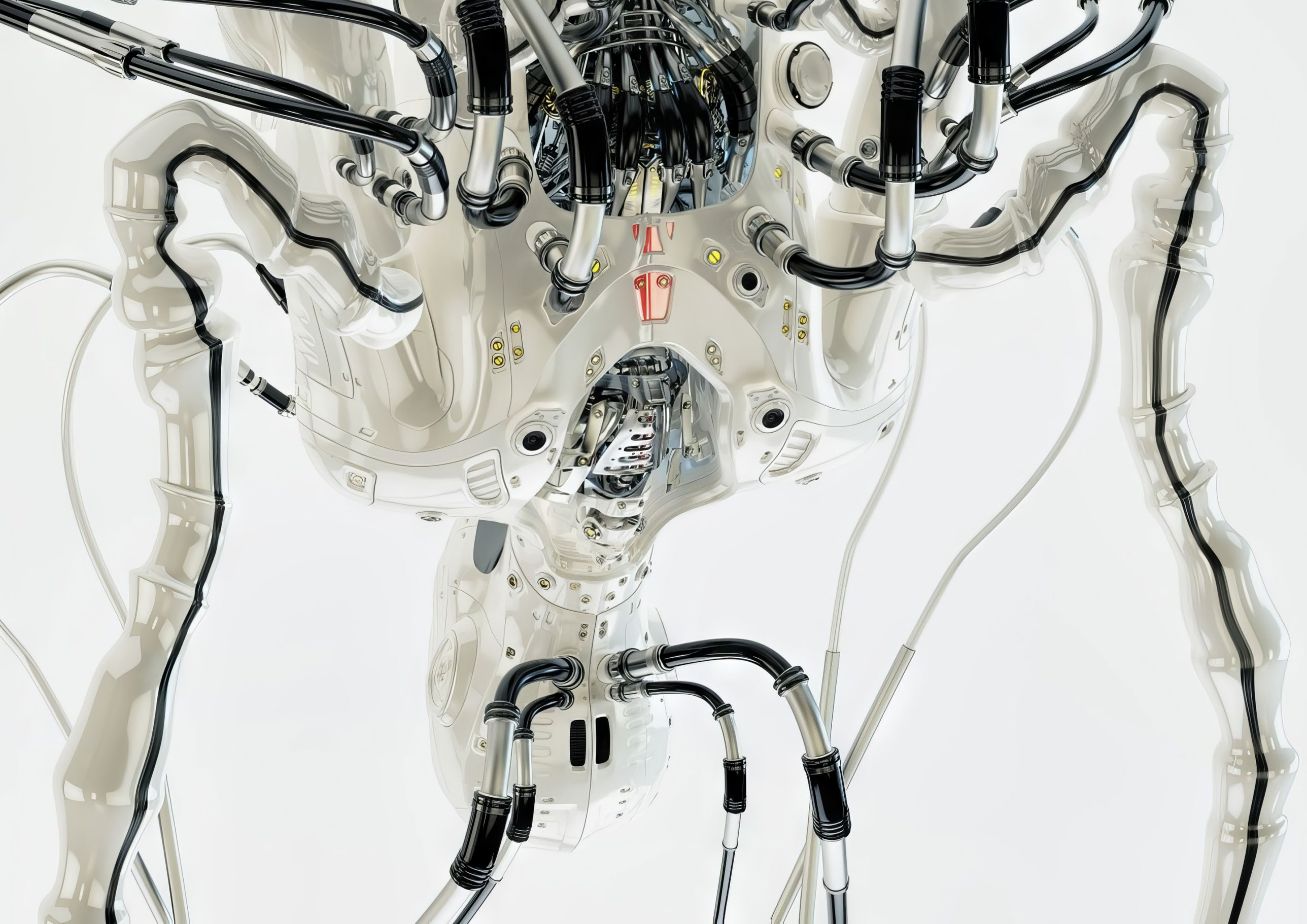


ОБУЧЕНИЕ С УЧИТЕЛЕМ + РЕГРЕССИЯ

В предыдущей главе мы получили общее представление о том какие бывают задачи и какие бывают методы их решения, еще раз перечислим их:

Задачи: регрессия, классификация, оптимизация, кластеризация

Методы: обучение с учителем, обучение без учителя, обучение с подкреплением, ансамблевые методы и нейронные сети.

В данной главе мы будем разбираться с тем, что из себя представляет **задача регрессии** и рассмотрим один из методов, который позволит ее решить, **метод обучения с учителем** (supervised learning).

Для начала нам необходимо осознать несколько базовых математических понятий и представлений:

1. Лекция – 1. Шапошников С. В. - Математический анализ I - Основные определения математического анализа (смотреть с самого начала и до 0:37:10). Ссылка: https://clck.ru/3Duvxv
2. Лекция – 2. Шапошников С. В. - Математический анализ I - Множество целых и рациональных чисел (смотреть с 0:11:43 до 0:44:30). Ссылка: https://clck.ru/3Duw2g
3. Лекция – 1. Овчинников А. В. – Понятия аналитической геометрии (смотреть с 50:40 и до 1:06:40). Ссылка: https://clck.ru/3Duw49

Функция – это правило, по которому каждому элементу одного множества соответствует один и только один элемент другого множества.

Декартово произведение – это множество упорядоченных пар. Например, если нам даны два множества X = {1,2,3} и Y = {1,2,3}, то их декартовым произведением X×Y будет множество {(1,1), (1,2), (1,3), (2,1), (2,2), (2,3), (3,1), (3,2), (3,3)}

