

Градиентный спуск. Логистическая регрессия

Маша Шеянова, masha.shejanova@gmail.com

Немного матанализа

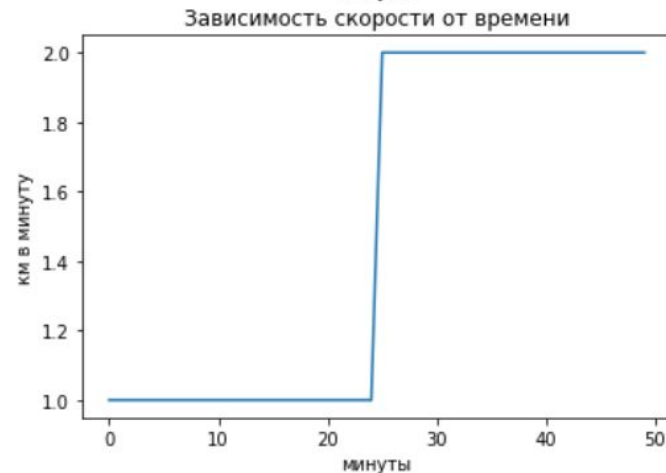
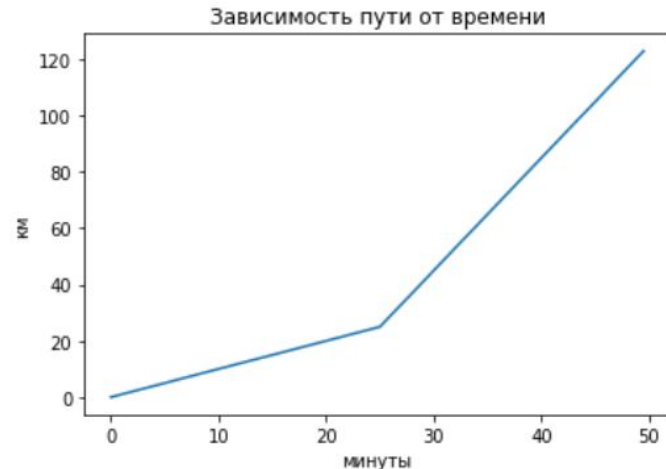
Что такое производная?

Мера, насколько быстро растёт функция.

Например, вот график зависимости пройденного пути от времени:

А вот — зависимость скорости от времени для того же случая:

Функция на втором графике — производная от функции на первом. Она показывает, что первые 25 минут скорость была 1 км в минуту, а потом удвоилась.

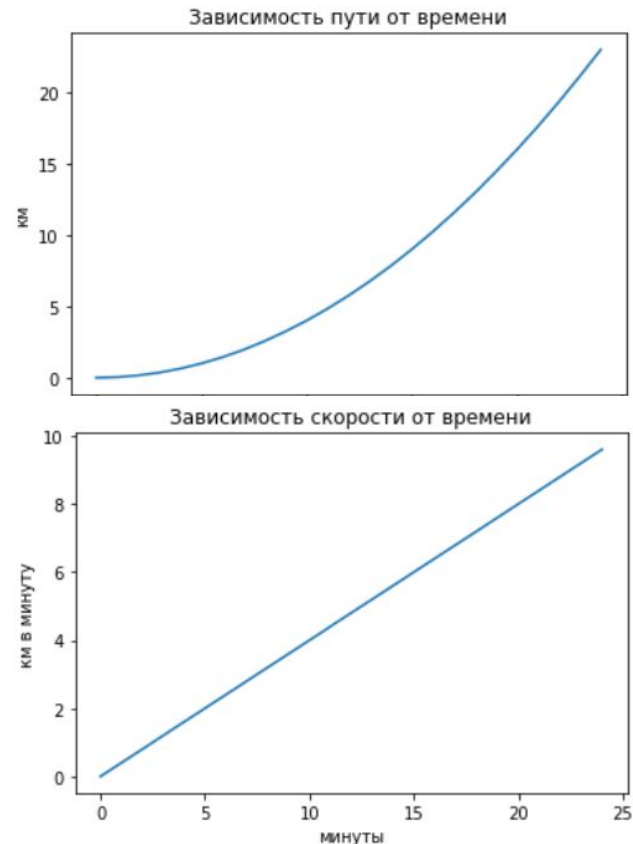


Интуитивное понимание производной

А вот в этом примере зависимость пути от времени нелинейна — скорость меняется в каждой точке.

