Классическое машинное обучение

Введение. Основные задачи. Линейная регрессия

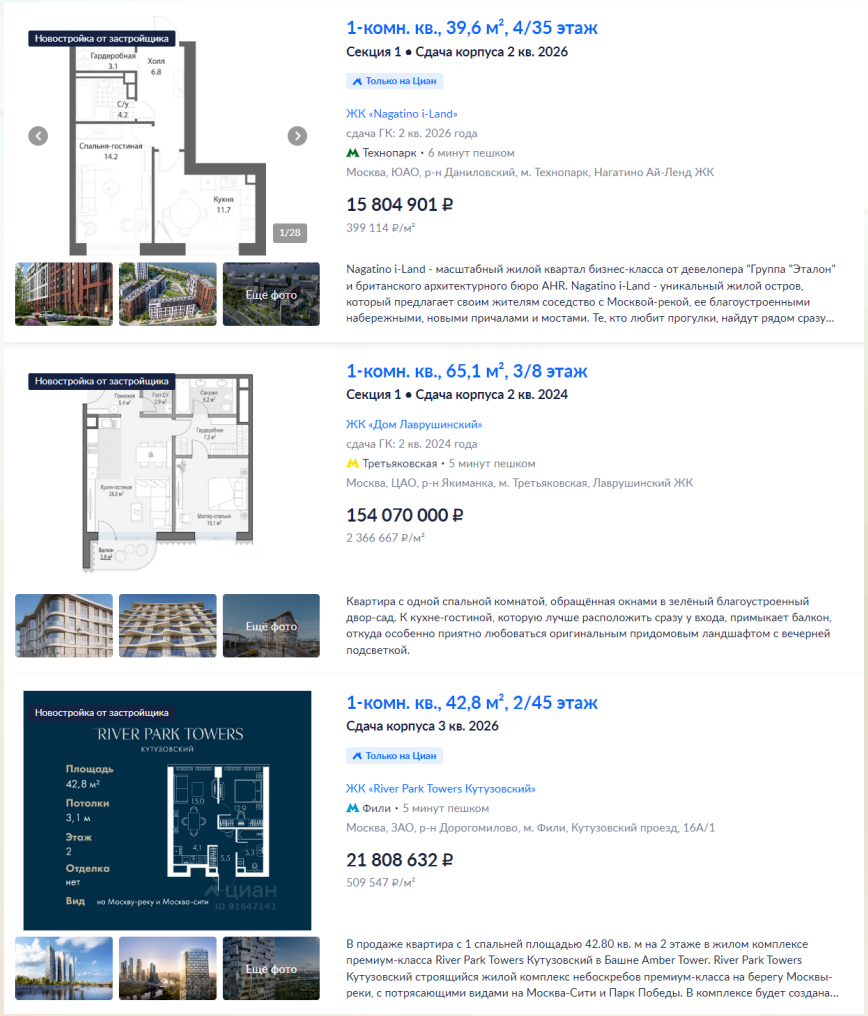
Что такое машинное обучение

- Машинное обучение – набор способов воспроизведения связей между событиями и результатом.
- Машинное обучение – обширный подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных обучаться.
- Machine learning – the field of study that gives computers the ability to learn without being explicitly programmed.

Пример: задача о квартирах

У нас есть:

- база данных по уже проданным квартирам с их характеристиками и ценами
- хотим оценить новую квартиру, о которой известны ее характеристики
- цена должна быть адекватной: не слишком низкой и не слишком высокой



