоп должен быть одинаков. Это отсылка и к законам физики, и математики, и много к чему. Только мы все время натыкаемся на одну и ту же проблему – а все ли мы учитываем, и не учитываем ли что-то лишнее?

Как собирать данные, откуда они берутся? Да откуда угодно. Их можно собирать вручную, что, видимо, приводит к меньшему числу ошибок, а можно автоматически. Любимый пример – гугловская ReCaptcha, которая просит искать то светофоры, то автомобили, то дорожные знаки. Так вот, каждый раз, когда вы устало тыкаете по нужным квадратикам, вы передаете «экспертно оцененные» данные серверу, а потом эти данные формируют хороший датасет для обучения беспилотных автомобилей и проч. За качественными датасетами идет охота. В настоящее время известное высказывание Архимеда давно перефразировали: «Дайте мне качественные данные, и я переверну мир».

Из огромного набора данных нам требуется выделить так называемые признаки – это и есть те предикторы и отклики, на которые машина должна смотреть при обучении. Когда предикторов много, модель учится и работает медленнее, значит и менее эффективно. Иногда лишние предикторы вообще могут мешать работе модели. В прочем, есть и алгоритмы, помогающие отобрать наиболее интересные признаки из большого набора, мы к этому еще вернемся.

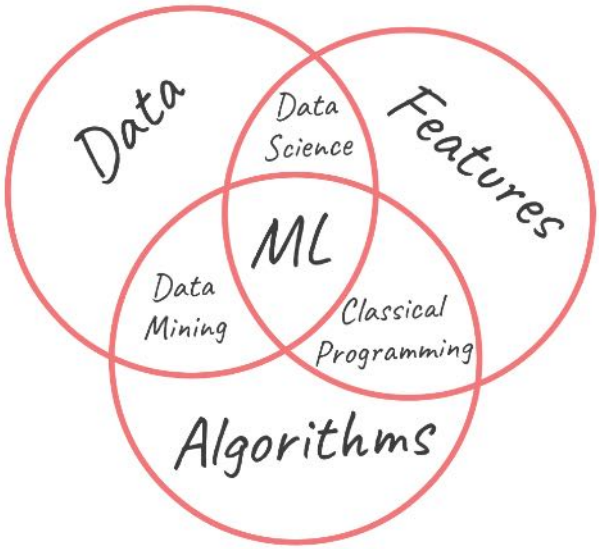

Рис. 1: Диаграммы терминов

Ну и последнее – это алгоритм. Одну и ту же задачу часто можно решать по-разному. От выбора метода решения будет зависеть и скорость работы модели, и точность ее предсказаний. Но есть одно важное НО! Если данные,

кхм, не очень, то ни один алгоритм делу не поможет.

В итоге все сказанное можно изобразить на такой вот картинке, которая показывает достаточно субъективное, но все же отличие машинного обучения от таких хайповых терминов, как: наука о данных (DataScience), интеллектуальный анализ данных (DataMining) и от стандартного программирования (ClassicalProgramming).

2.3 А не искусственный ли интеллект?

Давайте еще немного скажем о «множественных включениях», а именно ответим на вопрос: как соотносятся машинное обучение и искусственный интеллект? Часто вообще термины искусственный интеллект, машинное обучение, нейронные сети отождествляются и употребляются бессистемно. Между тем, это не одно и то же.

Искусственный интеллект (ИИ) появился в 1956, а его целью было, в общем-то, как и сейчас, заставить компьютер решать задачи, которые, как считается, подвластны в основном только людям, то есть те задачи, которые требуют интеллекта. Изначально исследователи работали над различными головоломками и задачей игры в шашки. Сейчас ИИ может относиться к чему угодно – от компьютерных программ для игры в те же шашки, до голосовых помощников, способных не только распознавать речь, но и отвечать на вопросы. Можно сказать, что искусственный интеллект относится к выводам компьютера: компьютер делает что-то «неглупое», а значит проявляет интеллект. Интеллект, в свою очередь, искусственный.

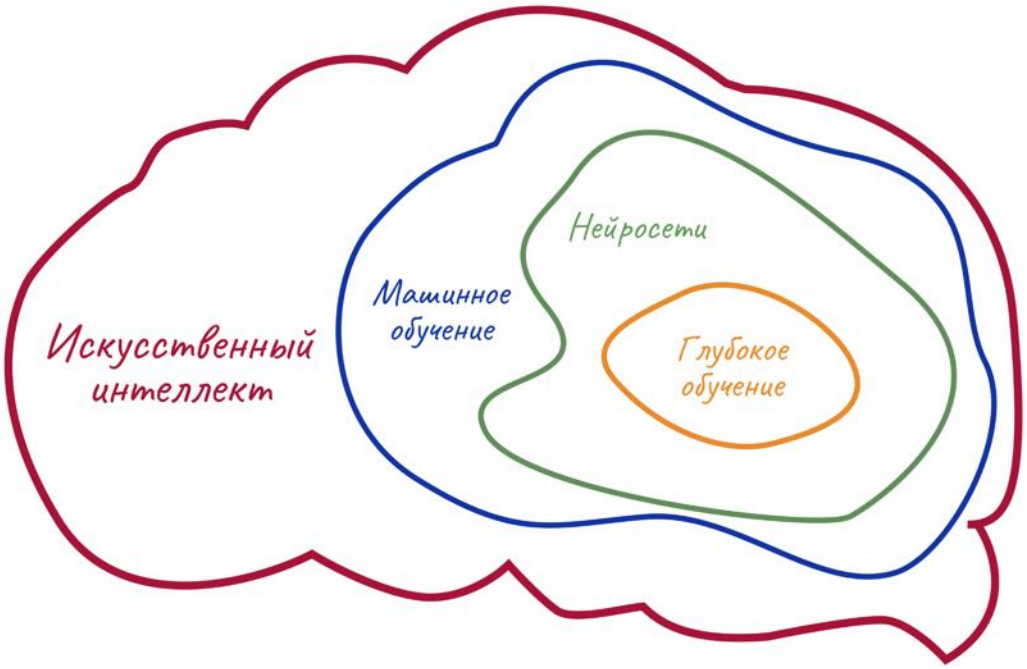

Рис. 2: МО, как часть ИИ

Машинное обучение – это одно из направлений искусственного интеллекта. Основной принцип МО – обучение на заранее данном наборе данных. В прочем, об этом мы уже говорили в примерах.

Глубокое обучение – это, так сказать, подмножество машинного обучение. Оно использует аппарат нейронных сетей, имитирующих человеческое мышление и, как следствие, принятие решений, для решения реальных задач. Мы не будем касаться глубокого обучения в нашем курсе.

Итак, общая схема, соответствующая описанному, представлена на рисунке 2.

Ну и давайте раз и навсегда, достаточно грубо скажем: а что может на сегодняшний день обученная машина? Итак, она может: строить предсказание, запоминать новое, воспроизводить имеющееся, выбирать лучшее. А чего не может? Уж точно не может выйти за рамки задачи, создать новое и поработить весь мир.

2.4 Карта мира машинного обучения

Посмотрите внимательно на рисунок. В нем каким-то образом классифицированы те или иные алгоритмы, относящиеся к машинному обучению. Способов классификации – масса, и здесь представлен всего один из множества возможных. Основные ветки машинного обучения – это классическое обучение, делящееся на обучения с учителем и без, обучение с подкреплением, ансамблевые методы и нейронные сети с глубоким обучением.