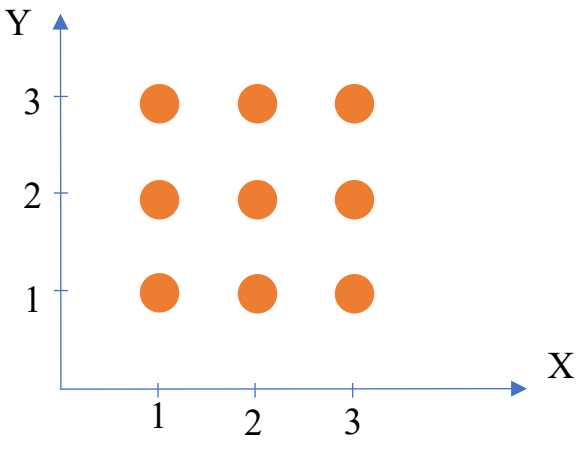
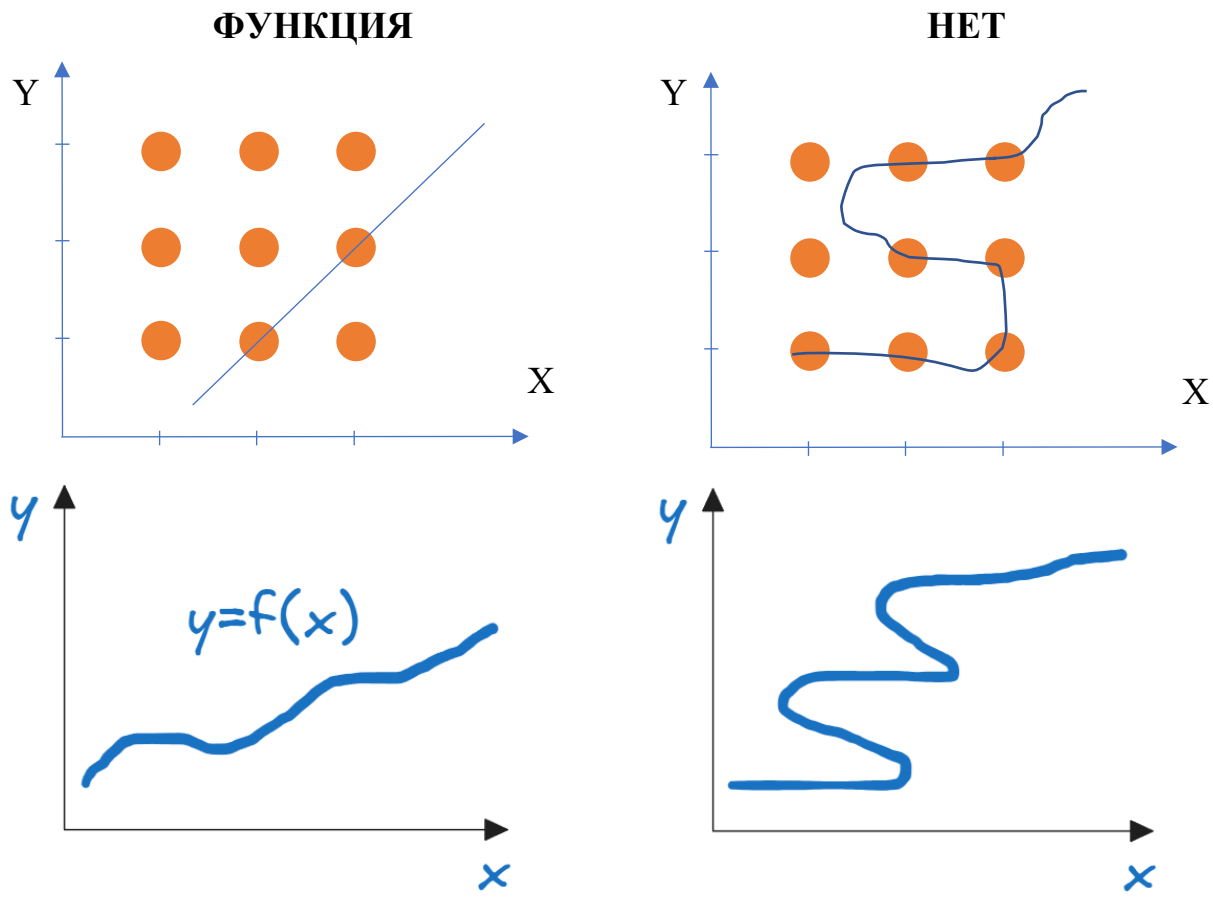
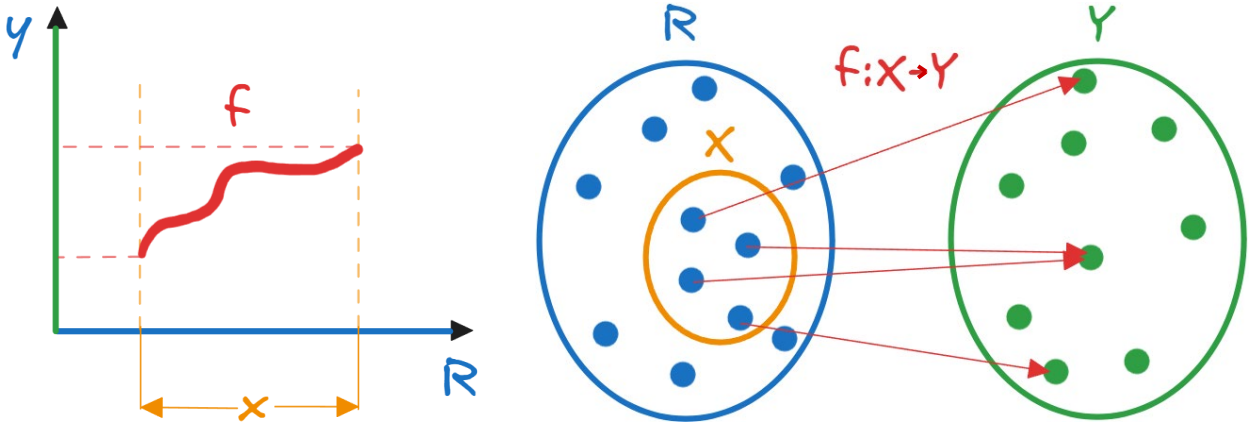


График функции — это подмножество декартового произведения. Причем слева функция (ее график), а справа не функция:



Замечание: очень часто на каких-то множествах выделяют подмножества и на них задают функцию.



Только и только после того, как вы полностью осознаете все базовые понятия переходите к понятию регрессия.

4. Что такое регрессия и какие виды регрессии имеются? Душкин объяснит. https://www.youtube.com/watch?v=qRXr21of4vA

Неформальное определение: **Регрессия** – это облако точек, которые мы хотим как можно точнее описать с помощью функции. Вот несколько примеров (см **рисунок 1**).