Формализация задачи

X – множество объектов

Y – множество ответов

$a: X \to Y$ – неизвестная зависимость

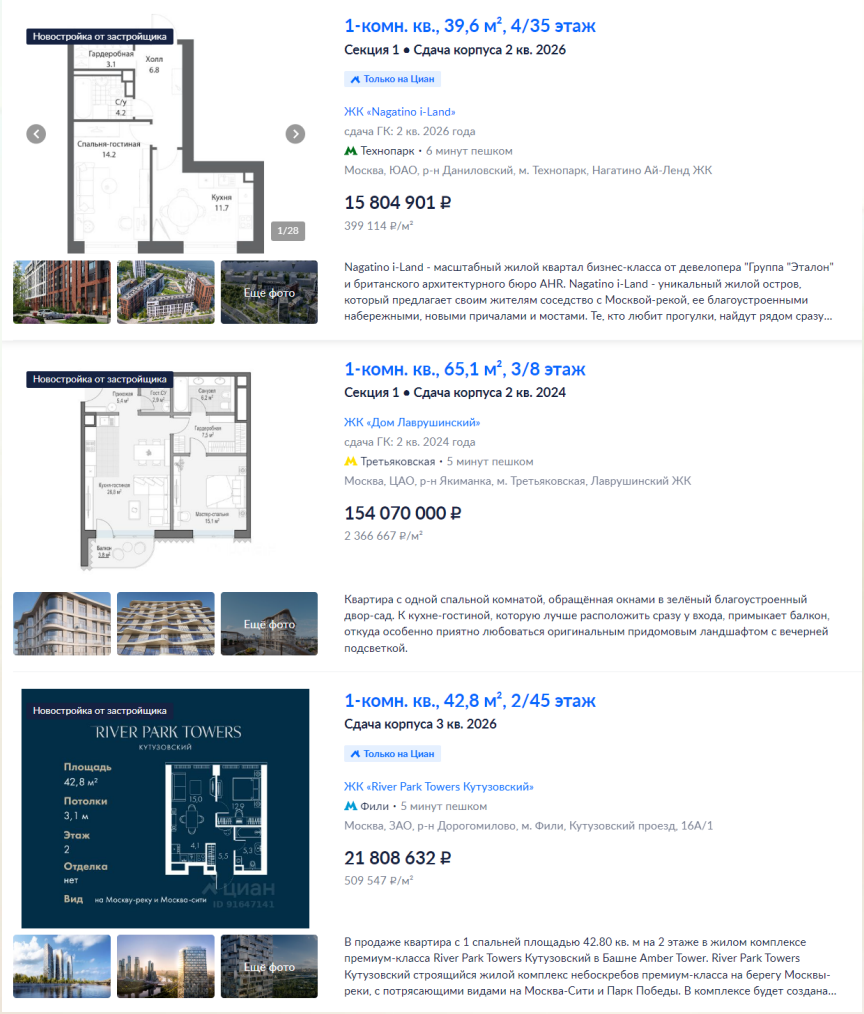
Дано:

$\{x_1, x_2, \dots, x_n\} \in X$ – обучающая выборка

$\{y_1, y_2, \dots, y_n\} \in Y$ – известные ответы

Найти:

$a: X \to Y$ – алгоритм (решающую функцию), приближающую y на всем множестве X



Признаковое описание объектов

- Признаки объекта *x* можно записать в виде вектора

$$\left(f_1(x), f_2(x), \ldots, f_n(x)\right)$$

- Матрица объекты-признаки:

$$\begin{pmatrix} f_1(x_1) & \cdots & f_n(x_1) \\ \vdots & \ddots & \vdots \\ f_1(x_m) & \cdots & f_n(x_m) \end{pmatrix}$$