Давайте теперь поговорим чуть подробнее про наш курс. Мы, в основном, будем заниматься обучением с учителем. Во второй лекции мы подробно разберем задачу регрессии: линейную регрессию, многомерную линейную регрессию, полиномиальную регрессию, а также скажем пару слов про гребневую или ридж-регрессию и метод регрессии LASSO. В третьей лекции мы затронем задачу классификации: обсудим классификатор на основе логистической регрессии, а также сравним его с линейной регрессией. В четвертой лекции мы продолжим заниматься классификацией и рассмотрим наивный байесовский классификатор, а также метод k ближайших соседей, поговорим про проклятие размерностей. Пятая же лекция перенесет нас в раздел обучения без учителя – к задаче кластеризации: мы рассмотрим алгоритмы K-средних и агломеративную (иерархическую) кластеризацию.

Для изучения алгоритмов уменьшения размерности, ансамблевых методов и обучения с подкреплением, мы приглашаем вас в курс, являющийся продолжением данного – «Advanced Machine Learning».

Ну что же, во введении, то есть в данной лекции, мы кратко пройдемся по каждой нарисованной ветке и расскажем про задачи, которые решаются теми или иными методами.

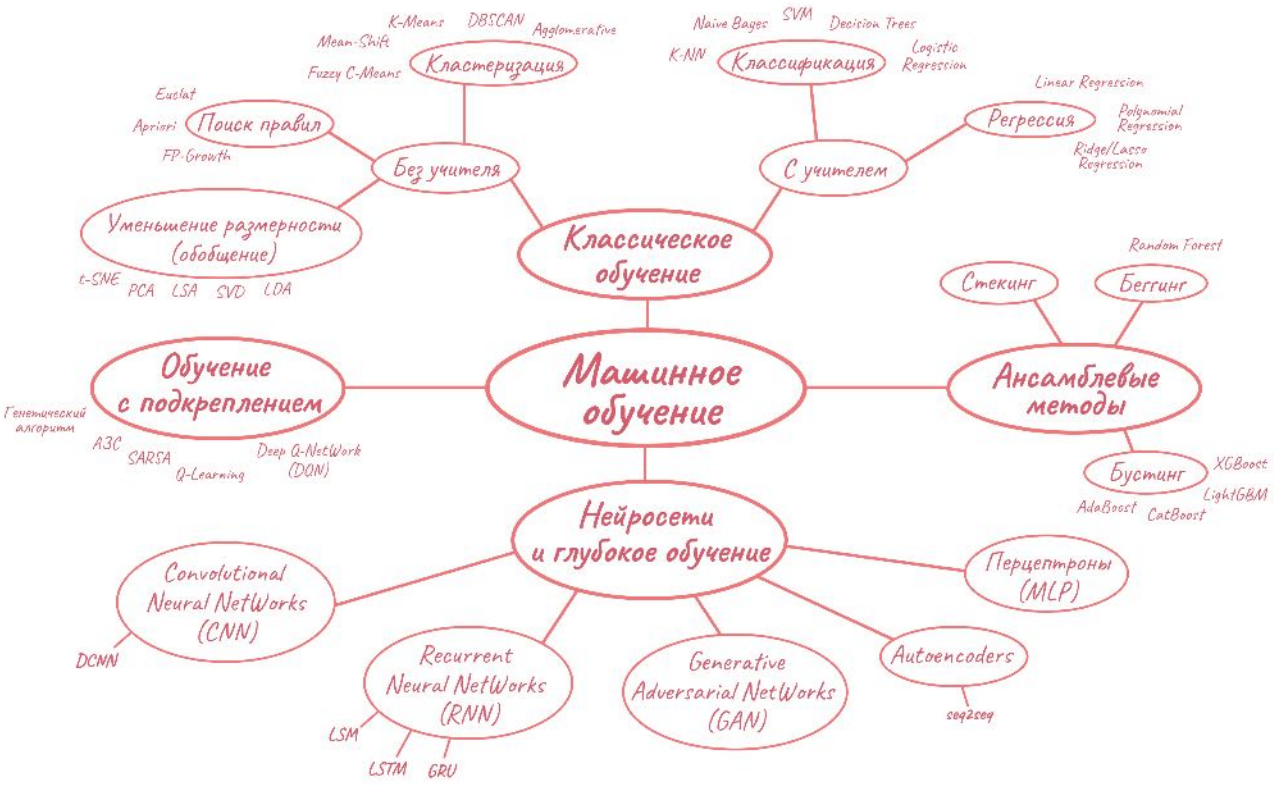

Рис. 3: Карта мира МО

3 Классификация алгоритмов машинного обучения

3.1 Классическое обучение: обучение с учителем

Все алгоритмы классического обучения основаны, в основном, на аппарате математической статистики, так что для понимания происходящего придется что-то да вспомнить. Но, честно говоря, они, в большинстве своем, настолько просты, что на пальцах объясняются даже ребенку. Но не нужно недооценивать эти методы: решение доброй половины задач современного общества основано именно на них.

Итак, классическое обучение делится на обучение с учителем и без. В обучении с учителем, что ясно и из названия, есть некий учитель, который говорит, как правильно. Иными словами, есть не только входные данные, но и реальные выходные (как ответы в сборнике задач). Например, объявления о продаже квартир содержат в себе «верные», установленные цены на жилье. В итоге, при обучении с учителем всегда есть некий тренировочный набор данных – набор, на котором модель обучается.

Обучение без учителя выходных данных не содержит. При обучении без учителя на машину просто вываливают гигабайты входных данных и говорят

что-то типа: ну, разберись, что тут к чему? Зачастую люди и сами не знают, что и к чему, и даже не знают есть ли что-то интересное в этих данных, а машина пытается найти скрытые закономерности и разложить все по полочкам. Представьте, например, что несколько однотипных огромных книг рассыпались на страницы, причем страницы не пронумерованы. Как думаете, легко собрать из этих страниц книги в их первоначальном виде? А если книги, к тому же, на иностранном языке? А если вы не знаете изначально исходное количество книг? Ох.

Обучение с учителем, как мы видели, и как снова же отражено на рисунке, глобально можно разделить на две ветки – регрессия и классификация.

Рис. 4: Классическое обучение

3.1.1 Задача регрессии

Задачу регрессии мы уже не раз вспоминали. Задача регрессии – это задача предсказания числа: цены квартиры или телефона, курса доллара на завтра, ожидаемого объема продаж, медицинских показателей до/после лечения и так далее. По сути своей компьютер решает задачу, известную из детства: есть набор точек, иллюстрирующих данные, и нужно их как-то соединить, чтобы получилось красиво и логично. И вот с этим-то самые большие проблемы.

Вот вам пример такой задачи, взгляните на рисунок 5. Данные, кстати, близки к реальным, хоть и сгенерированы. Рисунок показывает зависимость добавки к стоимости квартиры (в тысячах долларов) от времени транспортной доступности до ближайшего метро (в минутах). Всего сгенерировано 100 рекламных объявлений. В качестве предиктора X выступает время, а в качестве отклика Y – добавка к стоимости.