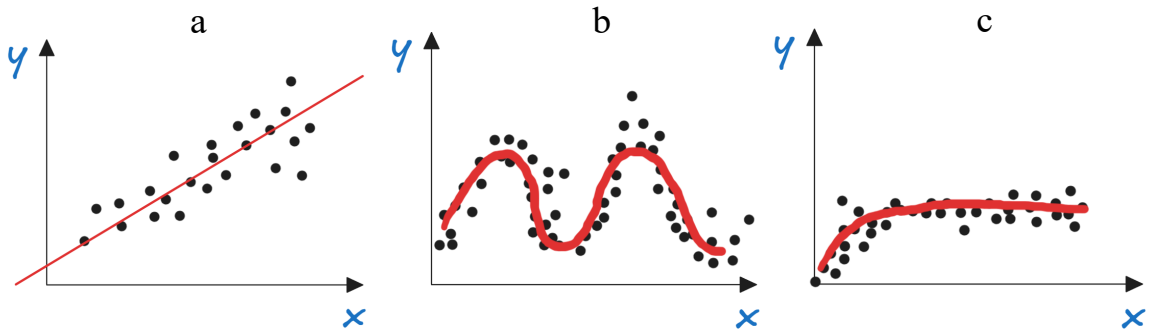


Рисунок -1 Виды регрессии:
a – линейная регрессия, b – полиномиальная регрессия
c – логарифмическая регрессия

Начнём с изучения **линейной регрессии**, так как это самый простой и понятный случай.

Чего мы хотим? Мы хотим решить задачу регрессии подобрав функцию, которая “рисует” прямую. То есть мы хотим рассмотреть случай, когда наше облако точек хорошо приближается линией (см. рисунок - 1,a).

Из школы нам известно, что функция, которая рисует линии, называется линейной функцией и она описывается уравнением **y = kx + b**. Коэффициент k отвечает за наклон прямой, а переменная b отвечает за смещение вдоль оси y, то есть b двигает нашу прямую вверх и вниз. Если вы этого не понимаете, я настоятельно рекомендую порисовать графики линейной функции.

Для расширения вашего кругозора:

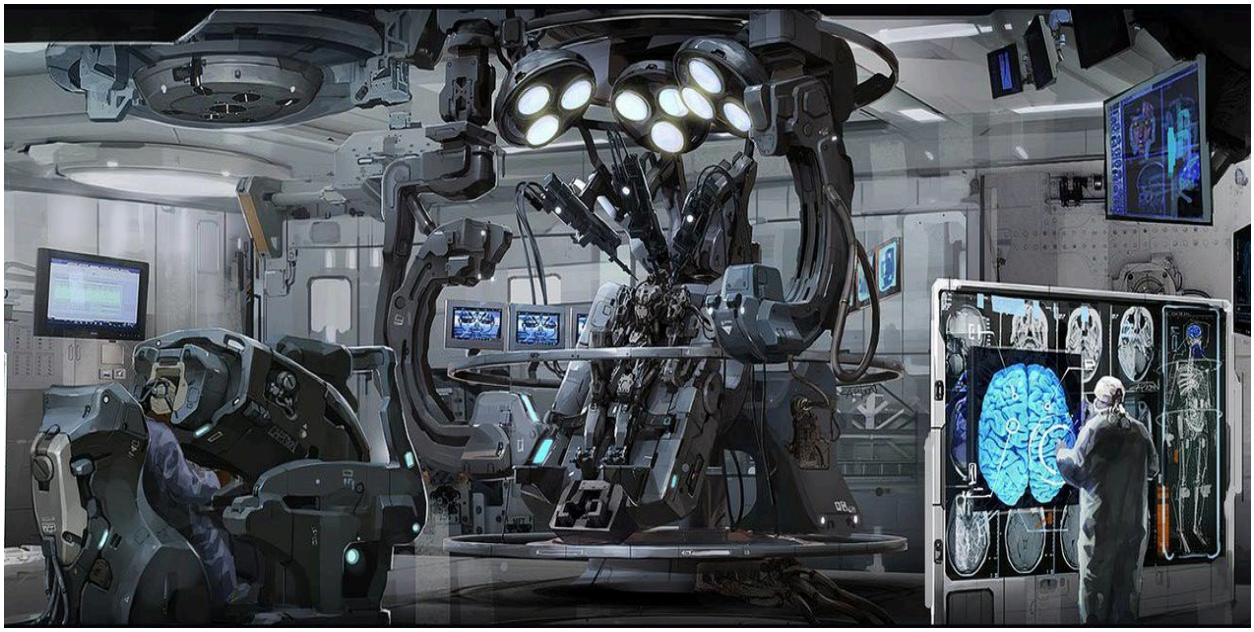
По сути, нам эта информация сейчас не нужна, но мне хочется открыть для многих секрет. На самом деле линейной функцией считается функция y = kx так как она отвечает свойствам аддитивности и однородности. А функция y = kx + b, называется *аффинным уравнением,* и эта функция не линейна, несмотря на то что тоже рисует линии.

Теперь ненадолго отложим регрессию и поговорим об общих идеях метода обучения с учителем и о том какие нам нужны входные данные для его реализации.

Представьте, что вы снова оказались в школе. Погодка сегодня выдалась чудесной. За окном светит осеннее солнце, легкий ветер слегка качает верхушки золотых деревьев, и где-то там вдалеке шелестят страницы учебников. В классе обсуждается новая тема, Звук мела о доску смешивается с приглушенными голосами одноклассников, задающих вопросы. Кто-то старательно записывает объяснения в тетрадь, а кто-то вполголоса переспрашивает соседа о том, что было сказано пару минут назад. “Вам все понятно?”, – спрашивает учитель: “Отлично, тогда приступаем к решению задач”. Учитель записывает на доске необходимые номера, которые следует прорешать, и вы приступаете к их выполнению. Решив все задачи, вы приносите тетрадь на проверку, учитель говорит, что у вас есть ошибки, вы не спешно садитесь обратно за парту и стараетесь их найти и исправить. И так по кругу – задача за задачей, исправление за исправлением.

В какой-то из дней учитель проводит контрольную работу, он дает вам похожие задачки на те, что вы уже решали, но которые вы ранее не видели. Если вы усвоили принцип, то вы запросто их решите, а если нет, то получите плохую оценку.