Стандартная постановка задач машинного обучения

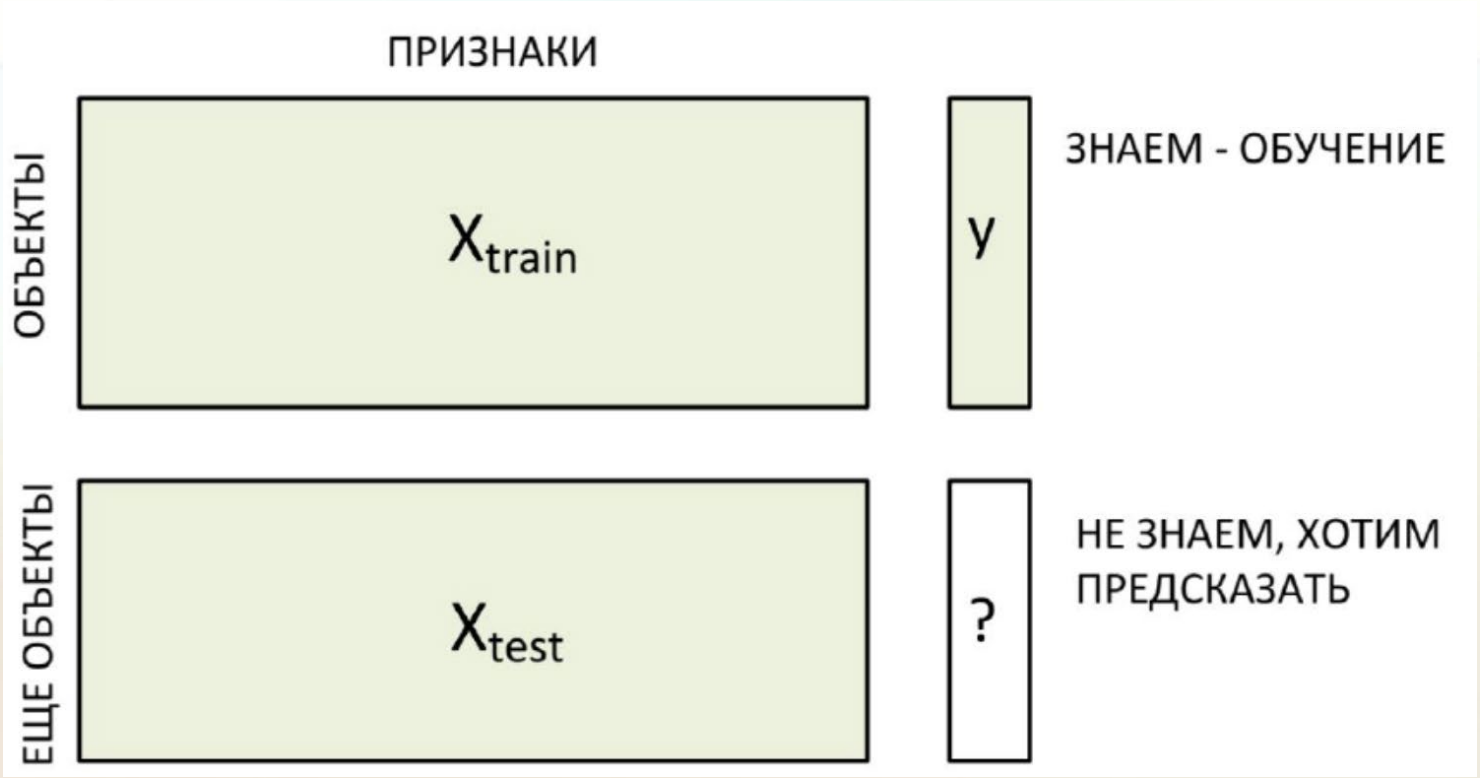


Схема получения предсказания

В задачах обучения с известными классами (обучение по прецедентам) всегда есть два этапа:

- Этап обучения (training): по выборке $X = \{(x_i, y_i)\}$ строим алгоритм *a*
- Этап применения (testing): алгоритм *a* для новых объектов выдает ответы *a(x)*

Определения

- **Признаки, факторы (features)** – количественные характеристики объекта
- **Обучающая выборка (training set)** – конечный набор объектов, для которых известны значения целевой переменной
- **Объекты** – абстрактные сущности (но компьютеры работают только с числами)
- **Признаки** описывают объекты с помощью чисел. Для их формирования обычно нужны эксперты (в нашем случае – профессиональные лингвисты)

Виды признаков

- Числовые
- Бинарные (0/1)
- Категориальные (название города, марка машины)
- Признаки со сложной внутренней структурой (изображение, текст)

Типы задач в зависимости от целевой переменной

- Классификация:
  - $Y = \{0,1\}$ - бинарная классификация
  - $Y = \{1, \dots, M\}$ - мультиклассовая (мультиномиальная) классификация
- Регрессия
  - $Y = R$ или $Y = R^n$
- Кластеризация
  - Целевая переменная отсутствует, нужно поделить данные по похожести
- Понижение размерности, визуализация данных

