# Нейронные сети и обработка текстов

## лекция 2. Краткий ликбез по алгоритмам классического мл-я

**Иван Андреевич Ташков, 2024г.**

# Задачи

## Обучение с учителем

- Регрессия (предсказать число)
- Классификация (предсказать класс)

- Не рассматриваем: кластеризация, ранжирование, реком. системы, временные ряды и т.д.
- Рекомендованный материал для самостоятельного подробного изучения — учебник ШАД, курс DLS

# Задача регрессии

линейная регрессия

# Задача регрессии

## Пример. Предсказание цены квартиры

Две выборки: train, test.

На train-е учим модель

На test-е выбираем лучшую

(опционально — val выборка)

Модель должна не просто заучить примеры, а выучить реальную зависимость

| | квартира | Квадратные метры | Расстояние до центра (км) | Расстояние до метро (мин) | Стоимость |
|---|---|---|---|---|---|
| train | Славянская площадь | 35 | 1.2 | 2 | **46кк** |
| train | м. Прокшино | 39 | 21.8 | 16 | **10.9кк** |
| train | Ленинский проспект | 39 | 6.5 | 16 | **16.9кк** |
| train | ВДНХ | 35 | 7.5 | 6 | **15.3кк** |
| test | Крылатское | 38 | 13 | 3 | **14.5кк** |
| test | Динамо | 32 | 5.1 | 5 | **13.9кк** |

# Задача регрессии

## предсказание цены квартиры

Давайте считать на сколько сильно мы ошибаемся в цене квартиры.

Например, мы предсказали 8кк вместо 10кк — значит мы ошиблись на 2кк.
Такая ошибка называется MAE (mean absolute error).

$$MAE = \frac{1}{N} \sum |predict - price|$$

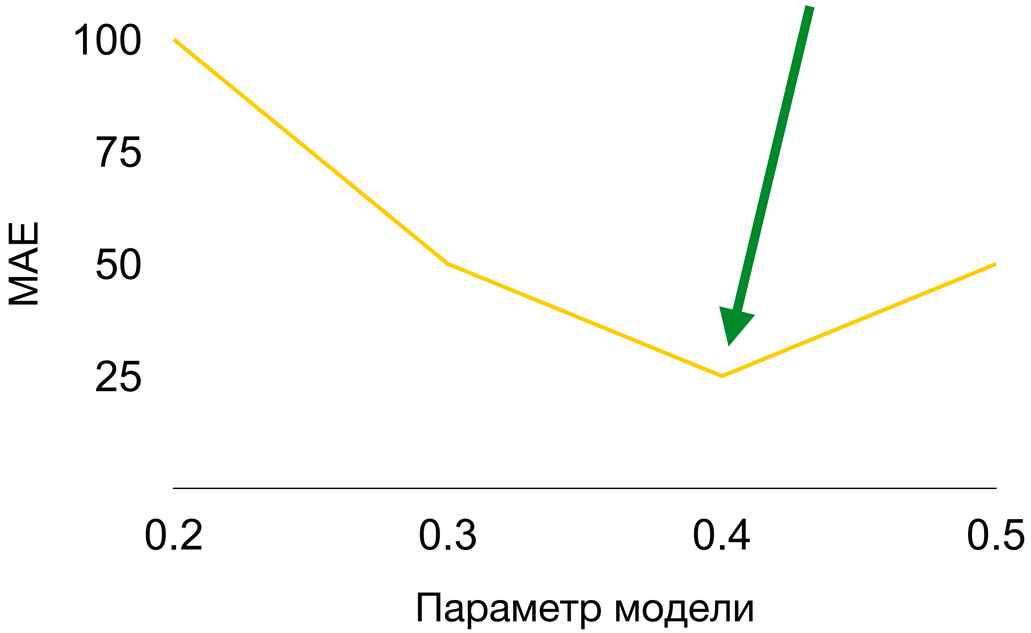
У модели справа MAE = 1.75. Чем меньше будет, тем лучше!

| | квартира | Стоимость | Предсказание | Разница предсказаний | MAE |
|---|---|---|---|---|---|
| test | Крылатское | 14.5кк | 17кк | 2.5кк | (2.5+1)/2 = 1.75 |
| test | Динамо | 13.9кк | 12.9кк | 1кк | |