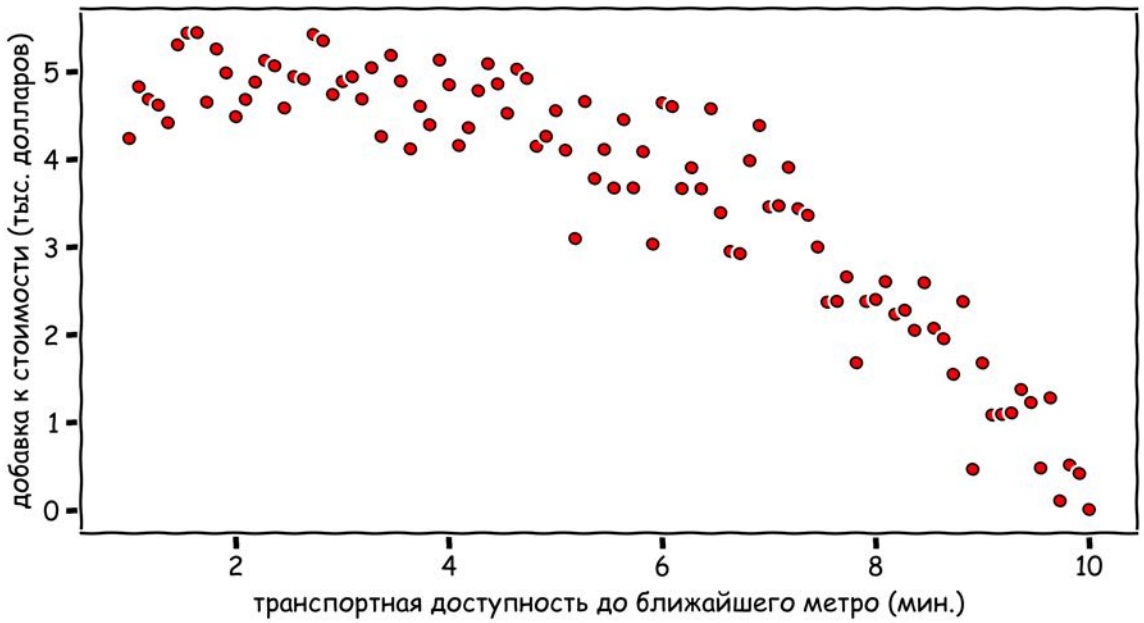

Рис. 5: Добавка к стоимости в зависимости от удаленности от метро

Ну и как же соединить точки, какая хоть примерно зависимость? Конечно, можно сделать то, что приходит, наверное, первым в голову, но сомнительно, что цена ведет себя именно так как показано на рисунке 6.

С этим «приближением» куча непоняток: чем объясняются такие скачки то вверх, то вниз? Почему функция такая сложная и зубчатая (или, как говорят математики, не гладкая)? Как получить вид функции для предсказания, какое у нее аналитическое задание? Все эти вопросы восходят к так называемой проблеме интерпретации модели – ее толкованию и трактованию.

Регрессия предлагает весьма наглядные модели: линейные или полиномиальные. Синим (рисунок 7) показано приближение линейной функцией (от

Рис. 6: Наивная оценка истинного распределения

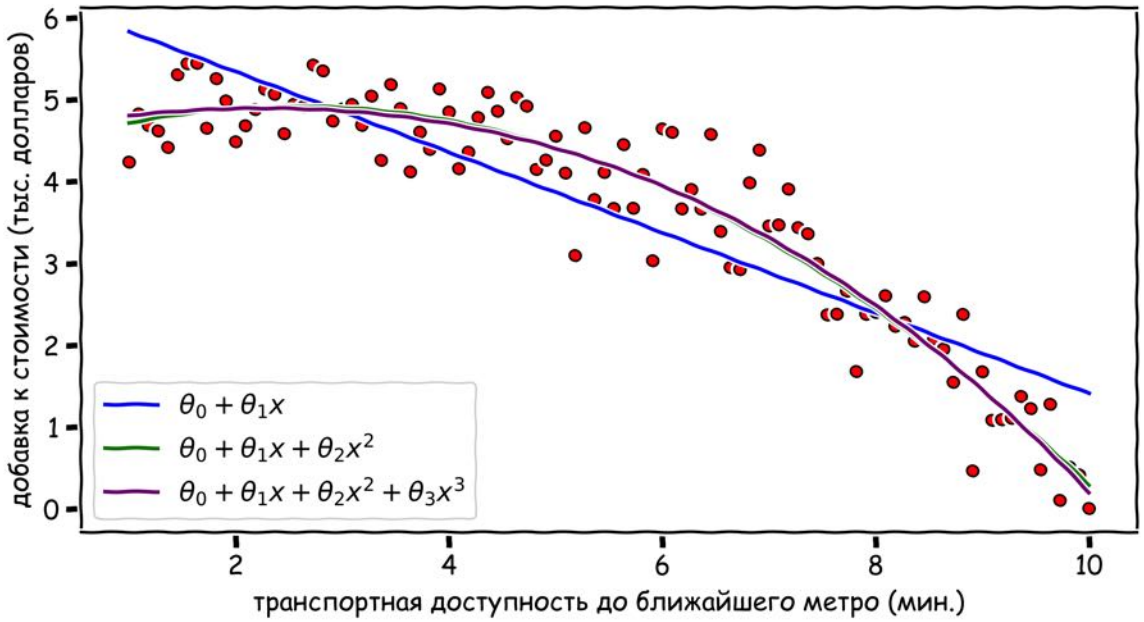

Рис. 7: Регрессия

того и линейная регрессия), зеленым – квадратичной, а фиолетовым – кубической. Видно, что квадратичное и кубическое приближения очень даже неплохо повторяют форму зависимости. Конечно, ошибки обусловлены ошибками модели, но такая модель, очевидно, очень наглядна и легко объясняема. Подробнее о том, как находить коэффициенты, мы обсудим во второй лекции.

3.1.2 Задача классификации

Задача классификации – задача отнесения объекта к одному из заранее известных классов. Не хотим вас пугать, но и задачи классификации вы решаете не менее регулярно, чем задачи регрессии. Скажем, после стирки носки нужно разделить по цвету (красные, синие, черные) и по размеру (большие – мужа, маленькие – жены). А при разборе сумок после похода в магазин овощи часто складываются в одну корзину, а фрукты – совсем в другую. И уж порошок точно среди них не валяется.

В итоге – задача классификации – это та задача, которую уже и ребенок решает с самого рождения: большое к большому, маленькое к маленькому, машинки к машинкам, а роботы – к роботам. А что, если это трансформер? Тут и взрослый встанет в ступор.

Рис. 8: Детский классификатор

Конечно, зачастую решаются более сложные задачи. Например, когда человек приходит в банк, чтобы взять кредит, сотрудник банка должен выяснить: кредитоспособен человек или нет. Понятно, что это делается на основе персональных данных человека: его дохода, его места работы, семейного положения, возраста, наличия других кредитов и многого-многого другого. Проанализировав всю сводку информации, банковский работник должен принять решение, к какому классу отнести этого «потенциального клиента» – «кредитоспособен» или «некредитоспособен».

Другой пример задачи классификации – обнаружение спама в электронной почте. Почтовый клиент должен определить, принадлежит ли письмо нежелательной почте (спаму) или нет. Конечно, это делается на основе различных данных, как, например, количество адресатов, количество слов «купить», «продать», «заработать», «репутации» данного адреса у почтового клиента. К вопросу фильтрации спама мы еще непременно вернемся в лекции о байесовском классификаторе.