Что отражает производная? Насколько скорость увеличилась **в каждой отдельной точке** графика сверху.

Но как посчитать это изменение в скорости?



Как посчитать производную?

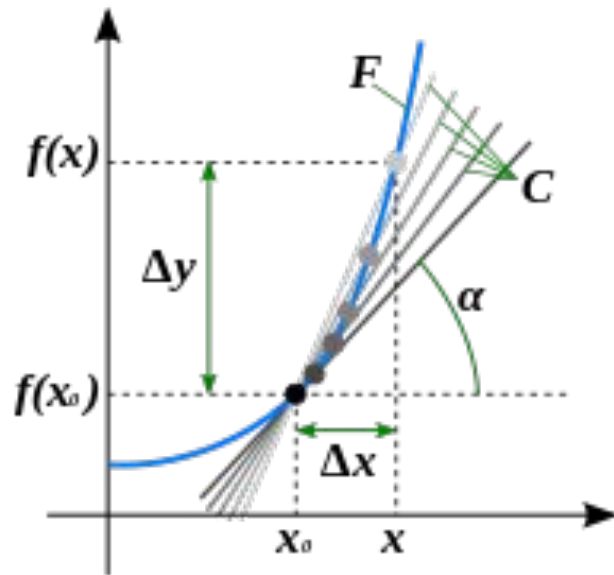
Буквально, надо понять как сильно от небольшого изменения в x поменялся y .

$$\Delta x = x - x_0 \quad \Delta f = f(x) - f(x_0)$$

Причём это соотношение надо посчитать на бесконечно малом промежутке:

$$f'(x_0) = \lim_{\Delta x \rightarrow 0} \frac{\Delta f}{\Delta x}$$

Другие способы обозначать производную: $f'(x_0) = f'_x(x_0) = Df(x_0) = \frac{df}{dx}(x_0)$



Как работают пределы

Предел функции (*lim*), интуитивно — это значение, к которому стремится функция, когда её аргумент приближается к определённой точке.

Формальные определения.

Чаще всего нам интересно $x \rightarrow +/\infty$ или $x \rightarrow 0$. В производной $\Delta x \rightarrow 0$.

Несколько полезных правил:

- предел константы = константа
- если мы пытаемся найти предел дроби, можно спокойно сокращать одинаковые множители из числителя и знаменателя