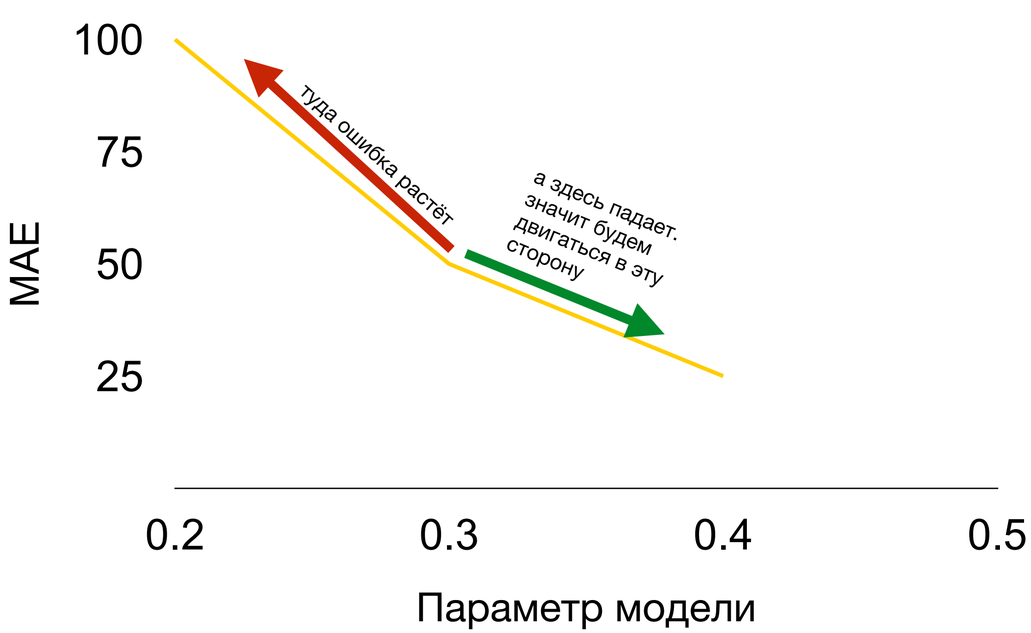
# Задача регрессии

## предсказание цены квартиры

**Идеальный вариант (мы видим, где минимум)**




**Реальный вариант (мы видим в какую сторону двигаться)**




$\omega_{t+1} = \omega_t - \alpha \nabla L$, $\alpha$ — параметр обучения

# Задача регрессии

## Лосс

$y = f_w(x)$ , $w$ — параметры, которые будем подбирать по данным

Как подобрать $w$? Нужен лосс (функция потерь), который будем минимизировать!
Например:

$$MSE(f, X, y) = L(f, X, y) = \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - y_i)^2 \to min$$

$$MAE(f, X, y) = L(f, X, y) = \frac{1}{N} \sum_{i=1}^{N} |f(x_i) - y_i| \to min$$

# Задача регрессии

## Линейная регрессия

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

$y = < w, x >$ , где $x = (1, x_1, x_2, \ldots, x_D)$

*(короткая запись для одного элемента)*

$y = Xw, X \in \mathbb{R}^{N \times D+1}$ *(короткая запись для всей выборки)*

# Задача регрессии

## Линейная регрессия

$$y = Xw, X \in \mathbb{R}^{N \times D+1}$$

$$Xw = y \;\; \rightarrow \;\; w = X^{-1}y$$

*(если $X$ — квадратная, обратимая матрица)*

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

*(если X содержит больше примеров, чем весов. псевдообратная матрица минимизирует квадратичную ошибку)*

# Задача регрессии

## Линейная регрессия

$$L(w) = \sum_{i=1}^{N} (x^T w - y_i)^2 = (Xw - y)^T (Xw - y)$$

$$\frac{\partial L}{\partial w} = 2X^T(Xw - y)$$ , *(см. подробнее матричное дифференцирование)*

$$X^T(Xw - y) = 0$$

$$w = (X^T X)^{-1} X^T y$$ *(надо, чтобы существовала обратная матрица)*

# Градиентный спуск

Пусть дана функция, хотим её минимизировать

$$f(x) \to min$$

Для функции одной переменной используем производную, для многих — градиент

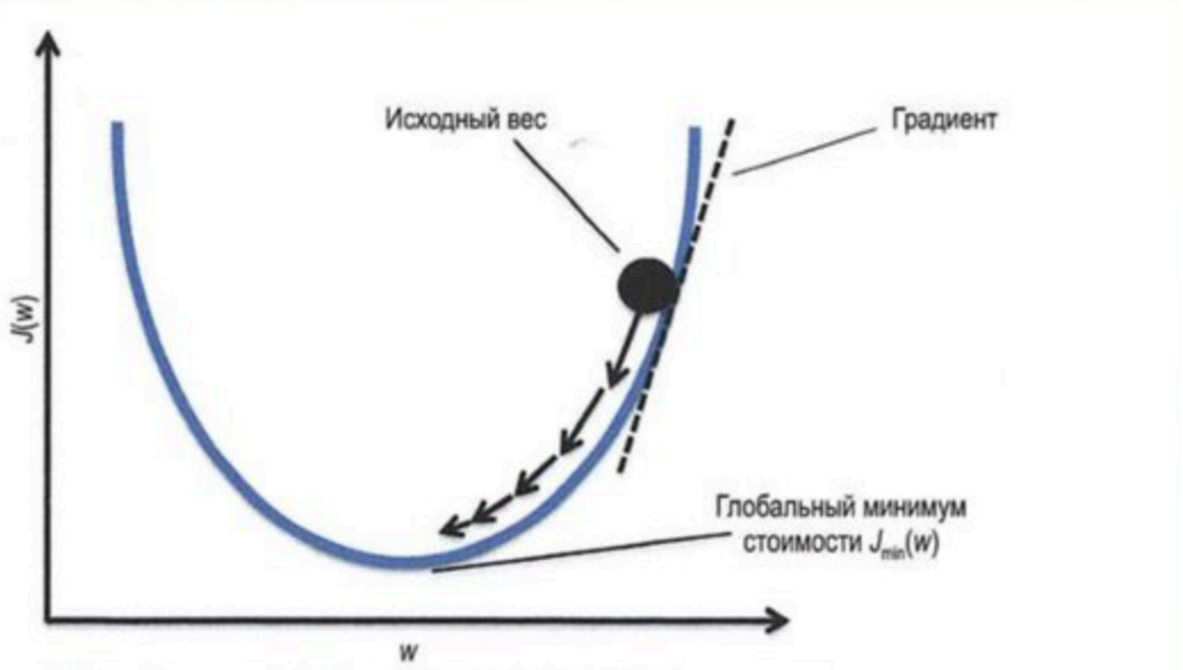
$$\nabla f(x_1, x_2, \ldots, x_n) = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})$$

# Градиентный спуск

Идём в сторону антиградиента

$$x_1 = x_1 - \alpha\frac{\partial f}{\partial x_1}, \dots, x_n = x_n - \alpha\frac{\partial f}{\partial x_n}$$