Задача классификации вовсе не обязана иметь лишь два исхода. Например, задача распознавания рукописных цифр, будучи задачей классификации, включает десять исходов, а задача распознавания слов – такое количество исходов, что его сложно даже произнести (а точно посчитать – тем более). Есть разные методы классификации. В этом курсе мы рассмотрим с вами логистическую регрессию, наивный байесовский классификатор и метод k-ближайших соседей, а в курсе «Advanced Machine Learning» еще и машины опорных векторов (SVM).

3.2 Классическое обучение: обучение без учителя

Мы уже сказали несколько слов про обучение без учителя ранее. Как утверждается, оно изобретено в 90ых годах, то есть гораздо позже, чем обучение с учителем. Используется обучение без учителя реже, но иногда просто нет выбора: ведь данные вовсе не всегда размечены, и далеко не всегда в них наведен должный порядок. Итак, согласно рисунку, обучение без учи-

теля включает в себя три основные ветки: кластеризация, поиск ассоциаций, уменьшение размерности.

Рис. 9: Классическое обучение

3.2.1 Кластеризация

Задача кластеризации тесно связана с задачей классификации. Наверное, каждому знакомы оба этих термина, но далеко не каждый может сразу сказать: а чем они отличаются? Так вот, кластеризация – это та же классификация, но без заранее известных кластеров. Машина сама должна искать похожие объекты и объединять их в классов. Количество кластеров заранее тоже неизвестно и в алгоритмах либо задается заранее человеком, либо, снова же, дается на откуп машине.

Посмотрите хотя бы на рисунок: на левом представлены картинки девяти котов. Ну и на сколько бы кластеров вы их разбили? Наверное, это зависит от степени погруженности в детали. На первый взгляд – это все просто коты. Но если начать решать задачу «найди отличия», то можно заметить, что у некоторых котов зрачок круглый, а у некоторых – овальный. Кроме того,

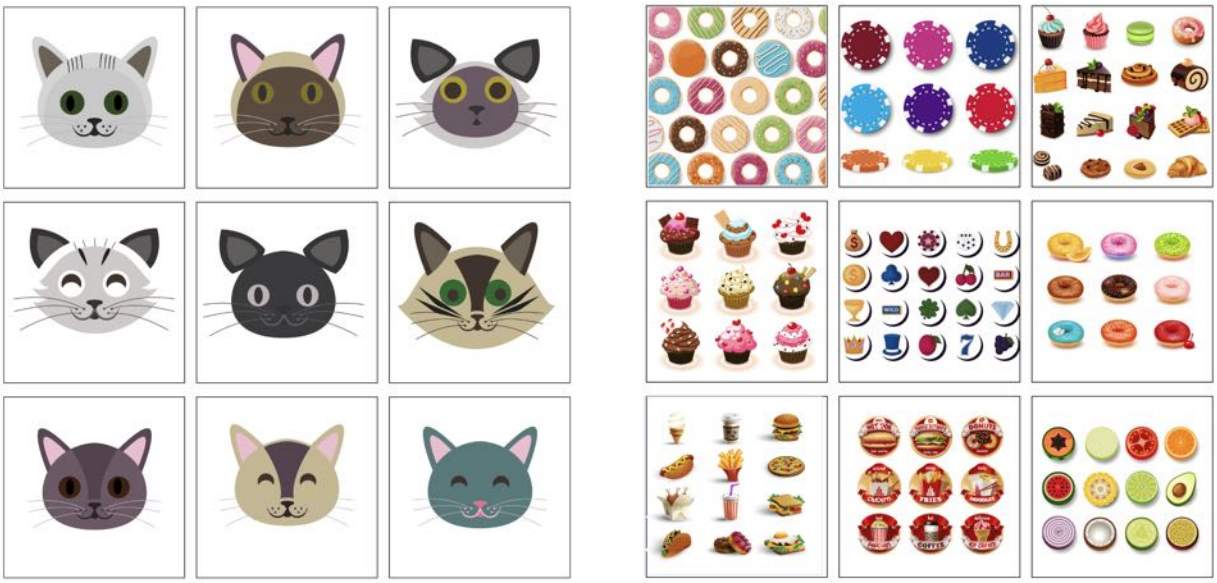

Рис. 10: Немного о кластеризации

у части котов радужная оболочка глаза желтая, у части – коричневая, а у части и вообще зеленая. Да и вообще, коты еще отличаются по окрасу: есть серые, есть голубые, есть серо-полосатые, есть бело-полосатые. Да, кстати, а это коты или кошки? Ну и сколько кластеров у нас уже получилось? А ведь если пригласить специалиста по котам, он, наверное, назовет еще с десяток отличий.

Тут естественным образом возникает вопрос: а сколько кластеров реально нужно? Это, естественно, зависит от задачи. Если достаточно просто определить тип животного, то хватит и одного, а если уже и породу, то, скорее всего, перечисленных нами не хватит. И мы еще в выигрышном положении: мы хотя бы знаем (вроде как), что все, что на картинках – представители кошачьих.

На второй картинке все менее радужно: тут присутствуют и пирожные, и выпечка, и какие-то непонятные то ли значки, то ли фишки. Как тут разобраться, если и экспертно, с помощью человеческого интеллекта, ничего не понять? А уж как тут работать машине, какие кластеры выбрать? Съедобное-нет, сдобное-нет, по форме, по цвету, по начинке, по оформлению... Да вариантов просто тьма. А может просто – все круглые? Согласитесь, задача не кажется простой.

В нашем курсе мы рассмотрим несколько алгоритмов кластеризации: это метод K-средних и агломеративная (иерархическая) кластеризация.

3.2.2 Ассоциации

Поиск ассоциаций, наверное, одна из самых свежих и мало разработанных веток машинного обучения. Сюда входят и анализ продуктовой корзины,

и прогноз акций и распродаж, и расстановка товаров на полках и так далее.

Например, вы покупаете чай. Стоит ли на вашем пути поставить разные сладости: печеньки, конфеты, торты? А стоит ли тут же выставить кофе? Как часто эти продукты берут вместе? Как часто, взяв чай, человек захочет взять и кофе? А может тогда рядом с кофе нужно поставить молоко для коктейлей, или, скажем, какие-то сиропы? Ведь если этого не будет, то покупатель может просто забыть купить тот или иной товар. А часто ему может просто не прийти это в голову, ведь раньше он и думать не думал добавить в кофе корицу или, например, карамельный сироп. Если вы владелец сети гипермаркетов, то эти вопросы будут крайне актуальны!

Или интернет-магазины, да и любые интернет-сервисы. Вы замечали, что стоит вам только поискать, скажем, какую-то бытовую технику (в прочем, как и любой другой товар), то в течение продолжительного времени то и дело сбоку экрана вы будете получать предложения различных магазинов по интересующим вас товарам и такие, и сякие, и с перламутровыми пуговицами – только купи. Это и есть – ассоциации.