Убрав все формальности, можно сказать, что метод обучения с учителем это в точности тот же процесс, что и описан выше. Это цикл проб и ошибок, подкрепленный обратной связью. Давайте рассмотрим его более подробно



Для начала нам необходимо собрать и разметить исходные данные. То есть у нас должны быть примеры, на которые мы точно знаем ответы. Для тех, кто забыл, что такое разметка данных напомню:

“КОШКА”

“НЕ КОШКА”

Эти исходные данные объединяются в единый dataset, который называется обучающей выборкой.

Обучающая выборка (training set) – это набор пар (объект, ответ), который используется для того, чтобы "научить" алгоритм машинного обучения решать конкретную задачу.

Представьте, что вы учите ребенка распознавать фрукты. Вы показываете ему несколько яблок и груш и объясняете, что вот это — яблоко, а это — груша. После того, как ребенок посмотрел на эти примеры, вы даете ему новое яблоко или грушу и спрашиваете, что это такое. Если он хорошо "учился" на тех примерах, которые вы ему показали, он сможет правильно распознать фрукт. Обучающая выборка в этом случае — это те фрукты, которые вы ему показали и объяснили, как они называются.

Почему это называется выборкой? Потому что вы не показываете ребенку все яблоки и груши, которые существуют или существовали на планете земля, а показываете ему только те, которые купили на базарчике около дома, то есть вы показываете ему только какую-то часть от общего.

После разметки данных наступает этап их цифровизации. Поскольку большинство моделей машинного обучения работают с числами, довольно-таки часто приходится переводить информацию (текст, изображения, звуки и тд.) в бинарный формат. Этот процесс выходит за рамки текущего обсуждения, но будет подробно рассмотрен в главах, посвящённых работе с данными. Мы в качестве примера рассмотрим самый просто случай – кодирование категориальных данных.

Напомню, что категориальные данные— это данные, которые не могут быть выражены числами напрямую, например: цвета, фрукты, имена и тд. Для работы с такими данными есть два популярных способа кодирования:

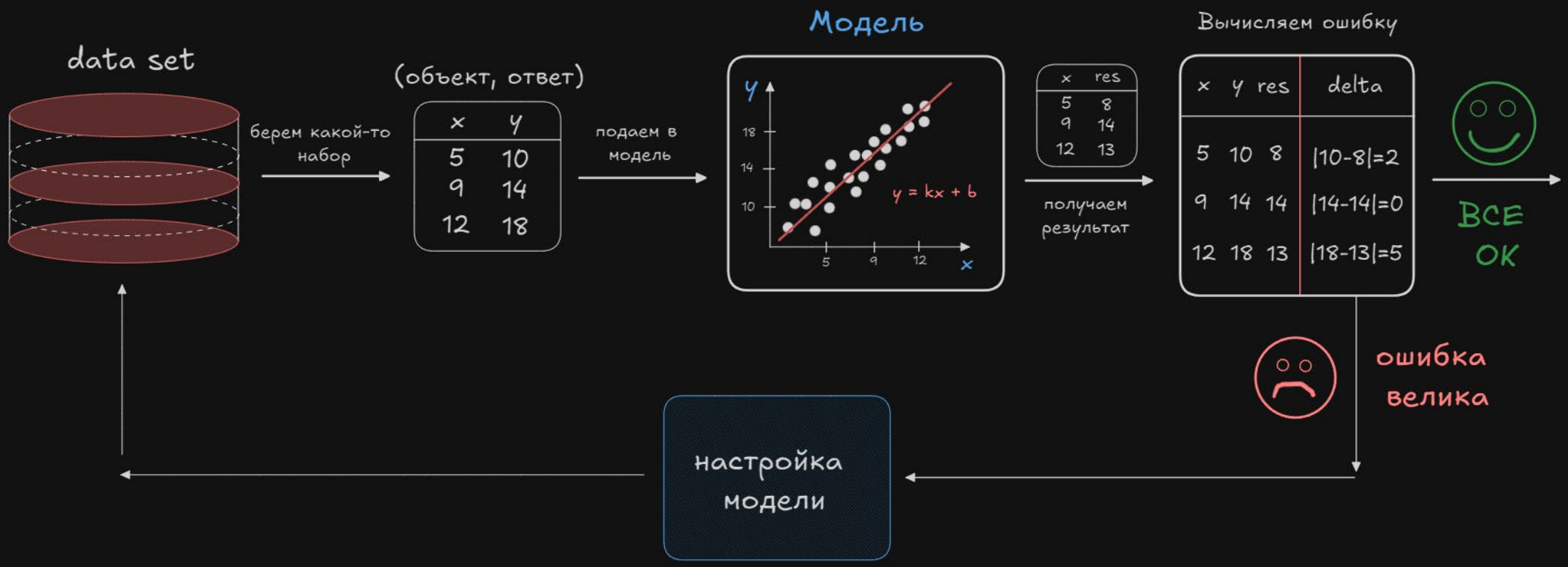
Label Encoding – присваивает каждой категории уникальный числовой идентификатор. Например яблоко – 0, груша – 1, банана – 2. Этот метод не всегда хорош, так как модель машинного обучения может воспринять такие числовые коды как упорядоченные значения, то есть предположить, что одна категория «больше» или «меньше» другой.

One-Hot Encoding - метод кодирования, при котором каждая категория представляется отдельной колонкой в наборе данных. Например, яблоко - (1,0,0), груша – (0,1,0), банан – (0, 0,1). Этот метод позволяет избежать ошибок, связанных с порядком категорий, однако если у вас 100 категорий, то вам потребуется добавить 100 столбцов, что сильно увеличит объем данных и возможно замедлит работу модели.

После кодировки из обучающей выборки выделяют 20-30% данных для тестирования. Это выборка называется **тестовой выборкой** (test set). Это аналогия с тем, как учитель в школе проводит контрольные для учеников. Он дает такие задачи, которые они раньше не видели, но похожие на те, что они решали ранее. Важно, чтобы модель никогда не видела эти данные во время обучения, так как иначе это приведет к переоценке ее реальных способностей. Вообщем выделили и отложили в сторонку.