короткая запись: $x = x - \alpha\nabla f$

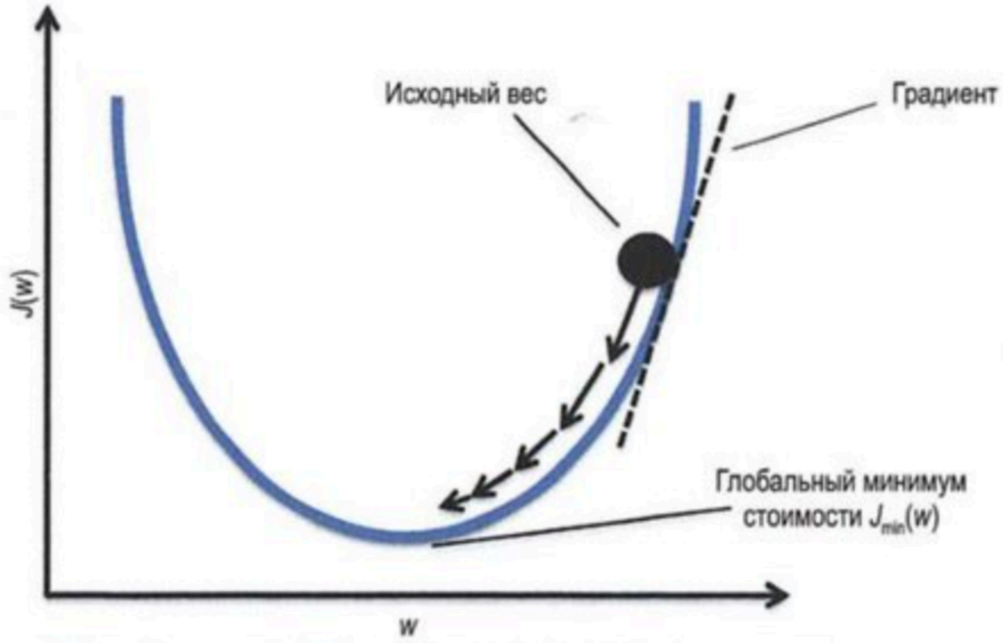


# Градиентный спуск
## пошагово

1) инициализируем $x_1, \ldots x_n$

2) делаем шаг обновления весов: $x = x - \alpha \nabla f$

3) повторяем много раз (либо когда шаг станет eps)



# Градиентный спуск

Функция потерь для MSE в линейной регрессии

$$L(w) = \sum_{i=1}^{N} (x_i^T w - y_i)^2$$

В качестве примера возьмём производную по первому аргументу

$$\frac{\partial L}{\partial w_1} = \frac{\partial (\sum_{i=1}^{N} (x_i^T w - y_i)^2)}{\partial w_1} = \sum_{i=1}^{N} \frac{\partial (x_i^T w - y_i)^2}{\partial w_1}$$

# Градиентный спуск

## стохастический градиентный спуск

$$\frac{\partial L}{\partial w_1} = \frac{\partial(\sum_{i=1}^{N}(x_i^T w - y_i)^2)}{\partial w_1} = \sum_{i=1}^{N}\frac{\partial(x_i^T w - y_i)^2}{\partial w_1}$$

$$\frac{\partial L}{\partial w_1} = \sum_{i=1}^{N} 2(x_i^T w - y_i)\frac{\partial x_i^T w}{\partial w_1}$$

$$\frac{\partial L}{\partial w_1} = \sum_{i=1}^{N} 2(x_i^T w - y_i)x_{i1}$$

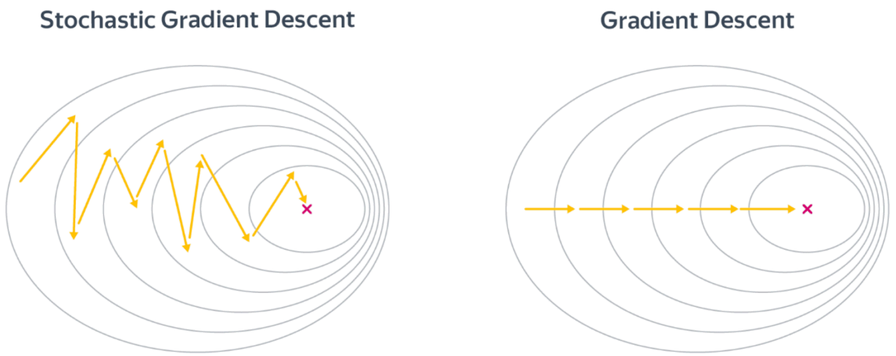
теперь мы можем сделать шаг в сторону убывания производной (антиградиента)!

# Градиентный спуск
## стохастический градиентный спуск

Считать производную по всей выборке вычислительно дорого

Давайте использовать случайную подвыборку (батч)

Одна эпоха обучения алгоритма — один проход по всем батчам из обучающей выборки



# Регуляризация

Является ли решение единственным?

Допустим, что у нас есть два признака тождественно равные друг другу

Тогда $w_i = 1,\ w_{i+1} = 0$
и $w_i = 2,\ w_{i+1} = -1$
будут занулять друг друга. Решение не единственно

# Регуляризация

Мультиколлинеарность — один признак приближённо выражается через линейную комбинацию остальных

Веса $w$ становятся нестабильными

Решение: регуляризация (ограничение на веса)

$$L^1 = \lambda \sum |w|$$
$$L^2 = \lambda \sum w^2$$

# Регуляризация

Итого минимизируем функцию потерь, например, с L2 регуляризацией

$$L(w) = \sum_{i=1}^{N} (x_i^T w - y_i)^2 + \lambda \sum w^2$$

$\lambda \sum w^2$ — регуляризационный член

Вопрос: в $L^1$ регуляризационный член недифференцируем в точке 0. Как же нам тогда использовать град. спуск?