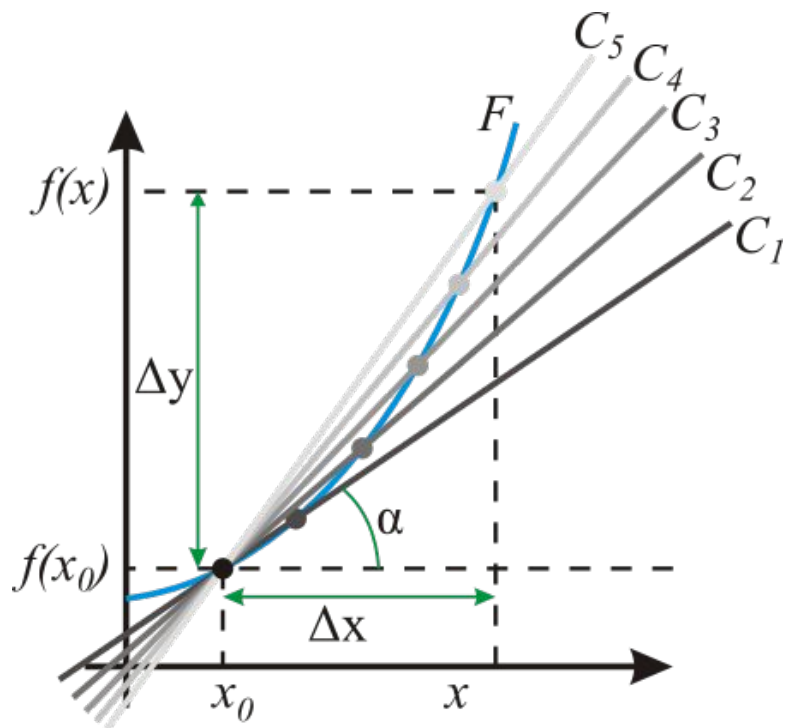
Потренируемся считать производные

Допустим, есть такая функция: $f(x) = ax + b$. Чему равна её производная?
Воспользуемся определением производной.

$$\begin{aligned} f'(x) &= \lim_{\Delta x \rightarrow 0} \Delta f / \Delta x = \\ &= \lim_{\Delta x \rightarrow 0} (f(x_1) - f(x_0)) / (x_1 - x_0) = && \# \text{ раскроем } \Delta f \text{ и } \Delta x \\ &= \lim_{\Delta x \rightarrow 0} (ax_1 + b - (ax_0 + b)) / (x_1 - x_0) = && \# \text{ подставим значения функции} \\ &= \lim_{\Delta x \rightarrow 0} (ax_1 - ax_0) / (x_1 - x_0) = \lim_{\Delta x \rightarrow 0} a * (x_1 - x_0) / (x_1 - x_0) = && \# \text{ сократим числитель} \\ &= \lim_{\Delta x \rightarrow 0} a = a && \# \text{ разделим обе части на } x_1 - x_0 \end{aligned}$$

Давайте теперь посчитаем производную функции $f(x) = ax^2 + b$.

Производная в точке

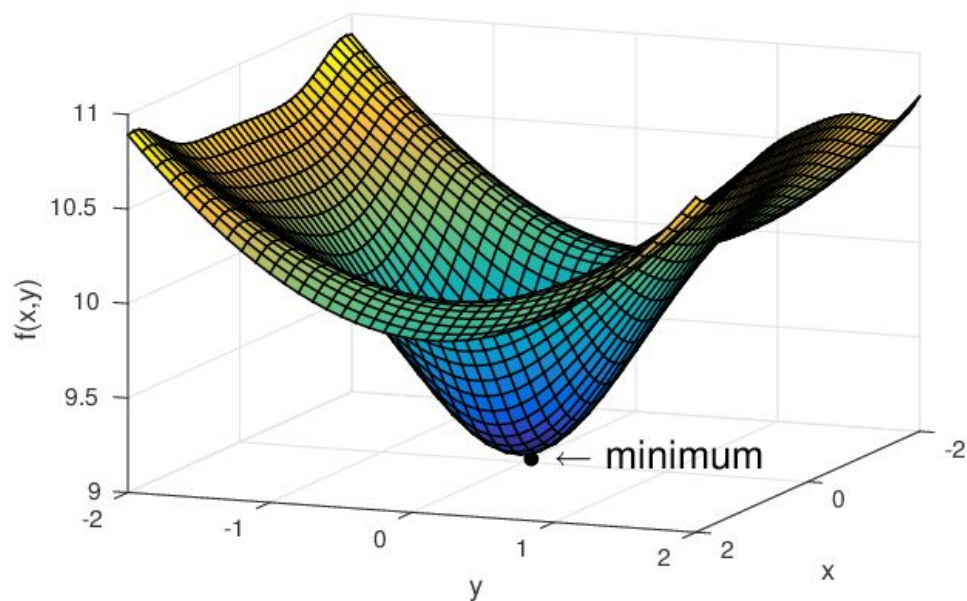


При $x_1 - x_0 \rightarrow 0$, секущая переходит в касательную.

Тангенс угла α наклона этой касательной в точке касательной — и есть производная в точке x_0 .

Частная производная

Но что, если мы имеем дело с функцией от двух (или больше) переменных?