3.2.3 Уменьшение размерности

Часто так бывает, что в данных имеется настолько много предикторов, что ничего не понять. Мы не можем данные ни визуализировать, чтобы высказать какие-то предположения о зависимостях (так как не хватает размерности пространства), ни понять вообще: а что в этих данных происходит. Тогда часто имеет смысл объединить признаки в один и получить какую-то более общую абстракцию. Ну например. Предположим, что у нас имеются данные о птицах в городе: среднего размера, серые, редко белые, иногда коричневые, с красными лапами. Ну что, можно выкинуть все эти данные и схлопнуть их до одного – перед нами голубь. Конечно, мы теряем информацию о конкретном голубе (да и иногда это может быть и вообще не голубь), но, честно говоря, абстракция голубь несет в себе куда больше смысла, чем все эти цвета, лапы и размеры. Да и модель обучается на меньших размерностях куда лучше и быстрее, чем на больших.

А иногда данные просто мало отличаются. Представьте, что мы рассчитываем среднюю массу продуктовой корзины человека с какими-то параметрами, и, например, знаем, что печенье весит 500 грамм $\pm 5\%$, что является погрешностью производителя. Ну и зачем нам это знать в нашей конкретной задаче? Ясно, что достаточно просто считать, что если человек берет упаковку печенья, то масса увеличивается на 500 грамм, и все! Вот и опять уменьшение размерности.

С некоторыми методами снижения размерности мы познакомимся с вами в курсе «Advanced Machine Learning».

3.3 Обучение с подкреплением

Обучение с тически порог искусственного интеллекта. Почему? Да потому что по сути дела машина просто-напросто пытается выжить в реальной среде. За каждое действие она получает баллы: положительные, если она сделала хорошо, нейтральные, если, условно, ничего особо не поменялось, или отрицательные, если опростоволосилась. Цель, как и у студентов, набрать как можно больше баллов! Почему это удобно?

Рис. 11: Обучение с подкреплением

Ну смотрите, есть такая достаточно известная и весьма сложная игра Го. И машина не так давно научилась обыгрывать человека. И что, скажете вы? Вроде в шахматах это происходит достаточно давно. Так вот, сравнительно недавно было доказано, что число комбинаций в игре Го физически невозможно просчитать, так как этих комбинаций больше, чем атомов во вселенной (число комбинаций состоит из 171 одной цифры :)). А вот в шахматах машине вполне себе реально просчитать просто ВСЕ будущие ходы, а значит ей достаточно просто выбрать лучшую для себя ситуацию. Чувствуете разницу? В Го требуется что-то кроме лобового брутфорса (перебора)! Интеллект, что ли?

Ну и что же делает машина? А она просто старается выйти из каждой ситуации с меньшими потерями, с наилучшими возможными баллами. Для этого машина производит миллионы симуляций в среде, считает свои очки, и выбирает что-то лучшее. Только вот проблема: а кто сказал, что исчерпаны и изучены все ситуации? Скажем, вот стоите вы на перекрестке, горит зеленый. И что, можно переходить? А вдруг из-за угла поворачивает водитель и не видит вас, а вы, в свою очередь, в наушниках, и не слышите визга

колес? Вы уверены, что машина просчитала такую ситуацию? А она вообще знала о такой возможности? Пока нет окончательного ответа на то, как предупреждать такие ситуации, исследователи придумывают все новые и новые «костыли» – локальные решения проблемы, а глобальных решений пока что нет.

Идея описанного лежит в основе алгоритма Q-learning, обсуждаемого в курсе «Advanced Machine Learning», и она очень условно проиллюстрирована на рисунке. За правильные ходы машина получает плюшки в виде плюсов, а за неправильные – минусы. В некотором смысле, обучение можно сравнить с дрессировкой животного: за верно выполненную команду вы поощряете собаку лакомством или игрушкой, а за неверное строго отчитываете, а иногда даете и подзатыльник. Вот и вся схема.

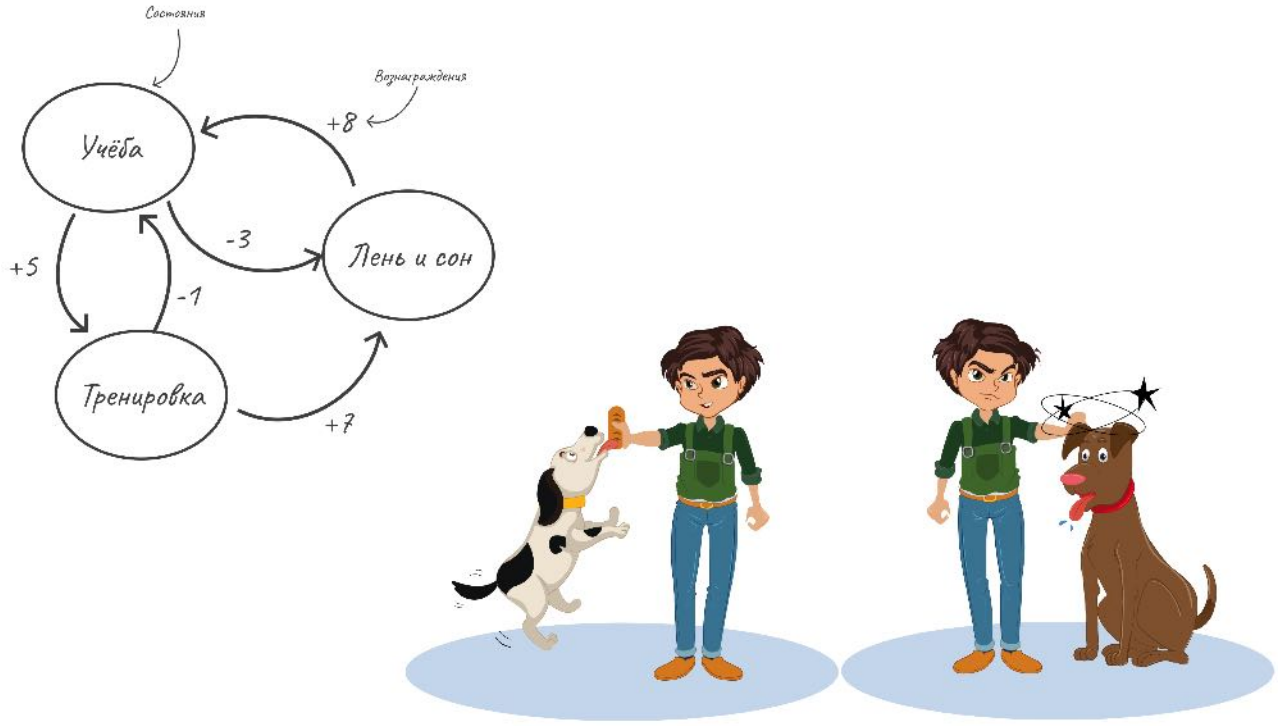

Рис. 12: Q-learning

3.4 Ансамблевые методы

На данный момент ансамблевые методы являются одними из самых точных при решении различных задач. Идея достаточно проста и логична – объединить разные методы по принципу дополнения друг друга: где один метод не справился и допустил много ошибок, там его ошибки исправит другой. При этом даже лучше, чтобы методы были скорее нестабильными и плавающими на входных данных, нежели суперэффективными и устойчивыми. Давайте коротенько пройдемся по некоторым подходам.

Стекинг действует по следующему принципу: обучи несколько различных алгоритмов и передай их на вход последнему, который и примет финальное решение.