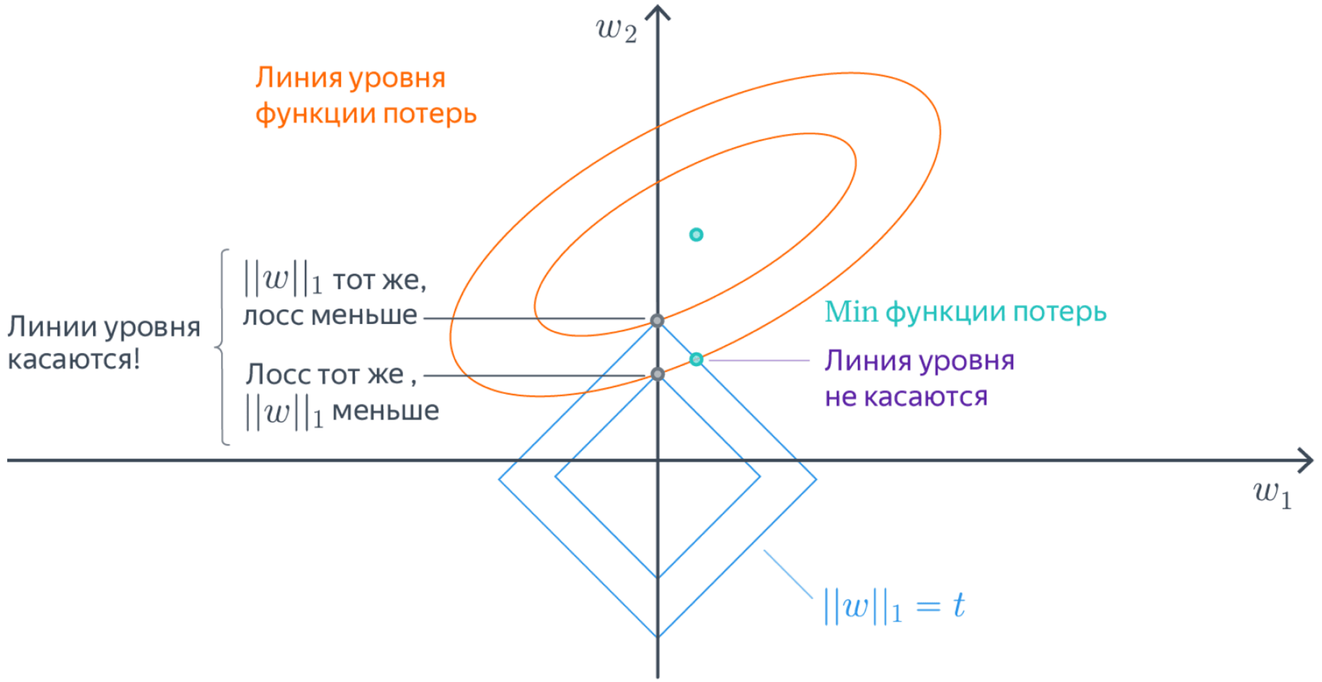
# Регуляризация

## Зануление весов

$$L(w) = \sum_{i=1}^{N} (x_i^T w - y_i)^2 + \lambda \sum |w|$$

$\sum_{i=1}^{N} (x_i^T w - y_i)^2$ — эллипс (2-мерный случай)