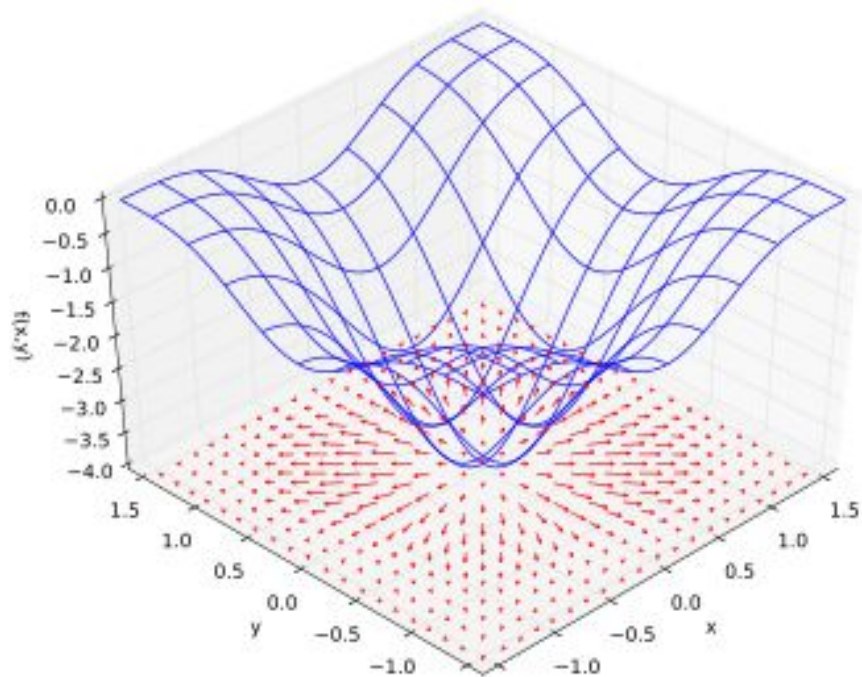


Частная производная

У функции от n переменных $f(x_1, x_2, \dots, x_n)$ нет одной общей производной — зато есть n частные производные.

$$\frac{\partial f}{\partial x_k}(a_1, \dots, a_n) = \lim_{\Delta x \rightarrow 0} \frac{f(a_1, \dots, a_k + \Delta x, \dots, a_n) - f(a_1, \dots, a_k, \dots, a_n)}{\Delta x}$$

Что такое градиент



Градиент — это вектор, элементы которого — значения всех возможных частных производных в конкретной точке.

Градиент соответствует вектору, указывающему направление наибольшего роста функции.

Градиентный спуск

Идея

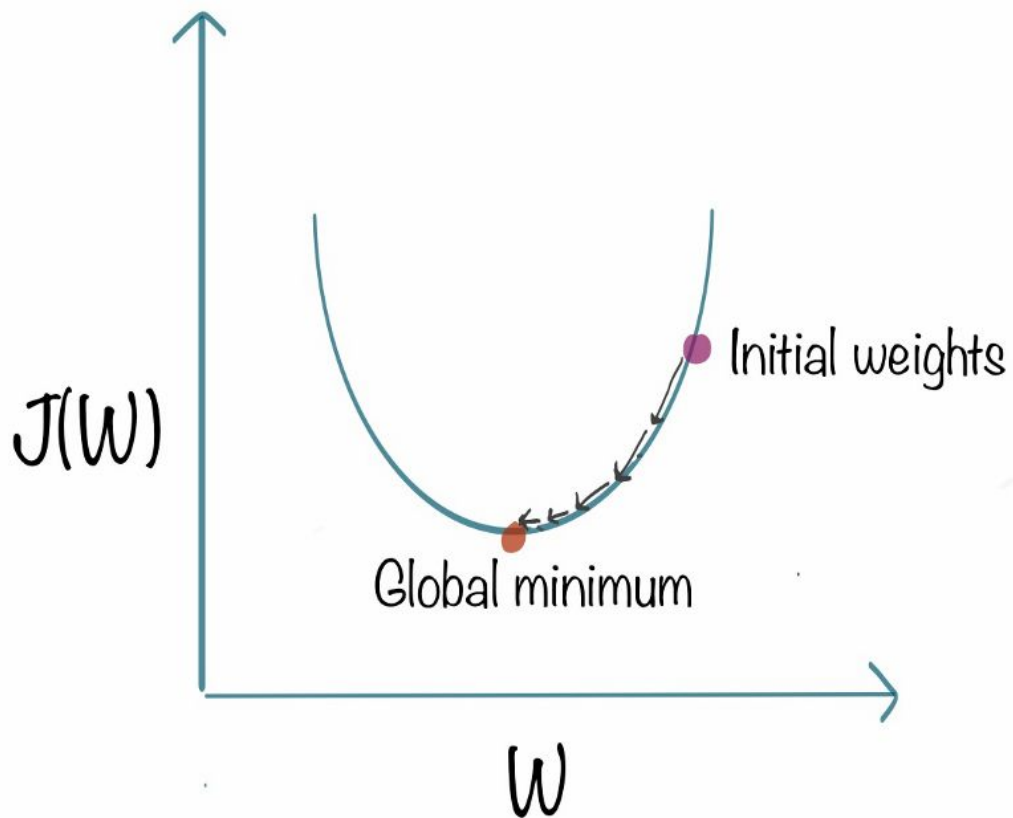
loss function = cost function = error function = функция потерь = $J(W)$