Беггинг предполагает обучение одного и того же алгоритма много раз на случайных выборках из исходных данных, а в конце усреднение ответов.

Бустинг предлагает обучать алгоритмы последовательно, причем при каждом следующем обучении новый алгоритм уделяет особое внимание ошибкам предыдущего. Как и в беггинге, выборки берутся из исходных данных, но не совсем случайно, что, правда, уже было сказано: к данным обязательно добавляются те, на которых ошибся предыдущий алгоритм.

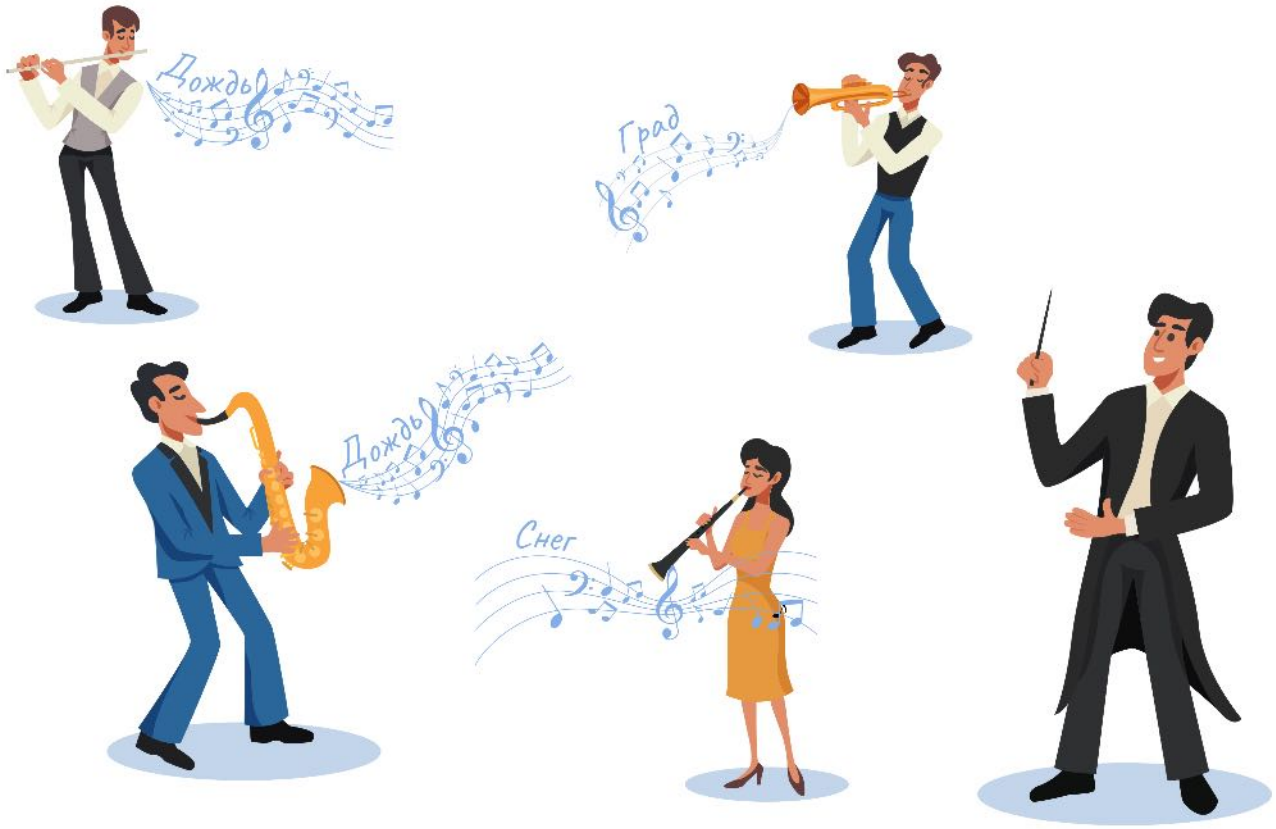

Рис. 13: Ансамблевые методы

Эти методы уже не так хорошо иллюстрируются, как предыдущие, их мы подробно обсудим в курсе «Advanced Machine Learning».

3.5 Совсем немного о нейросетях и глубоком обучении

Давайте начнем с того, что такое нейросеть. Нейросеть – это набор нейронов и связей между ними. Нет, вовсе не как в нашем организме. По сути дела нейрон – это функция с кучей входов и одним единственным выходом.

В итоге, нейрон выполняет вот какую задачу: на вход он берет всякие числа с «входов», что-то с ними делает (применяет к ним некоторое преобразование или функцию), и выдает на выход результат проделанной работы.

Что же такое связь? Связь – это канал, через которые нейроны шлют друг другу какие-то числовые значения. У каждого канала есть свой вес (ну или пропускная способность, или прочность связи). Например, если через связь с весом 0.5 проходит число 10, то на вход нейрону подается число $0.5 \cdot 10 = 5$.

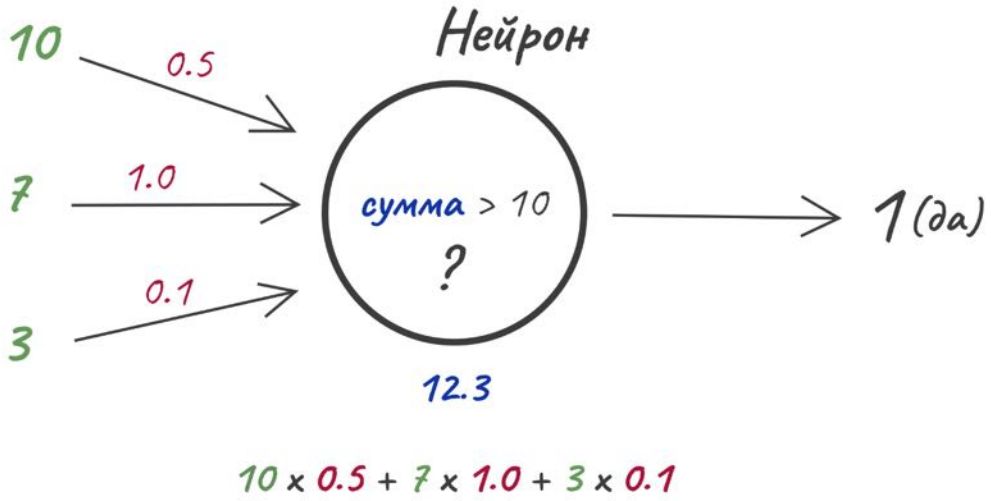

Рис. 14: Пример нейрона

Чтобы связи не были какими попало, нейроны решили связывать по слоям: внутри одного слоя нейроны не связаны между собой, а соединены с нейронами только предыдущего и следующих слоев. В реалиях все это представляется матрицами и вычисляется средствами линейной алгебры. Ну да не будем об этом.

В принципе, все. Мы рассмотрели основные задачи и схемы машинного обучения и готовы приступить к самим алгоритмам. Но для начала немного самой базовой статистики.

4 Чуть-чуть необходимой общей статистики

Мы, в общем-то, будем обсуждать различные статистические методы непосредственно перед задачами, для которых они нам потребуются. Но все-таки нам кажется полезным напомнить, как определяются основные характеристики (одномерной) выборки.