$\lambda \sum |w|$ — ромб

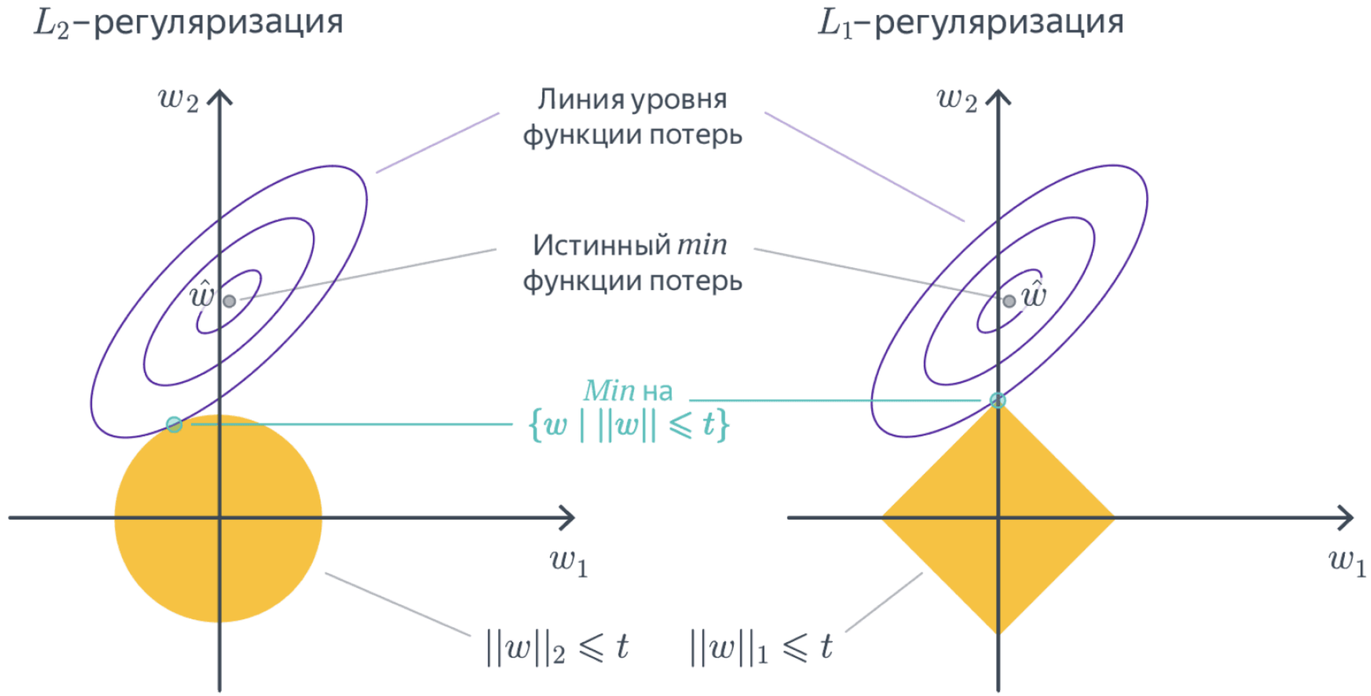


# Регуляризация

## Зануление весов

$\lambda \sum |w|$ — ромб ($L^1$)

$\lambda \sum w^2$ — окружность ($L^2$)



# Нормализация

$$x_{new} = \frac{x - \mu}{\sigma}$$

$\mu$ — выборочное среднее

$\sigma$ — стандартное отклонение

1. Град спуск лучше работает с нормированными данными, чем с огромными или очень маленькими числами
2. Веса становятся интерпретируемыми
3. Регуляризация "штрафует" все признаки с одинаковой силой

# Признаки

Вопрос: как учитывать зависимости двух признаков между собой?

Вопрос: почему мы не рассматриваем полиномиальную регрессию ($x^2, x^3$ и т.д.)

# Задача классификации

логистическая регрессия

# Задача классификации

$Y \in \mathbb{R}$ — задача регрессии

$Y \in \{0,1\}$ — задача классификации

Давайте предсказывать кто изображён на фото:
кошечка или собачка ($Y \in \{cat, dog\}$)

**Результат генерации DALL-E 3**

подводка: "Generate 10 images that are lying on the table in a chaotic way. Each of the image shows either a kitty or a puppy"

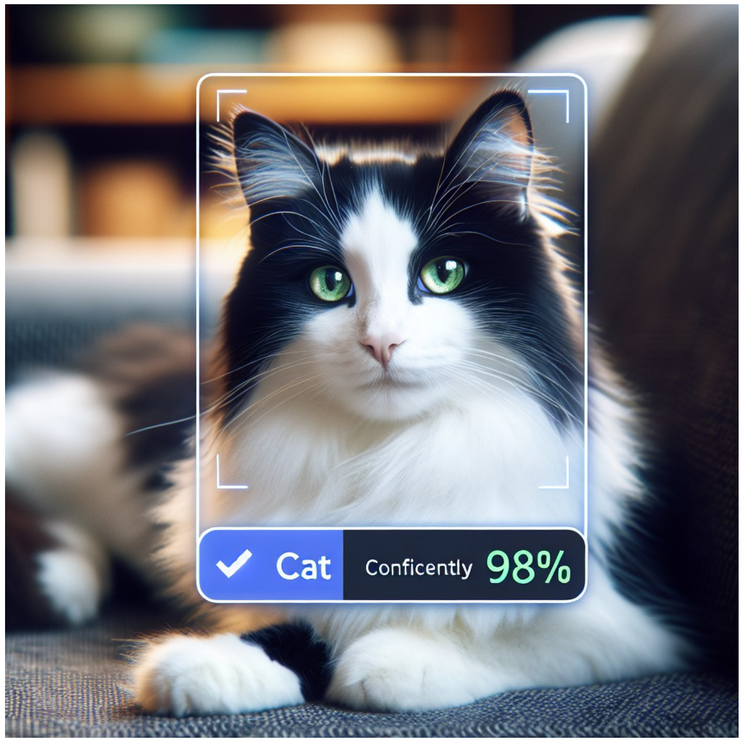
# Логистическая регрессия

Линейная регрессия предсказывает значения от $-\infty$ до $+\infty$

Будем считать, что лин. регрессия выдаёт на выходе $log(\frac{p}{1-p})$ , где

$p$ — вероятность первого класса (например, вероятность кошечки)

$1-p$ — вероятность нулевого класса (собачка)




подводка: "Generate an image where automatic ML algorithm identifies a cat on photo. This image should be contain probability that the photo shows cat not a dog computed by ML algorithm"