Виды обучения

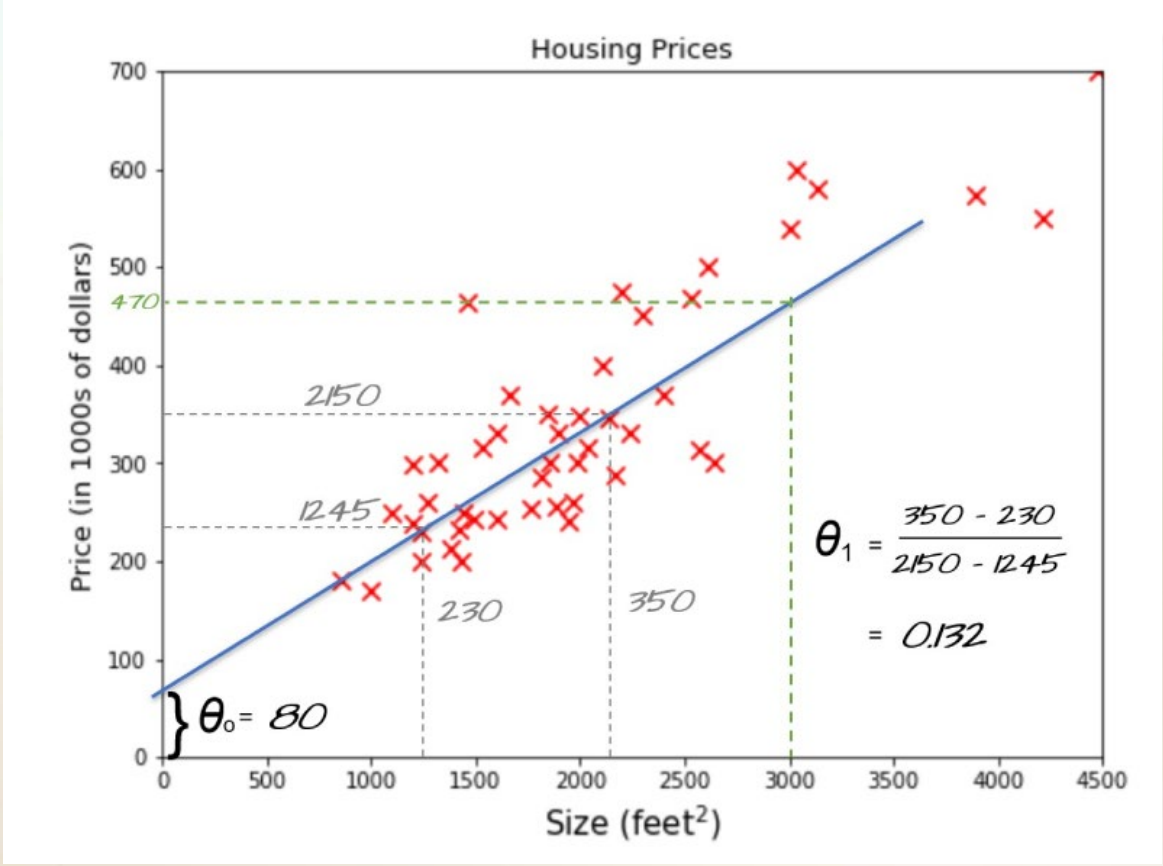
- Обучение с учителем (supervised):
  если нам известны значения целевой переменной
- Обучение без учителя (unsupervised):
  если значений целевой переменной нет
- Обучение с подкреплением (semi-supervised):
  если невозможно заранее задать все правильные значения целевой переменной (обучаем робот-пылесос, программу - игрока в го и т.п.)

Линейная регрессия

Основные понятия машинного обучения. Обучающая и валидационная выборка. Целевая переменная. Метрики, оценка качества. Функционал ошибки. Градиентный спуск.