После начинается процесс обучения модели. Схематично его можно представить следующим образом:

Схема обучения модели с учителем



Принцип работы

1. Мы подаем модели заранее подготовленные, размеченные данные (x,y).
2. Она решает какую-то определенную задачу. В данном случае на рисунке изображена задача линейной регрессии. Модель для нашего облака точек ищет наилучшую прямую изменяя коэффициенты k и b. После того как она это сделает, модель по этой построенной прямой найдет значения y для тех x, которые мы ей передали. Пример для пояснения: изначально точке x = 5 соответствовало значение y = 10. Модель построила прямую и по ней для аргумента x = 5 нашла значение y = res = 8.
3. Далее модель выдает нам результат (res), и мы вычисляем какова его ошибка (delta).
4. Если ошибка незначительная, то все ОК.
5. Если ошибка большая, корректируем модель и запускаем снова. И так да тех пор пока ее результаты не будут близки к нашим ответам.

Несколько важных уточнений

a. При первом запуске коэффициенты k и b выставляются моделью как правило случайным образом. То есть у нас может получится прямая очень далекая от нашего облака точек.

b. На схеме ошибка (delta) вычисляется как *абсолютное значение*:

$$delta = |y - res|$$

В реальной практике для оценки качества работы модели используют более сложные способы, они еще называются **функциями потерь** (lose function). Для линейной регрессии чаще всего используют функцию потерь, которая называется **метод наименьших квадратов** (Least Squares Method, МНК).

c. Для блока “Настройка модели” используют разные методы оптимизации, самым распространенный из них является – **градиентный спуск** (Gradient Descent).

d. При объяснении принципа работы метода обучения с учителем, я очень часто использовал слово МЫ, мы изменяем, мы вычисляем и тд. Но на самом деле большая часть процесса происходит автоматически:

- **Человек** подготавливает данные, выбирает модель, подбирает функцию оптимизации для настройки модели, а также оценивает и интерпретирует результаты.
- **Модель** автоматически обучается, корректируя значения k и b, вычисляет ошибку, обновляет параметры и в конечном итоге находит наилучшее решение.

Промежуточные выводы: благодаря тому, что мы разметили данные, модель получила возможность самокорректироваться, автоматически улучшая свои предсказания. Этот процесс можно описать так: на каждом этапе алгоритм пересчитывает параметры и изменяет положение прямой таким образом, чтобы она всё лучше соответствовала облаку точек. Модель постепенно "выправляет" свою линию, корректируя угол наклона и смещение, пока не найдёт наилучшее положение, которое минимизирует ошибку.

Для тех, кто не понял вот явный пример. У нас есть множество пар (объект, ответ), которые создают облако точек. Мы обучили модель строить линии, то есть передали ей аффинное уравнение $y = kx + b$. Далее мы запускаем процесс и после нескольких итераций видим вот такую вот картину (см **рисунок 2a**). Нас этот результат не устраивает, и мы опять запускаем цикл. Алгоритм продолжает корректировать значения k и b, до тех пор, пока ошибка не станет минимальной или не будет нас удовлетворять (см **рисунок 2c**).

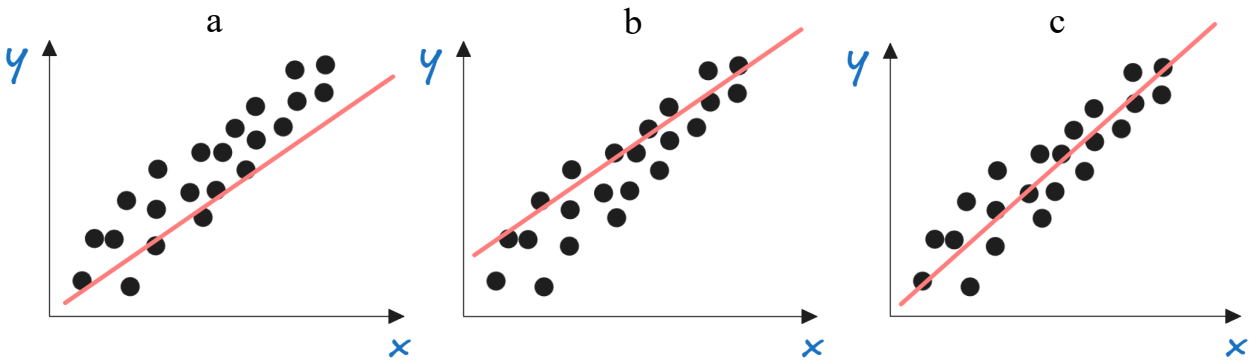


Рисунок – 2 Настройка (корректировка) модели

Давайте теперь поймем, почему какая-то прямая описывает регрессию лучше, а какая-то хуже и как это определить. Для этого еще раз взглянем на картинки 2a и 2c. Интуитивно нам ясно, что линия на картинке 2a находится где-то в стороне, а прямая на рисунке 2c непосредственно в гуще событий, но как это описать математически? Не торопитесь сразу читать ответ, постарайтесь додуматься самостоятельно.

Ответ: нам необходимо построить прямую так чтобы расстояние от каждой точки до прямой было минимальным. То есть сумма расстояний должна стремиться к минимуму.

Однако расстояние мы можем посчитать двумя способами (см **рисунок 3**). Какой нам выбрать?