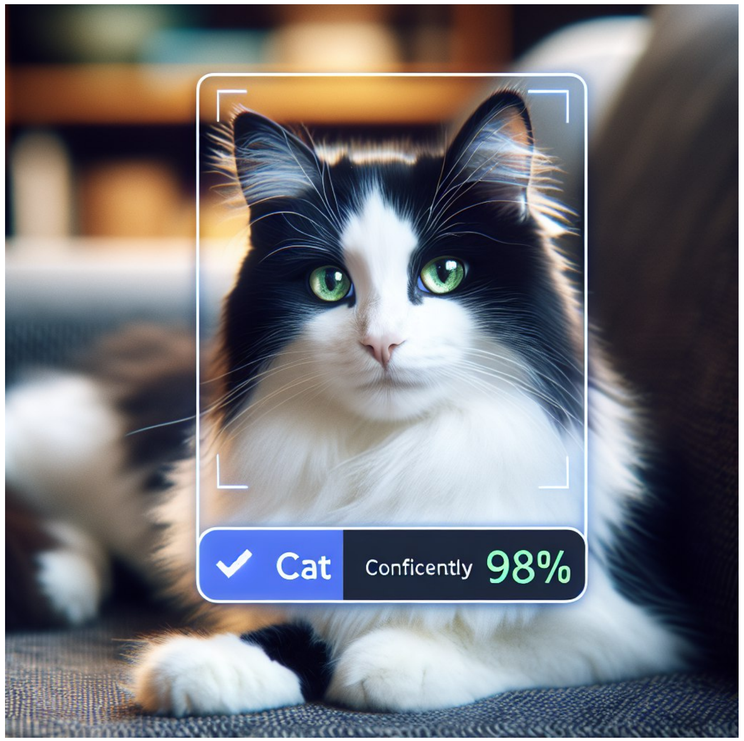
# Логистическая регрессия

$$< w, x_i > = log(\frac{p}{1-p})$$

$$e^{<w,x_i>} = \frac{p}{1-p}$$

$$p = \frac{1}{1 + e^{-<w,x_i>}}$$

Ф-ия сигмоиды: $\sigma(z) = \frac{1}{1 + e^{-<w,x_i>}}$




подводка: "Generate an image where automatic ML algorithm identifies a cat on photo. This image should be contain probability that the photo shows cat not a dog computed by ML algorithm"

# Логистическая регрессия

## Лосс

$$L = logloss = -\frac{1}{N}\sum_{i}(y_i * log(p_i) + (1 - y_i) * log(1 - p_i))$$

только одно из двух слагаемых внутри суммы ненулевое

Большие ошибки сильно штрафуются

подводка: "Generate a young slavic school student who is taught by a teacher how to distinguish cats vs dogs on examples"

# Логлосс
## Вывод формулы

Метод максимального правдоподобия

На сколько вероятно получить данные значения $y$ (кошечки или собачки), при заданных $X$ (факторы) и весах $w$

$$P(y \mid X, w) \rightarrow max$$

Выборка у нас фиксирована ($y$ и $X$)
Зато мы можем изменять w!

# Логлосс

## Вывод формулы

$$P(y \mid X, w) = \prod_i P(y_i \mid x_i, w) = \{\text{распр. Бернулли}\} = \prod_i p_i^{y_i}(1 - p_i)^{1-y_i}$$

Монотонное преобразование, чтобы работать с суммами

$$log(\prod_i P(y_i \mid x_i, w)) = \sum_i (y_i log(p_i) + (1 - y_i) log(1 - p_i)) \to max$$

Тогда $(-1) \cdot \sum_i (y_i \cdot log(p_i) + (1 - y_i) \cdot log(1 - p_i)) \to min$

Поделим ещё на N (кол-во элементов в обучающей выборке) и получим нужный логлосс

# Логлосс

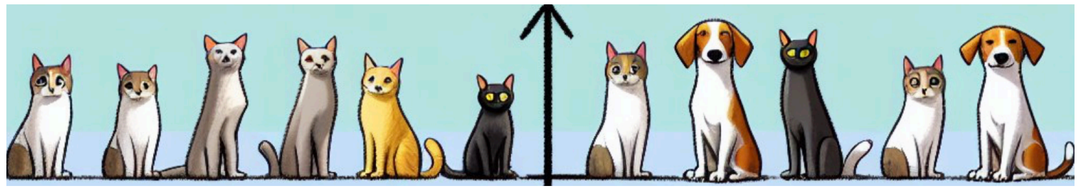
## Градиент

$$L = logloss = -\frac{1}{N}\sum_i (y_i * log(p_i) + (1 - y_i) * log(1 - p_i))$$

Остаётся вычислить градиент и двигать параметры в сторону антиградиента

$$\nabla L(w, X, y) = -\sum_i x_i(y_i - \sigma(< w, x_i >))$$

Вопрос: какое значение вероятности считать достаточным (пороговым), чтобы ответить, что на фото изображена кошечка?



# Многоклассовая логистическая регрессия

## Градиент

$$Y \in \{cat, dog, otter, horse \dots\}$$

На каждый класс делаем свою линейную модель. Их результаты переводим в вероятности

$$softmax(z_1, \dots, z_j, \dots, z_k)_j = \frac{e^{z_j}}{\sum_k e^{z_k}} = \frac{e^{<x,w_j>}}{\sum_k e^{<x,w_k>}}$$

Каждая из компонент больше 0. Сумма вероятностей = 1.

Аналогично с помощью ММП получим лосс — кроссэнтропию

$$\frac{-1}{N} \cdot \sum_i (y_{true\_i} \cdot log(p_{true\_i})) \to min$$

# Метрики

# Метрики

Метрика — показатель качества работы алгоритма на задаче

Простейший пример метрики классификации — accuracy (точность = доля правильно угаданных классов)

| | | | | | |
|---|---|---|---|---|---|
| **Правильный ответ** | кошка | собака | кошка | собака | собака |
| **Предсказание алгоритма** | собака | собака | кошка | собака | кошка |