Задача линейной регрессии

- Наши признаки:
  - площадь квартиры
  - расстояние до метро
  - этаж
  - ...
- Наша целевая переменная:
  - цена квартиры
- Наша задача:
  - построить прямую так, чтобы для наших *x* ее *y* был максимально похож на правду



Линейная регрессия: веса

- Цель: подобрать такие коэффициенты уравнения прямой, чтобы по нашим признакам можно было угадать примерный ответ (целевую переменную):

$$w_1 x_1 + w_2 x_2 + \ \ldots + w_0 = y$$

- $x_1, x_2, \ldots$ - это наши признаки (площадь квартиры, время до метро...)
- y – это целевая переменная (цена квартиры)
- $w_1, w_2, \ldots$ - это веса, или коэффициенты
- w_0 - это свободный коэффициент (шум)

Как будем учить?

1. Возьмем случайные веса
2. Посчитаем предсказанные игреки для всех известных объектов
3. Сравним с правильными ответами
4. Поправим веса, чтобы наши игреки стремились к правильным ответам
5. Вернемся к пункту 2
6. ???
7. PROFIT!

Как сравнивать игреки?

- Очевидно, нужно узнать, на сколько в среднем ошибается алгоритм, то есть:

$$\frac{1}{n}\sum_{i=1}^{n}(y_{pred} - y_{true})$$

- Что в такой формуле не нравится?

Метрики оценки качества

- MSE: $\frac{1}{n}\sum_{i=1}^{n}\left(y_{pred}-y_{true}\right)^2$ и RMSE: $\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_{pred}-y_{true}\right)^2}$
- MAE: $\frac{1}{n}\sum_{i=1}^{n}\left|y_{pred}-y_{true}\right|$
- Коэффициент детерминации: $R^2(a,X)=1-\frac{\sum_{i=1}^{l}(a(x_i)-y_i)^2}{\sum_{i=1}^{l}(y_i-\hat{y})^2}$
- MSLE
- MAPE
- SMAPE
- …

Как подобрать веса?

- Возьмем, например, MSE: очевидно, что мы хотим, чтобы он был поменьше (чем меньше MSE, тем меньше ошибка модели)
- Следовательно, нам нужно минимизировать функцию ошибки
- То есть, уравнение, которое нам нужно решить (в матричной форме):

$$\frac{1}{n}\left|\left|X_w - y\right|\right|^2 \to \min_w$$

- Это называется метод наименьших квадратов

Аналитическое решение МНК

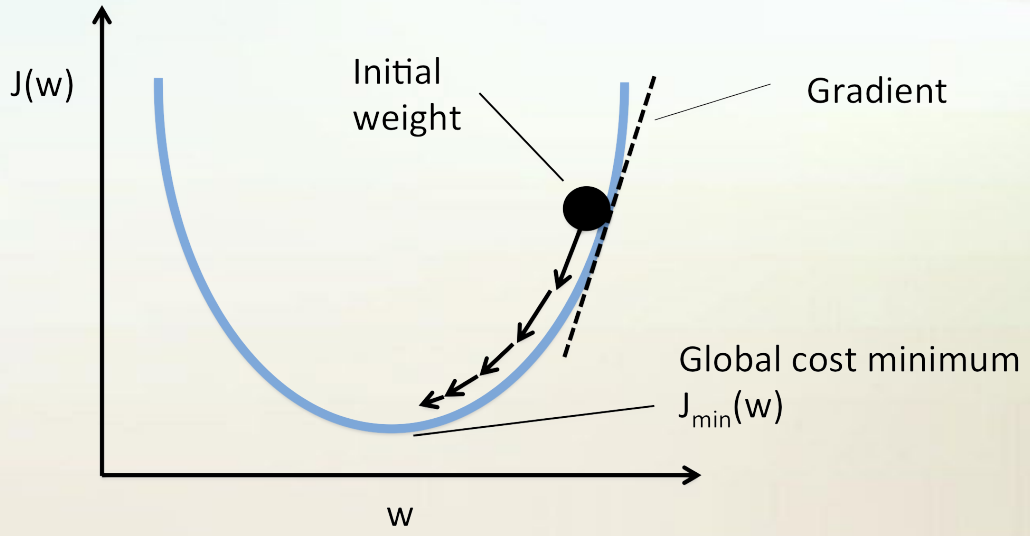
$$w = (X^T X)^{-1} X^T y$$

Недостатки:

- Обращение матрицы – сложная операция ($O(n^3)$ от числа признаков)
- Матрица $X^T X$ может быть вырожденной или плохо обусловленной
- Если функционал ошибки будет другим, можем вообще не решить задачу

Градиентный спуск

- Градиент – вектор, в направлении которого функция растет
- Антиградиент – вектор, противоположный градиенту
- Если будем двигаться в направлении антиградиента, найдем минимум
- *(Вспоминайте Лагутина)*



Градиентный спуск

- Пусть у нас только один вес w (для простоты)
- Инициализируем вес случайным числом: $w^{(0)}$
- При добавлении к весу антиградиента $-\frac{\partial Q}{\partial w}$ функция $Q(w)$ убывает.
- Вычисляем конкретные значения производной для каждого объекта в выборке
- Считаем среднее арифметическое из них
- Вычитаем из веса $w^{(0)}$
- Повторяем с начала

Градиентный спуск

- Если у нас несколько весов, то делаем это для каждого из них.
- Общая формула изменения веса:

$$w^{(k)} = w^{(k-1)} - \nabla Q(w^{(k-1)})$$

- Обычно еще добавляют коэффициент, чтобы сразу весь градиент не вычитался:

$$w^{(k)} = w^{(k-1)} - \eta \nabla Q(w^{(k-1)})$$

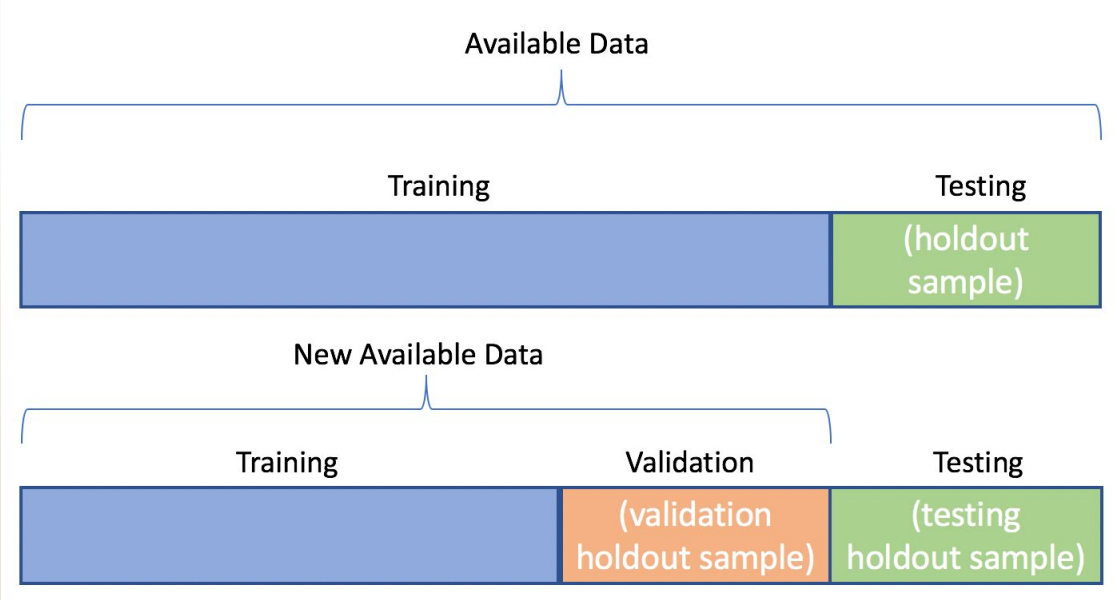
- Этот параметр называется learning rate. Мы еще много будем про него говорить на курсе по нейронкам

Как проверить качество?