Её мы хотим минимизировать.

Теперь мы умеем находить, в каком направлении функция растёт быстрее всего. Но нам нужен минимум функции потерь, а не максимум!

Решение очевидно: найдём градиент и пойдём в обратную сторону.

С какой скоростью? Растёт быстро — с большой, медленно — с маленькой.



Источник картинки — очень понятно про то, как оно работает и какое бывает.

Шаги:

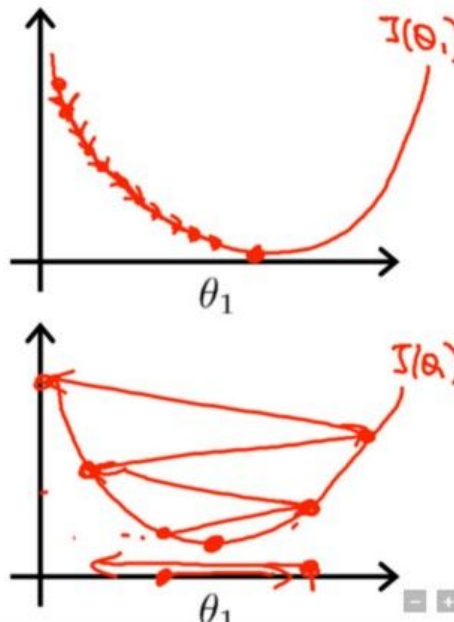
- подобрать случайные коэффициенты
- вычислить градиент функции потерь в этой точке
- обновить коэффициенты
- повторять, пока не сойдётся

Learning rate

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow.

If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.



Learning rate is a hyper-parameter that controls how much we are adjusting the weights of our network with respect the loss gradient. ([отсюда](#))

Каким бывает градиентный спуск

- **Batch gradient descent**

Считает градиент функции потерь с параметрами W сразу для всех обучающих данных. Работает жутко медленно.

- **Stochastic gradient descent (SGD)**

Рандомно выбирает точку данных каждый раз

- **Mini-batch gradient descent**

Выбираем кусочек выборки и по нему считаем

Что делать, если всё ещё ничего непонятно

Непонимание градиентного спуска, в принципе, не мешает вам решать типичные задачи готовыми инструментами. Но может мешать улучшать модель и решать проблемы, если что-то пойдет не так.