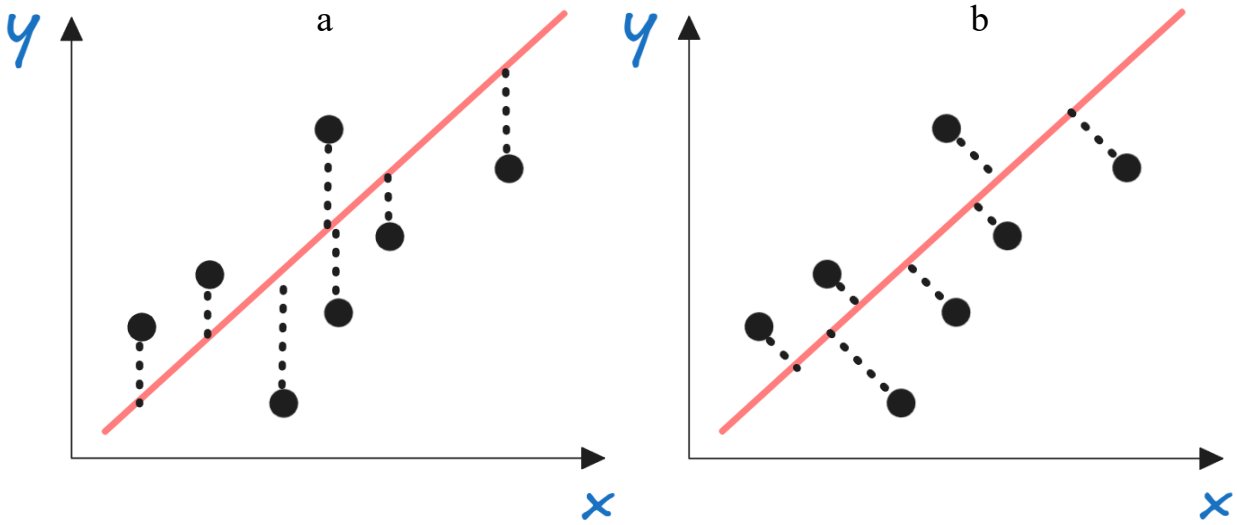


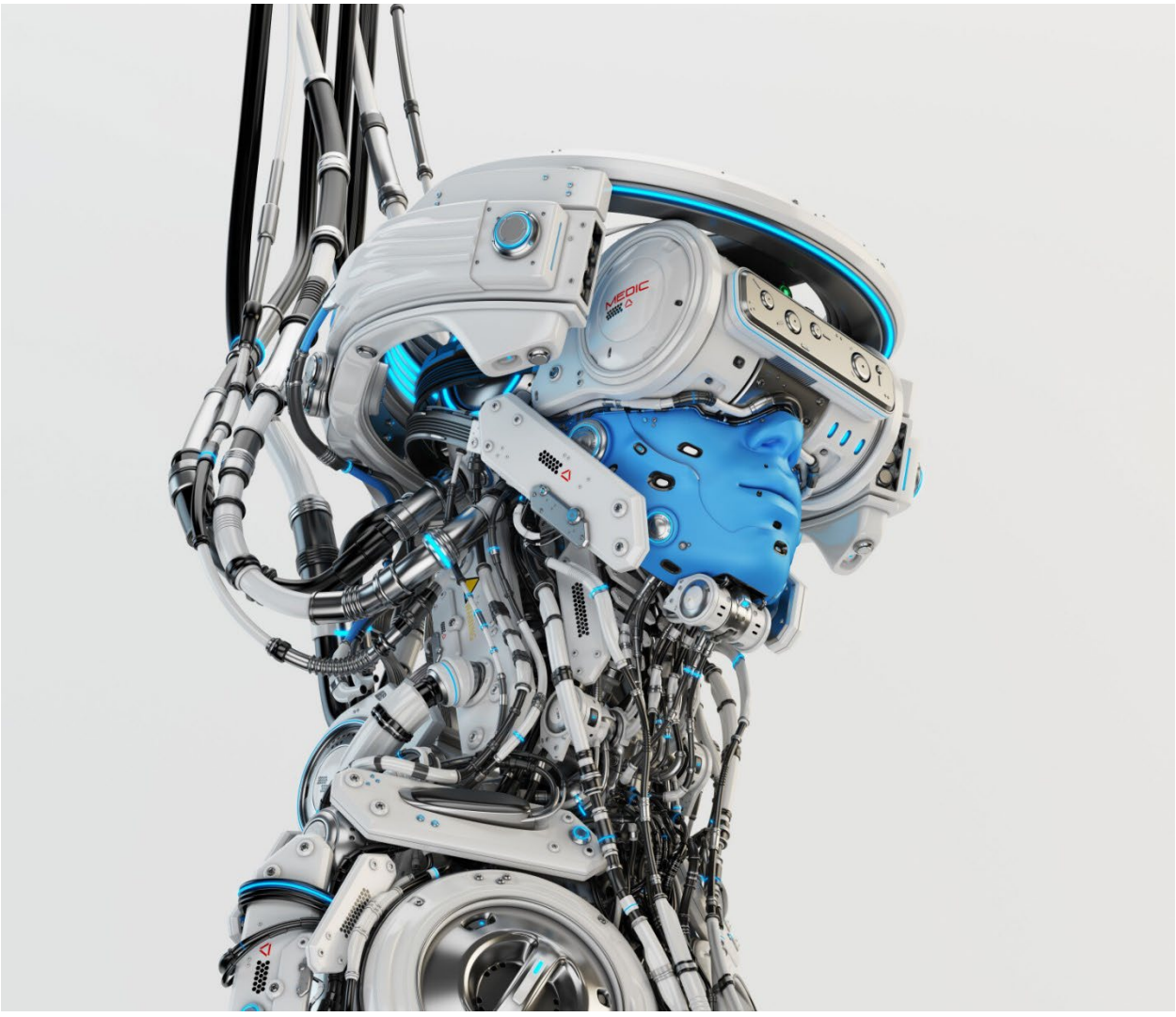
Рисунок 3 Два способа нахождения расстояний
a – вертикальный b - перпендикулярный

Два способа имеют место быть, однако запрограммировать и найти расстояние в первом случае (a) намного проще чем это сделать во втором (b). Для большей конкретики смотрим видеоролики:

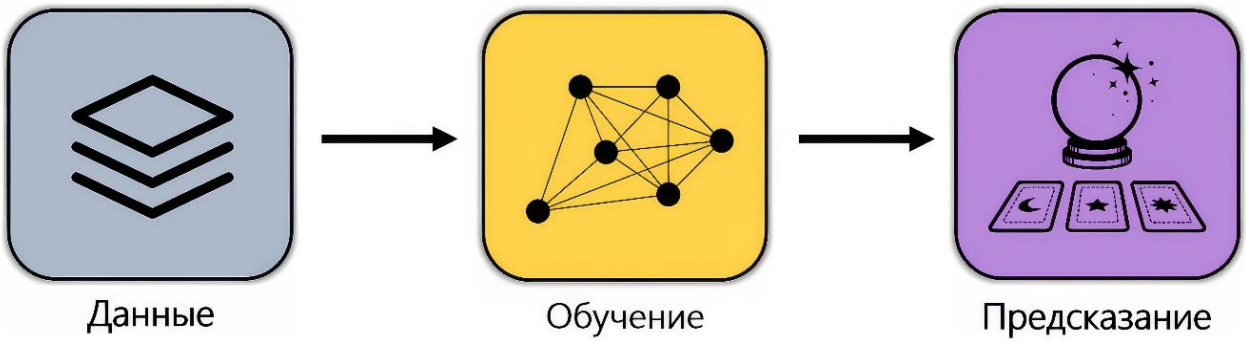
1. Как работает метод наименьших квадратов. Душкин объяснит. Ссылка: https://www.youtube.com/watch?v=KLcJX9xnhTU
2. Метод наименьших квадратов (МНК) (смотреть до 5:00, или до конца если понимаете, что такое производная). Ссылка: https://www.youtube.com/watch?v=MKJanV3BZGg

Промежуточный вывод: чаще всего ошибка (delta) в линейной регрессии вычисляется с помощью метода наименьших квадратов (МНК). Суть МНК заключается в том, что модель стремится минимизировать сумму квадратов расстояний между реальными значениями (y) и предсказанными результатами (res). Это позволяет постепенно улучшать предсказания и находить наилучшее решение для задачи.