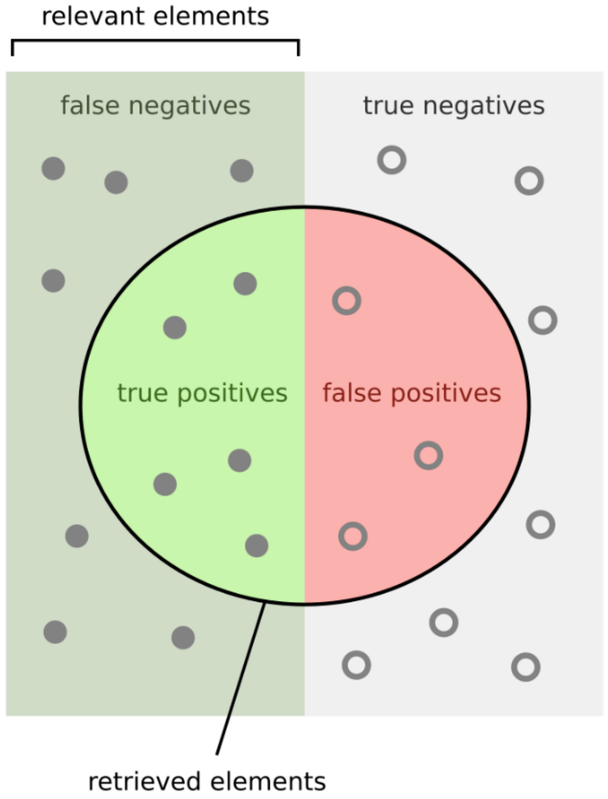
accuracy = 3/5

# Метрики

Accuracy — не учитывает дисбаланс классов

Пытаемся найти всех кошечек!
True positive, False positive, True negative, False negative

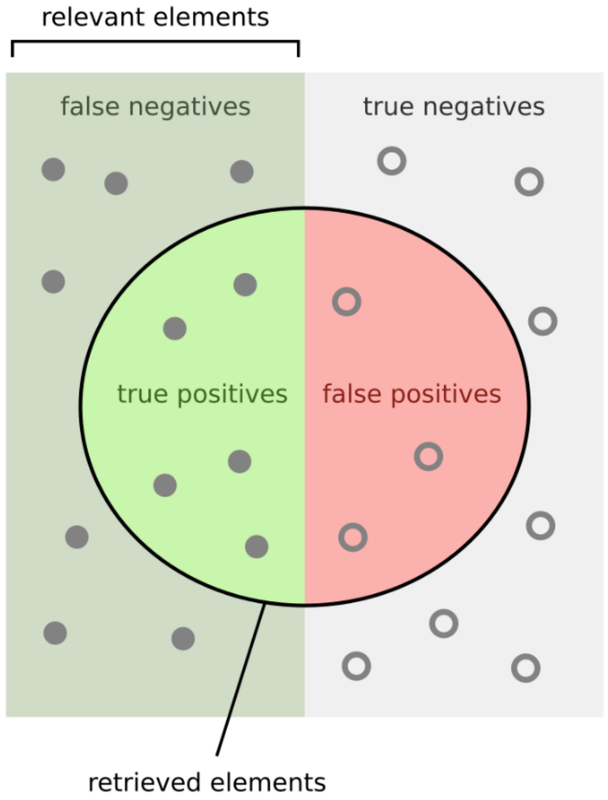
TP — кошечки, которых мы нашли
FP — собачки, которых мы посчитали кошками
TN — собачки, которых мы посчитали собачками
FN — кошечки, которых мы не нашли



# Метрики

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$




Precision — доля правильно предсказанных кошечек, среди найденных

Recall — доля найденных кошечек, среди всех существующих

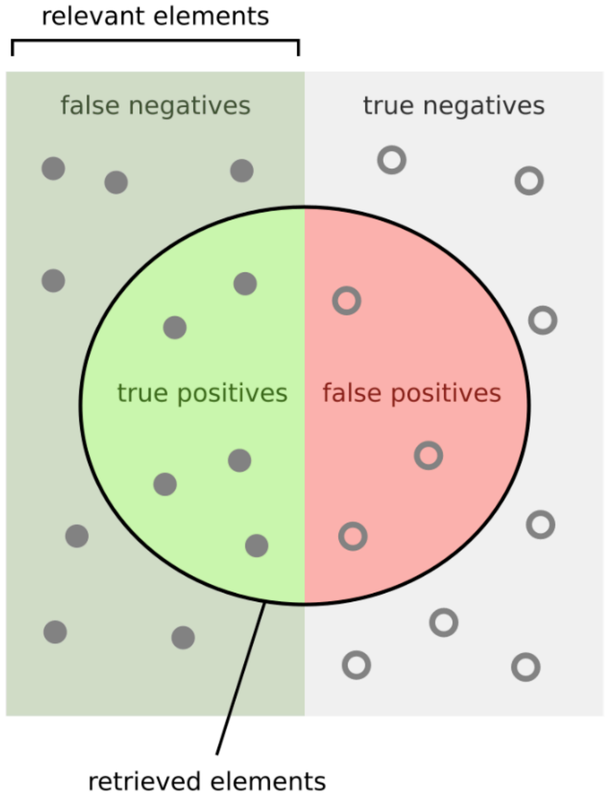
# Метрики

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

Среднее гармоническое между Precision и Recall

Вопрос: почему бы не взять среднее арифметическое?

$$F_{beta} = (1 + \beta^2)\frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall}$$