Все обычно начинается с генеральной совокупности. Генеральная совокупность ξ – это и есть та случайная величина, с которой мы имеем дело,

которая дает нам данные. Давайте сразу привяжемся к какому-то примеру. Пусть генеральная совокупность ξ – это та сумма денег, которую человек оставил в супермаркете – вполне себе такие данные. Ясно, что значение этой суммы, то есть значение ξ, меняется как от человека к человеку, так и у одного и того же человека при разных походах в магазин. На практике множество значений ξ всегда конечно, ведь, например данные берутся за прошедший месяц (да и просто за какой-то период времени), а за месяц было обслужено конечное число покупателей.

Мы считаем, что случайная величина ξ подчиняется какому-то закону, этот закон мы будем называть ее распределением. Наверное, из курса статистики вы помните популярные распределения: нормальное, равномерное, распределение Бернулли, биномиальное распределение. На данном этапе не очень важно понимать досконально что означает этот термин – распределение, но важно понимать одно: мы предполагаем, что какой-то закон, по которому меняется ξ, есть. А законы отличаются чем? Правильно, вероятностями значений. В нашем примере, ясное дело, количество потраченных денег не может быть отрицательным. Значит, у нашей случайной величины такое распределение, что вероятность принять отрицательные значения равна нулю. В то же время, в супермаркете редко можно увидеть кого-то, кто заплатит больше 500 долларов. Значит вероятности того, что ξ принимает значения большие, чем 500 долларов чрезвычайно малы. Ну а что там между этими границами? Это, похоже, и интересно узнать. Как? По выборке.

Выборкой объема (или размера) n из генеральной совокупности ξ называется набор независимых случайных величин $X_1, X_2, ..., X_n$, имеющих распределение такое же, как и ξ. Часто еще говорят о случайном векторе $\vec{X} = (X_1, X_2, ..., X_n)$. Мотивировка остается прежней: если поменять день или час наблюдения, то поменяется и выборка, именно потому элементы выборки – случайные величины. При конкретном же наблюдении выборка – это набор из n чисел $x_1, x_2, ..., x_n$ или $\vec{X} = (x_1, x_2, ..., x_n)$.

Оказывается, по выборке можно оценивать математическое ожидание, дисперсию, среднее отклонение случайной величины. Давайте вспомним, как это делается?

Что такое математическое ожидание? Если говорить на пальцах, то это среднее вероятностное значение рассматриваемой случайной величины. Что означают эти слова? Ну смотрите, давайте немного упростим рассматриваемый общий пример. Будем рассматривать только тех людей, кто ничего не потратил, или потратил очень много. Величина чека равна нулю, если покупатель ничего не купил в магазине – не такое уж редкое явление. В то же время потраченная сумма, большая 500 долларов, хоть и возможное, но очень редкое явление. Зададимся вопросом, сколько «в адекватном среднем» тратит человек на покупку в магазине (еще раз, при рассмотрении только

тех, кто либо ничего не тратит, либо очень много)? Ну что, среднее арифметическое? То есть

$$\text{Средние траты} = \frac{0+500}{2} = 250?$$

Не кажется ли это странным? Ведь, скажем (данные взяты из головы), на 100 учтенных покупателей, попадающих под критерии, 99 тратят ноль, и только один тратит 500. Видимо, резонно записать это так:

ξ	0	500
P	$\frac{99}{100}$	$\frac{1}{100}$

.

Это знакомая запись – таблица распределения дискретной случайной величины. В первой строчке стоят значения, а во второй – вероятности (вычисленные на основе частоты). Что разумно назвать математическим ожиданием $\mathsf{E}\xi$? Похоже, величину

$$\mathsf{E}\xi = 0 \cdot \frac{99}{100} + 500 \cdot \frac{1}{100} = 5.$$

Не кажется ли вам, чисто интуитивно, что полученная величина куда ближе к истине?

Но тут проблема: значений вероятностей мы не знаем. Именно поэтому в статистике математическое ожидание мы хотим оценить (характеристика-то полезная). Легко понять, что кандидатом является так называемое выборочное среднее

$$\overline{X} = \frac{X_1 + X_2 + ... + X_n}{n},$$

которое на конкретном наблюдении $\vec{X} = (x_1, x_2, ..., x_n)$ превращается просто в среднее арифметическое элементов выборки

$$\overline{X} = \frac{x_1 + x_2 + ... + x_n}{n}.$$

Рассмотрим, для примера, вот такую выборку

$$X = (0, 11, 2, 3, 9, 2, 8, 6, 3.4, 8, 7.5, 9, 4, 8, 6).$$

Тогда выборочное среднее вычисляется, как

$$\overline{X} = \frac{0+11+2+3+9+2+8+6+3.4+8+7.5+9+4+8+6}{15} \approx 5.793.$$

На рисунке 15 синие точки – это элементы выборки. Красная точка отвечает выборочному среднему $\overline{X}$.

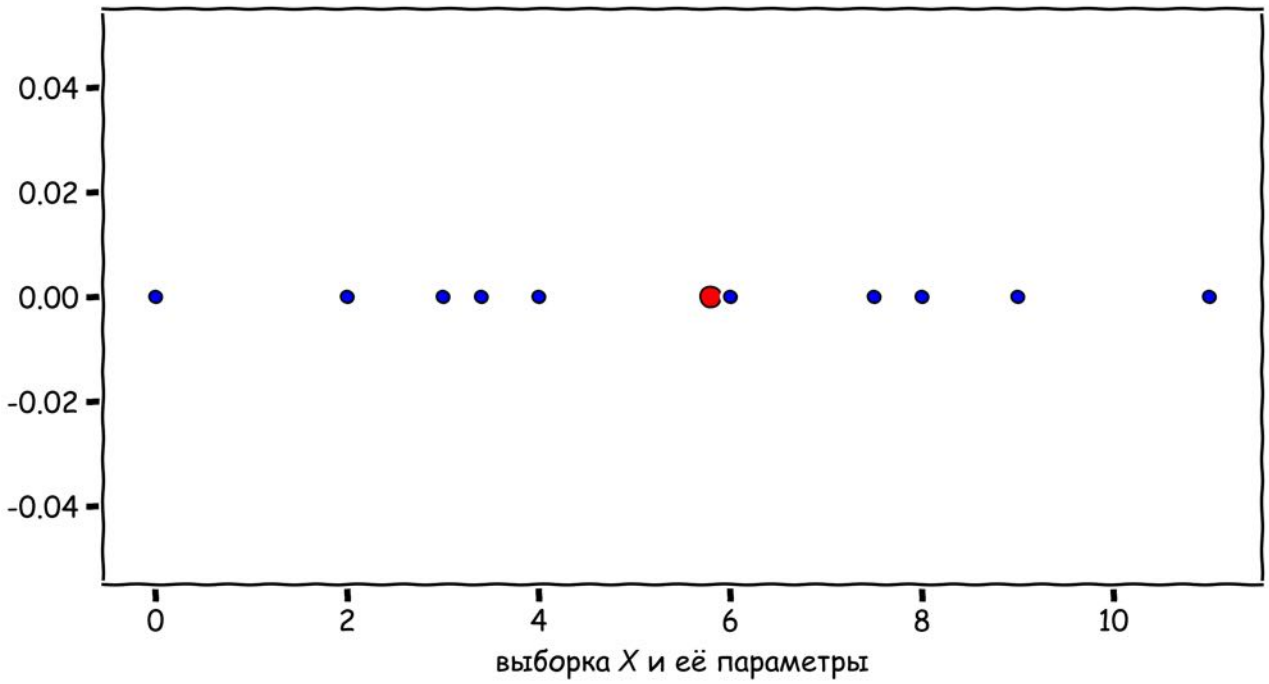

Рис. 15: Выборка X и ее среднее $\overline{X}$

Вот видите, красная точка и правда примерно «посередине» наших синих точек – элементов выборки. Можно доказать, что с ростом объема выборки выборочное среднее все лучше и лучше приближает истинное математическое ожидание генеральной совокупности ξ.