Если всё ещё ничего непонятно, keep calm and:

- пройдите небольшой курс по multivariate calculus на khan academy
- посмотрите [вот это видео](#) про градиентный спуск
- прочитайте [эту](#) и [эту](#) статью
- если удастся сформулировать вопросы, feel free to ask

Логистическая регрессия

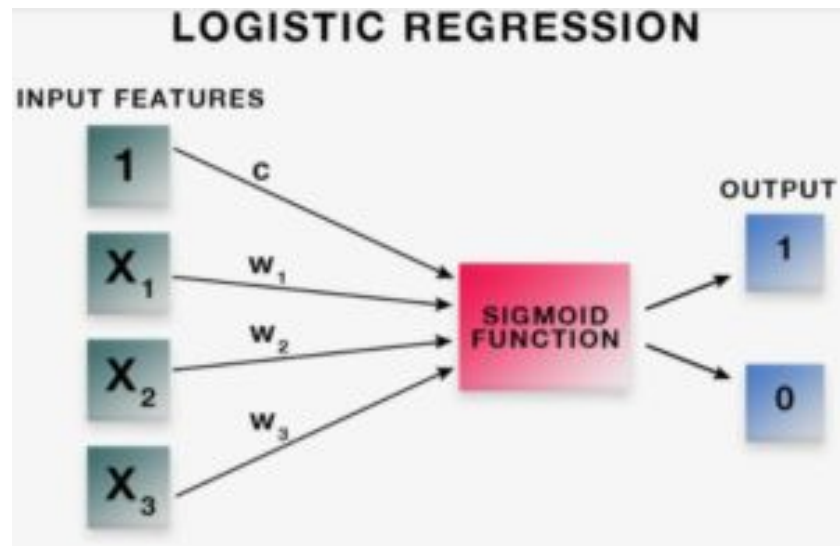
Для чего

Несмотря на название, это алгоритм классификации.

Подбирает веса коэффициентов, скормливает

$$x * w = (x_1, x_2, \dots) * (w_1, w_2, \dots)$$

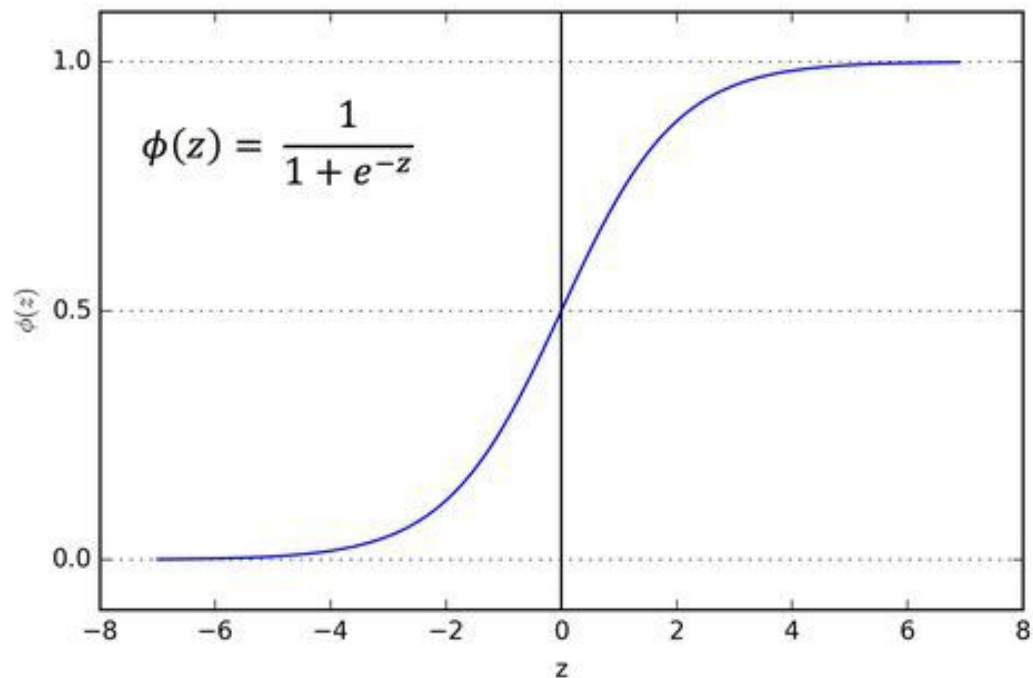
функции-сигмоиду, которая принимает значения от 0 (класс 1) до 1 (класс 2).