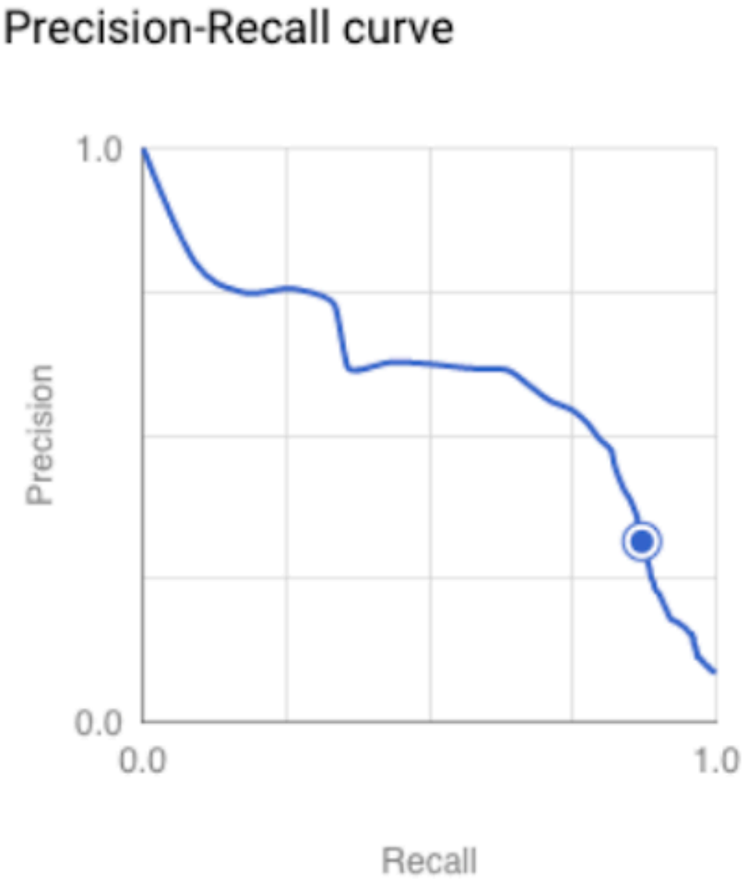
# Метрики
## Precision recall curve

Precision recall curve

Последовательно двигаем порог и считаем Precision, Recall

Precision at recall — значение precision при фиксированном recall (например, 0.9)

Recall at precision — значение recall при фиксированном precision (например, 0.8)



# Метрики
## area under curve

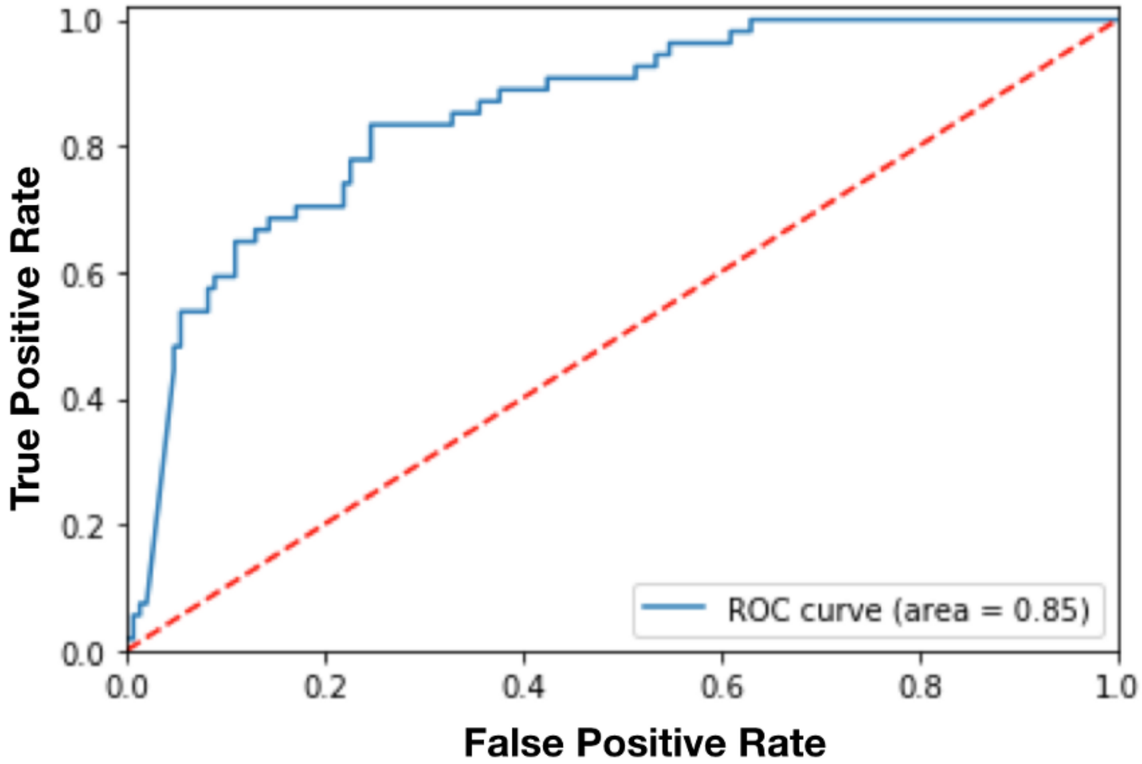
$$TPF = \frac{TP}{TP + FN}$$ (true positive rate)

$$FPR = \frac{FP}{TN + FN}$$ (false positive rate)

Двигаем порог и считаем TPR и FPR по новой

AUC — площадь под ROC кривой и принимает значения от 0 до 1

Любое значение AUC ниже 0.5 — хуже случайного предсказания



# Метрики
## регрессия

Mean squared error, Root mean squared error, Mean absolute error

$$MSE(y_{true}, y_{pred}) = \frac{1}{N}\sum_{i=1}^{N}(f(x_i) - y_i)^2$$

$$RMSE(y_{true}, y_{pred}) = \sqrt{MSE(f, X, y)}$$ (для совпадений размерностей)

$$MAE(y_{true}, y_{pred}) = \frac{1}{N}\sum_{i=1}^{N}|f(x_i) - y_i|$$

$$MAPE(y_{true}, y_{pred}) = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - f(x_i)|}{|y_i|}$$

# Спасибо!