- Допустим, мы обучили наш алгоритм. Как удостовериться, что он хорошо работает?
- Очевидно, считаем все те же метрики
- Метрика качества может быть такая же, как функция ошибки, а может быть другой
- Но нельзя ее считать на той выборке, на которой мы учились: это будет нечестно

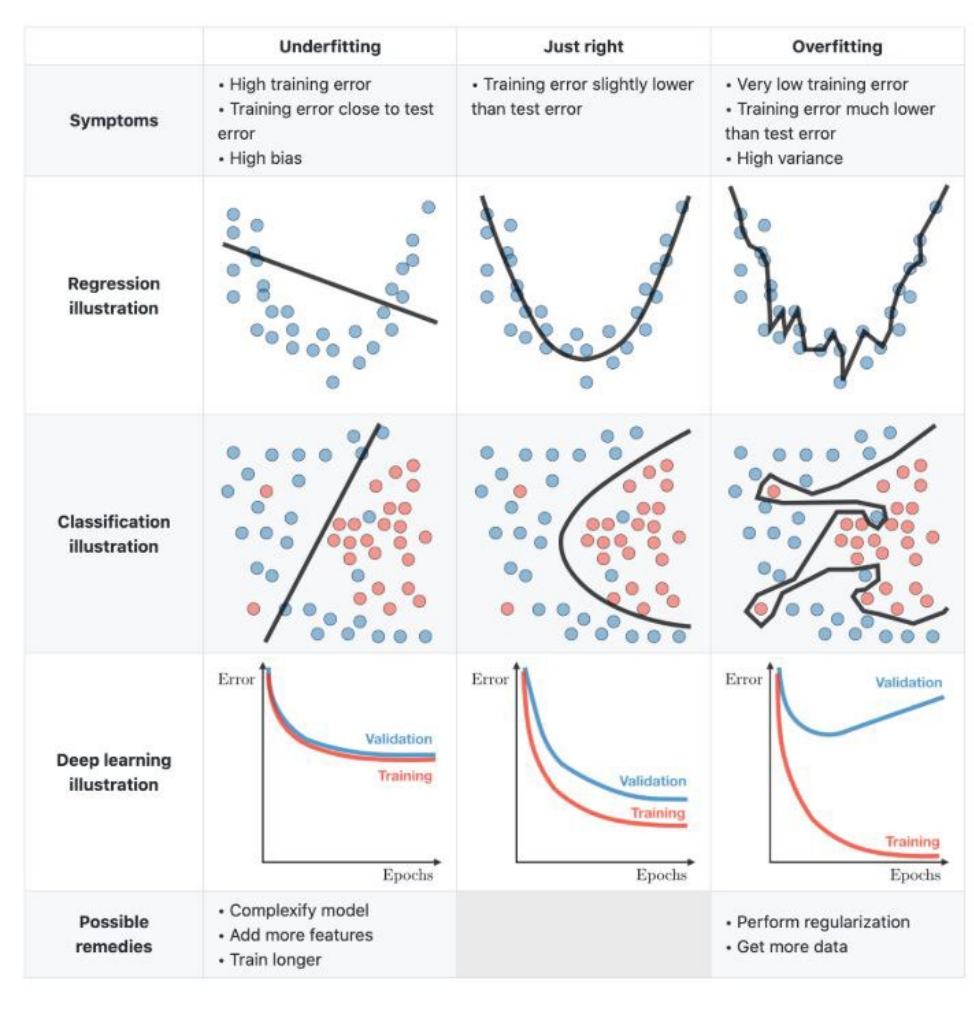
Как проверить качество?

- Следовательно, перед обучением нужно отложить какое-то количество данных, чтобы модель их не видела



Переобучение и недообучение

- В алгоритмах классического МО очень важна работа с признаками
- Если признаки линейно зависимы между собой, то высок риск переобучения
- Работа с фичами – это искусство



Наконец - практическая часть!

устанавливаем scikit learn, если еще не!

pip install scikit-learn

(conda install -c anaconda scikit-learn)