$$y = \text{logistic} (c + x_1 * w_1 + x_2 * w_2 + x_3 * w_3 + \dots + x_n * w_n)$$

$$y = 1 / 1 + e [-(c + x_1 * w_1 + x_2 * w_2 + x_3 * w_3 + \dots + x_n * w_n)]$$

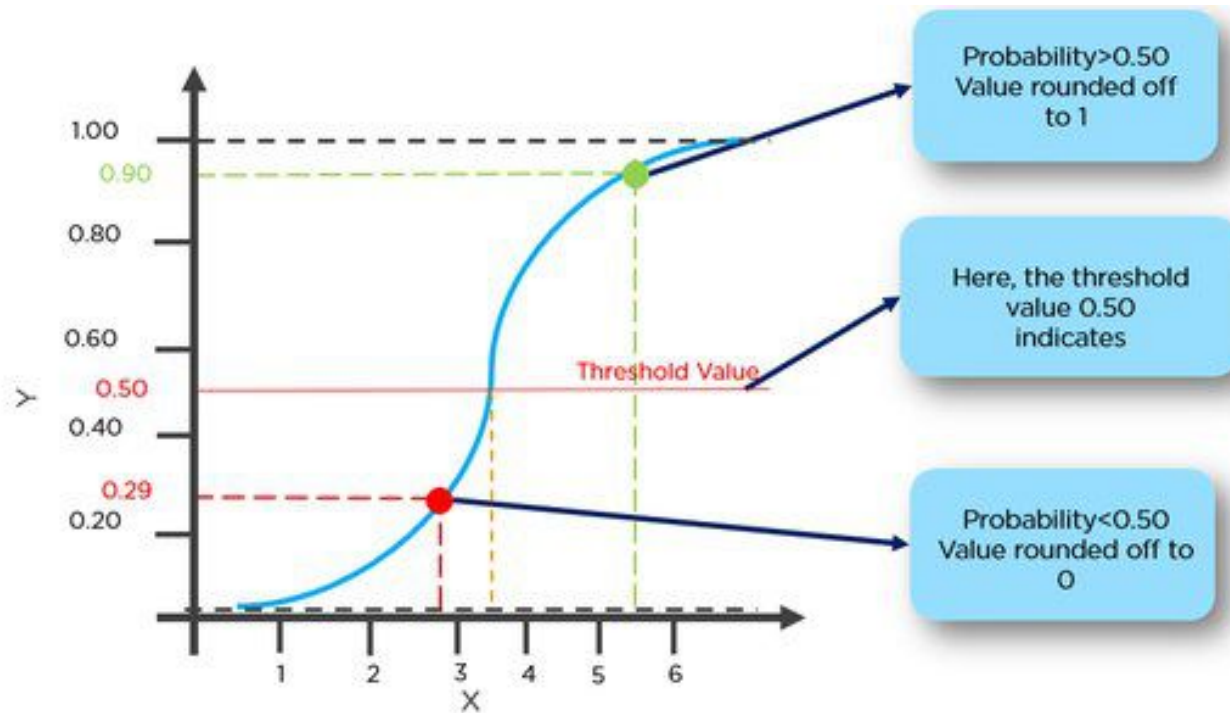
Sigmoid function



z — вектор признаков,
перемноженных на
параметры:

$$w_1 * x_1 + w_2 * x_2 + \dots$$

Что в итоге



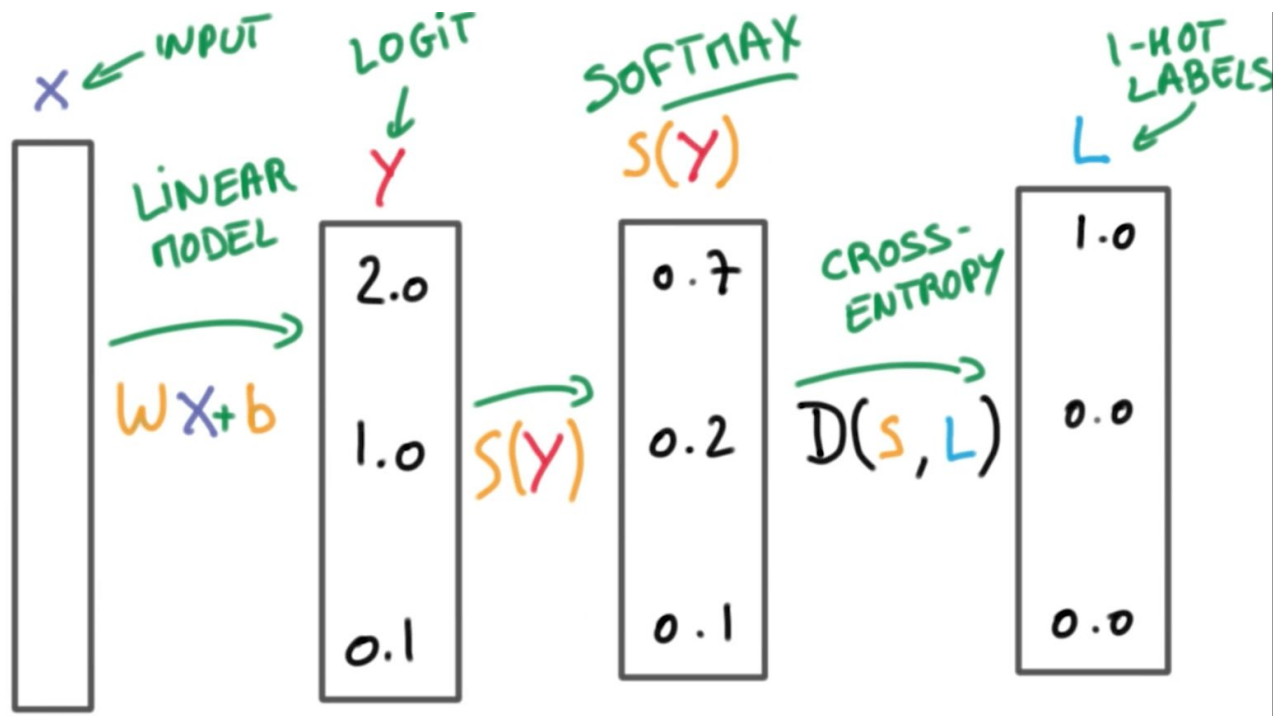
Мультиклассовая классификация и Softmax

Softmax — нормализация вектора вероятностей (= сделать так, чтобы сумма вероятностей была 1).

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad \text{for } j = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K$$

Логистическая регрессия выдаёт вероятность принадлежности к определённому классу vs. все остальные — для каждого класса. Чтобы понять, как какому классу всё-таки объект относится, прогоняем эти вероятности через softmax.

Мультиклассовая классификация и Softmax



Гиперпараметры

Параметры vs гиперпараметры

Параметр — это внутренняя характеристика модели, значение которой может быть выведено из данных. Это, например, коэффициент при признаке “слово *КОТИК*”.

Гиперпараметр — это характеристика, “внешняя” по отношению к модели. Например, k в kNN . Его нельзя вывести из данных при обучении, и надо подбирать как-то отдельно.

(Определения из [статьи](#)).

Подбор гиперпараметров

Как подобрать гиперпараметры?

- можно пробовать менять разные варианты руками
- можно перебирать их в цикле

```
for df in range(0, 20):  
    vec = TfidfVectorizer(min_df=df)
```

- а можно использовать Grid Search

```
from sklearn.model_selection import GridSearchCV
```

Ресурсы

Почитать (англ)

- [очень понятная статья про градиентный спуск](#)
- [Machine learning for humans: градиентный спуск, регрессия](#)
- [gradient descent into madness](#)
- [градиентный спуск своими руками](#)
- [про learning rate](#)
- [логистическая регрессия с кодом](#)
- [понятно про softmax](#)
- [про Grid Search](#)

Посмотреть

- почему градиент работает
- как работает градиентный спуск на примере нейросеток (5:20 — 12:20)