Теперь прежде, чем выдавать итоговый вердикт о том, что модель “хорошая”, нам необходимо ее протестировать. Это финальный этап, на котором проверяется, насколько хорошо модель справляется с данными, которых она не видела раньше. Мы начинаем прогонять ее по тестовой выборке (по тем самым 20-30%, которые мы оставили в самом начале) и если все ОК, то пользуемся моделью, а если нет - опять настраиваем параметры, или вовсе выбираем другой метод обучения.



Теперь спрашивается, а зачем нам, собственно говоря, это нужно и чего мы по итогу добились? Ответ не заставит себя ждать если вы вспомните основную концепцию машинного обучения, а точнее ее последний шаг – Предсказание.



Оказывается, мы научились что-то там предсказывать, но вот что именно пока не совсем понятно. Чтобы в этом разобраться вспомним ассоциативный пример из предыдущей главы:

Регрессия – это предсказание стоимости квартиры на основе ее квадратуры, местоположения, этажа и тд. Мы хотим узнать конкретную цену покупки или продажи.

Давайте продемонстрируем как нам это удастся сделать. Выберем для начала один признак, например зависимость стоимости квартиры от расстояния до ближайшей станции метро. Пусть условно чем квартира ближе находится к метро, тем цена ее выше и соответственно, чем дальше, тем ниже.

Мы хотим продать свою квартиру не продешевив. Для этого мы прошлись по району и собрали данные о стоимостях соседних квартир. В результате чего получили несколько пар значений (расстояние до метро, цена) и изобразили их на графике (см **рисунок 4a**). Однако около нашего дома квартиры не продаются, и мы все еще так и не узнали за какую цену нам стоит ее выставлять.

Чтобы продать квартиру по дороже и не потерять в цене, мы строим модель машинного обучения и тренируем ее с помощью метода обучения с учителем. В результате чего после нескольких итераций получаем желаемый прогноз – 20$ (см **рисунок 4b**)

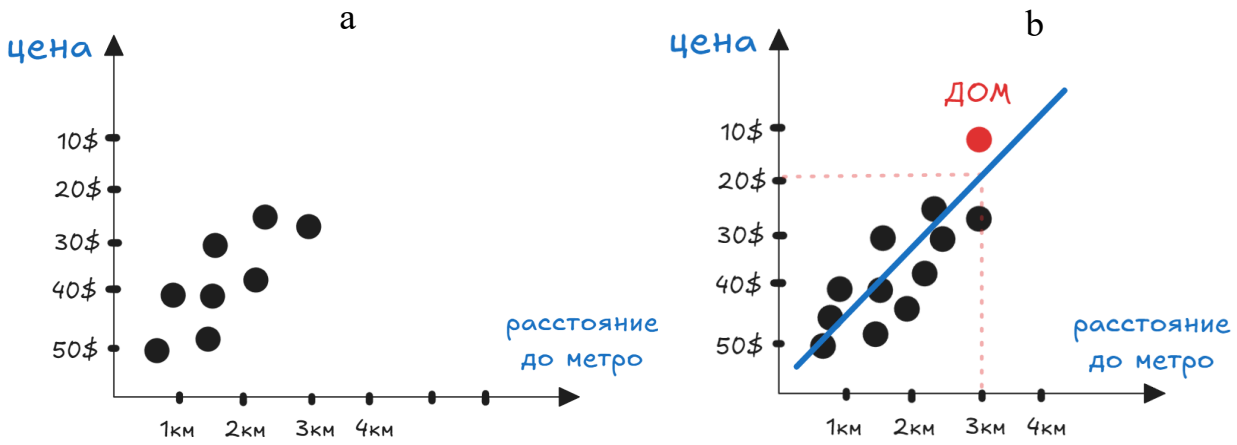


Рисунок 4 – Пример использования линейной регрессии
a – стоимость соседних квартир; **b** – прогнозирование стоимости квартиры

Конечно, данную задачу можно решить и другими способами, не прибегая к машинному обучению, однако для нас это самый простой и наглядный пример того, как происходит процесс предсказания.

Отлично, теперь после того, как мы познакомились с общей идеей, будем ее детализировать. Пока что все на чем основывалась наша модель это аффинное уравнение прямой $y = kx + b$ и по большому счету все что мы делали это наклоняли прямую или двигали ее вверх и вниз, до тех пор, пока ошибка не становилась минимальной. То есть просто изменяли k и b.

В общем случае параметры, которые мы изменяем для корректировки модели называются весами. **Вес** (weights) – это параметр, который определяет, насколько сильно тот или иной фактор влияет на результат. И его действительно в буквальном смысле можно воспринимать как что такое тяжелое, весомое, значимое.

Для того чтобы вы прочувствовали данный термин вернемся, к примеру с квартирами. В реальной жизни стоимость квартиры зависит от большого числа факторов: это и ремонт, и квадратура, и местоположение, и этаж, и расстояние до метро, и наличие парковки и много чего еще. Следовательно, нам нужна такая модель, которая будет зависеть от нескольких величин, причем влияние этих величин может быть разным. Подумайте о том, как это можно осуществить?