Вторая важная характеристика – это разброс. Все на том же рисунке отчетливо видно, что синие находятся не очень-то близко друг к другу. Естественно задаться вопросом: а насколько они разлетаются от среднего в разные стороны? В какой-то степени можно провести аналогию со стрельбой по мишени: хочется попасть в «яблочко», но зачастую это не удается, попадание происходит в какую-то другую точку мишени.

Вот и в примере с покупками в гипермаркете, мы можем примерно понять: насколько велик разброс от среднего? Зная разброс, можно прогнозировать выручку за какой-то период времени как в «худшем», так и в «лучшем» случаях. Давайте что-то посчитаем. В примере с гипермаркетом мы рассматривали случайную величину ξ, имеющую следующее распределение:

ξ	0	500
P	$\frac{99}{100}$	$\frac{1}{100}$

и математическое ожидание

$$\mathsf{E}\xi = 0 \cdot \frac{99}{100} + 500 \cdot \frac{1}{100} = 5.$$

И что, разброс равен $\max|\xi - \mathsf{E}\xi| = 495$? Опять же, странно, если это так, ведь значение 500 крайне маловероятно. Может опять, взять средний разброс? Вот

и появляется дисперсия, которая, как мы напомним, определяется, как

$$\mathsf{D}\xi = \mathsf{E}\left(\xi - \mathsf{E}\xi\right)^2.$$

Легко видеть, что распределение случайной величины $\left(\xi - \mathsf{E}\xi\right)^2 = \left(\xi - 5\right)^2$ описывается следующей таблицей:

$$\begin{array}{c|c|c} \left(\xi-5\right)^2 & 25 & 495^2 \\ \hline \mathsf{P} & \frac{99}{100} & \frac{1}{100} \end{array}.$$

Ну а тогда

$$\mathsf{D}\xi = \mathsf{E}\left(\xi - 5\right)^2 = 25 \cdot \frac{99}{100} + 495^2 \cdot \frac{1}{100} = 2475.$$

Ого, это ведь огромное число! Намного больше, чем то, что мы получали весьма «наивным» способом. Мы нигде не напортачили? Смотрите, мы же вычисляем математическое ожидание квадрата и получаем на самом деле что-то вроде среднего квадрата разброса. Сам же разброс может быть оценен так называемым среднеквадратическим отклонением σ, равным

$$\sigma = \sqrt{\mathsf{D}\xi}.$$

В нашем примере $\sigma = \sqrt{2475} \approx 49.75$, что куда ближе к истине.

Снова, так как распределение вероятностей обычно неизвестно, то дисперсию нужно оценить. Хорошей оценкой дисперсии оказывается величина

$$S^2 = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2.$$

Иногда используют и такую оценку

$$S_0^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2,$$

но мы не будем вдаваться в такие детали. Отсюда, конечно, получается и оценка для отклонения σ:

$$\sigma^* = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2}, \quad \sigma_0^* = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2}$$

Вернемся к нашему совсем синтетическому примеру – к выборке

$$X = (0, 11, 2, 3, 9, 2, 8, 6, 3.4, 8, 7.5, 9, 4, 8, 6).$$

Проведя вычисления, получим, что $S^2 \approx 9.63$ и $\sigma \approx 3.1$. Это значит, что в большинстве случаев значения нашей случайной величины должны находиться в диапазоне

$$\left(\overline{X} - \sigma^*, \overline{X} + \sigma^*\right).$$

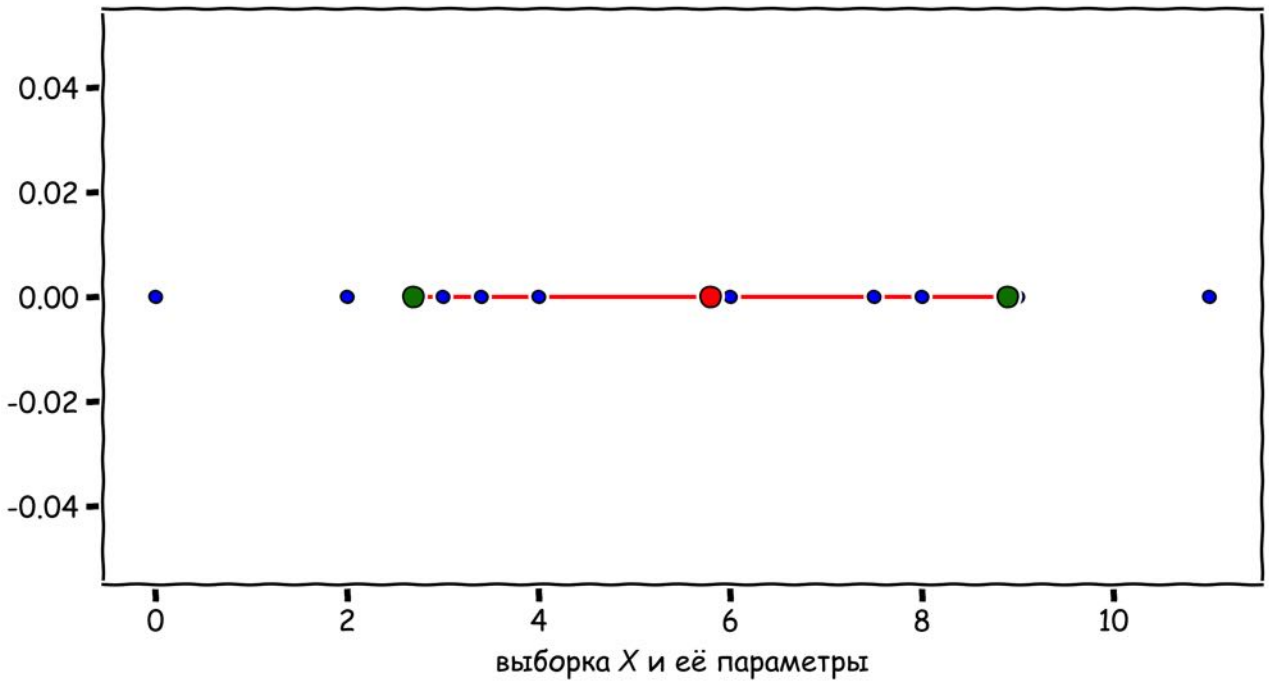

Рис. 16: Выборка X и ее параметры

На рисунке представлены выборка, изображенная синими точками, выборочное среднее, красной точкой, и границы интервала $\overline{X} \pm \sigma^*$ зелеными точками. Как видно, большинство элементов выборки находятся как раз-таки между зелеными точками.

Думаем, этих предварительных замечаний достаточно, чтобы окунуться в методы машинного обучения, а также в приложения этих методов к различным задачам. Удачи :)