Если вы подумали и не догадались вот вам подсказка. Сейчас наше уравнение имеет вид y = kx + b и в нем учитывается только расстояние до метро. Допустим мы хотим учитывать еще и размер квартиры, ее квадратуру. Для этого мы прибавим к нашему уравнению еще одну переменную, умноженную на коэффициент. В результате чего получим следующую запись:

$$y = k_1 x_1 + k_2 x_2 + b$$

Регулируя параметры k_1 и k_2, мы можем изменять значимость того или иного свойства квартиры, например: цена квартиры = 100 ед., мы понимаем, что она в большей мере зависит от квадратуры (x_1) нежели чем от расстояния до метро (x_2), тогда мы можем настроить наши веса k_1 и k_2 таким образом, чтобы значимость x_1 была больше, чем x_2:

$$\begin{array}{ccccccc} y & & \mathbf{k_1} & x_1 & \mathbf{k_2} & x_2 & b \\ 100 = & & \mathbf{0{,}8} & * 90 + & \mathbf{0{,}1} & * 70 + & 21 \end{array}$$

Отлично! у нас получилось создать уравнение, которое помогает учитывать еще один параметр. Теперь посмотрим на него со стороны математики. Это $y = k_1x_1 + k_2x_2 + b$ уже никакое не уравнение прямой. Тогда что это такое мы получили?

Любители линейной алгебры уже наверно догадались, для всех остальных поясняю, это уравнение плоскости. Но как оно связанно с нашей задачей?

Важный момент: когда мы добавили в наше уравнение еще одно свойство, мы перешли из двумерного представления в трехмерное. Теперь наше облако точек находится как бы в 3D пространстве. А уравнение прямой с помощью которой мы решали задачу регрессии превратилось в уравнение плоскости. Теперь, мы будем настраивать модель путем изменения положения плоскости в пространстве, также наклоняя ее и двигая вверх и вниз.

Добавив еще одно свойство, мы перейдем в 4D, еще одно в 5D и так далее. В результате чего получим вот такое уравнение:

$$y = k_1x_1 + k_2x_2 + k_3x_3 + \ldots + k_nx_n + b$$

$$y = b + \sum_{i=1}^{n} k_i x_i$$

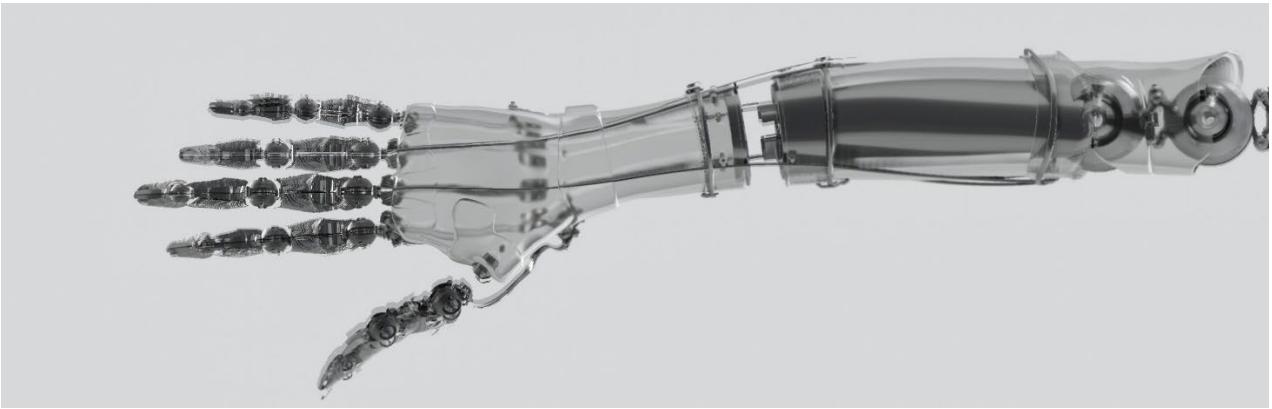
Это уравнение называется **моделью линейной регрессии** и обычно оно переписывается в такой вид:

$$a(x) = W_0 + \sum_{i=1}^{d} W_i x_i$$

где W_0 - называют баесом (сдвиг) или свободным коэффициентом, а параметры $W_1, W_2, \ldots, W_d$ – весами.

Закрепим: геометрически модель линейной регрессии можно представить как переход от уравнения линии (в случае с одной переменной) к плоскости (с двумя переменными) и, в дальнейшем, к гиперплоскости при увеличении количества признаков (фич). Каждый новый параметр добавляет дополнительное измерение в пространство, где находится наше облако точек. Алгоритм оптимизации изменяет веса этих параметров так, чтобы линия, плоскость или гиперплоскость наиболее точно "прилегали" к этому облаку. Цель заключается в том, чтобы минимизировать ошибки между предсказанными значениями и фактическими данными.

Думается мне этого вполне достаточно. Нам осталось уточнить области применения, узнать, что такое градиентный спуск, и все это перевести в программный код. Дел как вы понимаете целый вагон и маленькая тележка, но этим мы займемся в следующий раз, а пока сделаем небольшой перерыв и закрепим информацию.



ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Как вы объясните разницу между кодированием категорий через Label Encoding и One-Hot Encoding, если вам нужно решить задачу регрессии? В каких ситуациях предпочтительнее использовать один из методов?
2. Каким образом метод наименьших квадратов упрощает процесс нахождения оптимальной прямой в задаче линейной регрессии? Почему вычисление вертикальных расстояний предпочтительнее перпендикулярных?
3. Почему в процессе обучения с учителем важно, чтобы тестовая выборка оставалась "невидимой" для модели во время тренировки?
4. Как метод обучения с учителем сравним с процессом обучения в школе, и что именно играет роль "обратной связи" в машинном обучении?
5. Что происходит с моделью линейной регрессии, когда мы добавляем больше признаков (фич)? Как это отражается на ее математическом представлении и обучении?

ЗАДАЧИ НА ЛОГИКУ

Представьте, что вы обучаете модель линейной регрессии для предсказания стоимости квартир на основе двух признаков: площади и расстояния до метро. Модель показывает высокую точность на обучающей выборке, но на тестовой выборке предсказывает неправильно. Какой логический вывод можно сделать о причинах этого явления? В чём ошибка, и как её исправить?

ДОПОЛНИТЕЛЬНЫЙ МАТЕРИАЛ

1. Что такое линейная регрессия? Душкин объяснит
   Ссылка: https://www.youtube.com/watch?v=rE43fnoQNPw

2. Как устроено машинное обучение с учителем? Душкин объяснит
   Ссылка: https://www.youtube.com/watch?v=7cX-g5vlzUc