Введение в глубинное обучение

Кантонистова Елена Олеговна

План занятия

- Идея нейронных сетей
- Полносвязные нейронные сети
- Функции активации
- Практика

Когда нужны нейронные сети?

Классический ML:

- Конструируем и вычисляем признаки
- Обучаем на них алгоритм

Используем, когда объекты обладают разнородными по смыслу (и осмысленными) признаками.

Нейронные сети:

- Сами извлекают признаки из данных

Используем, когда признаки объекта однородные и не несут смысла (пиксели на изображении, буквы в тексте и т.д.)

Схема нейрона в мозге

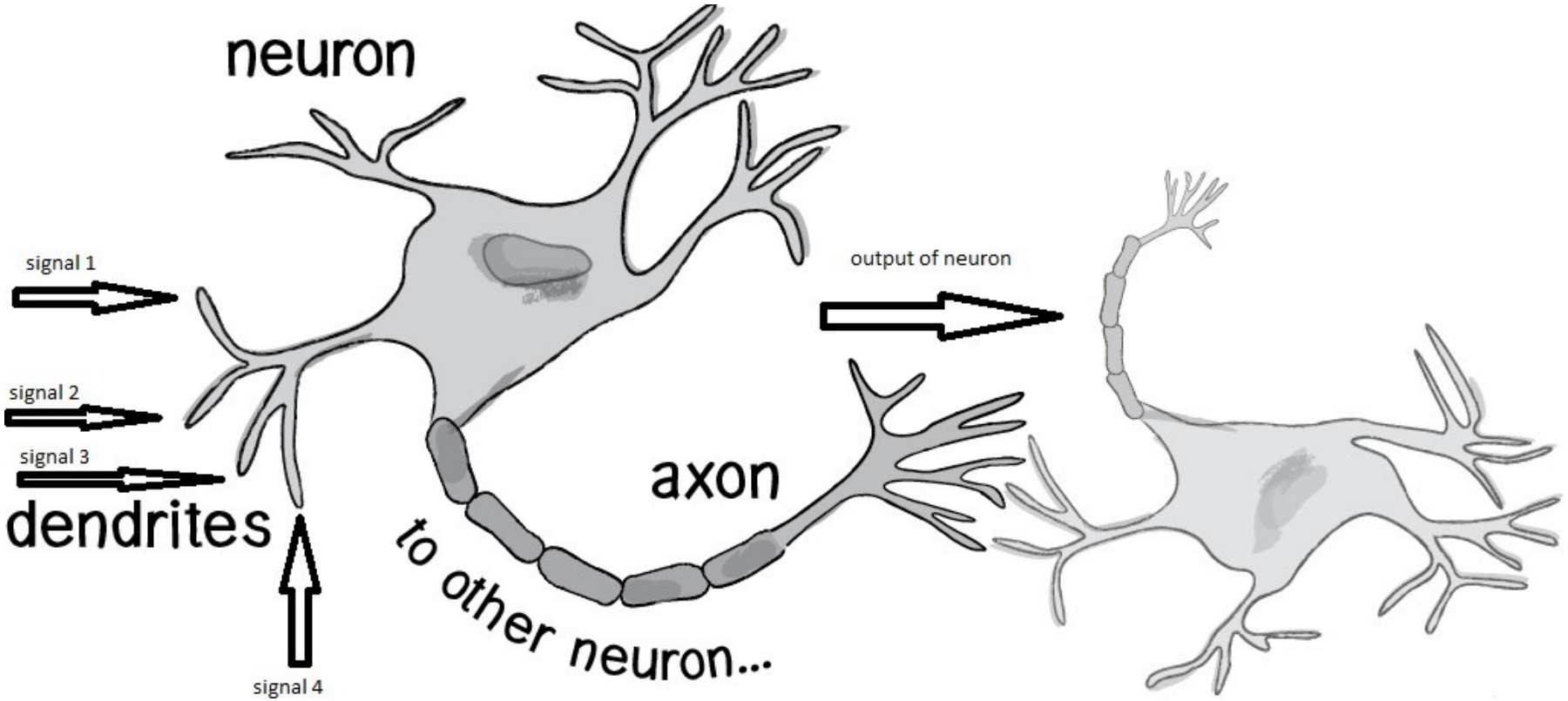
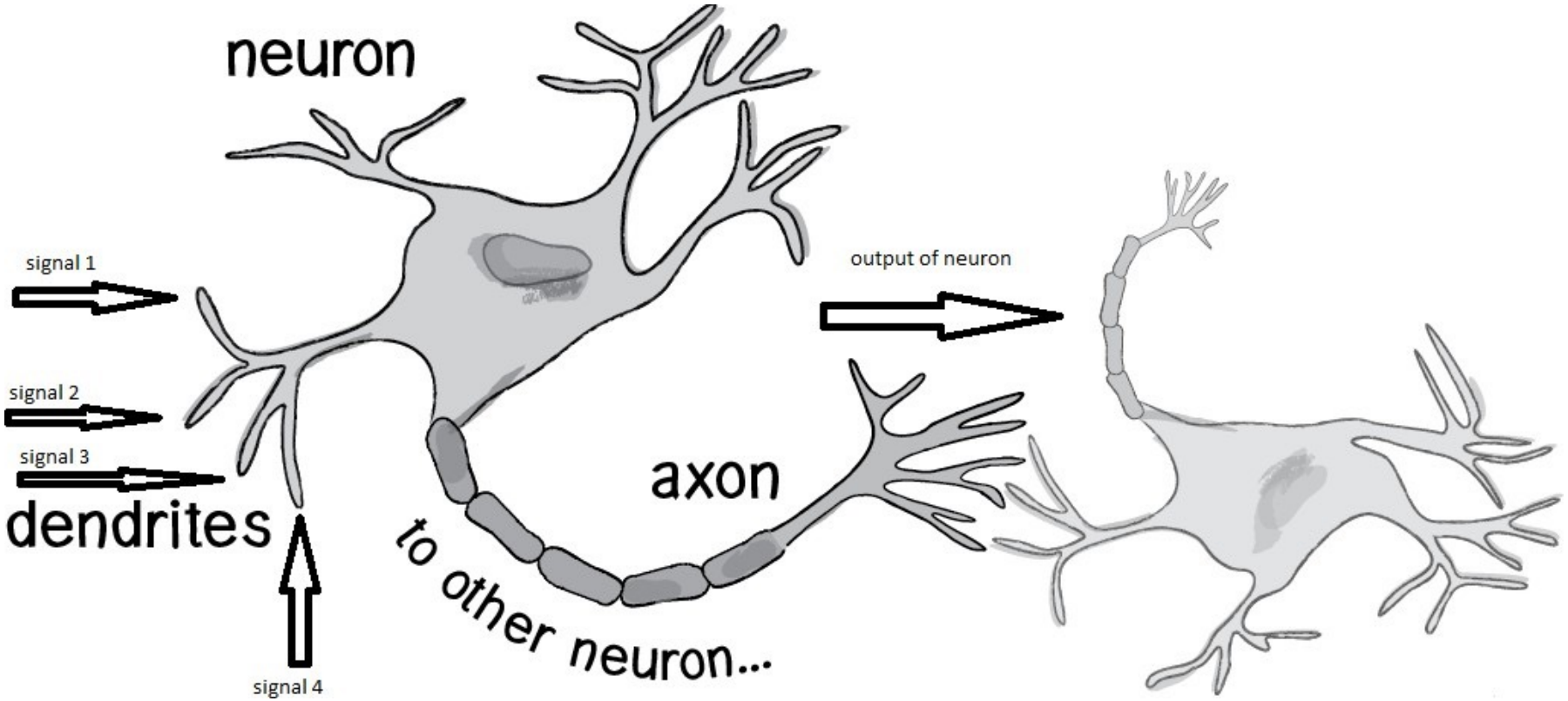


Схема нейрона в мозге



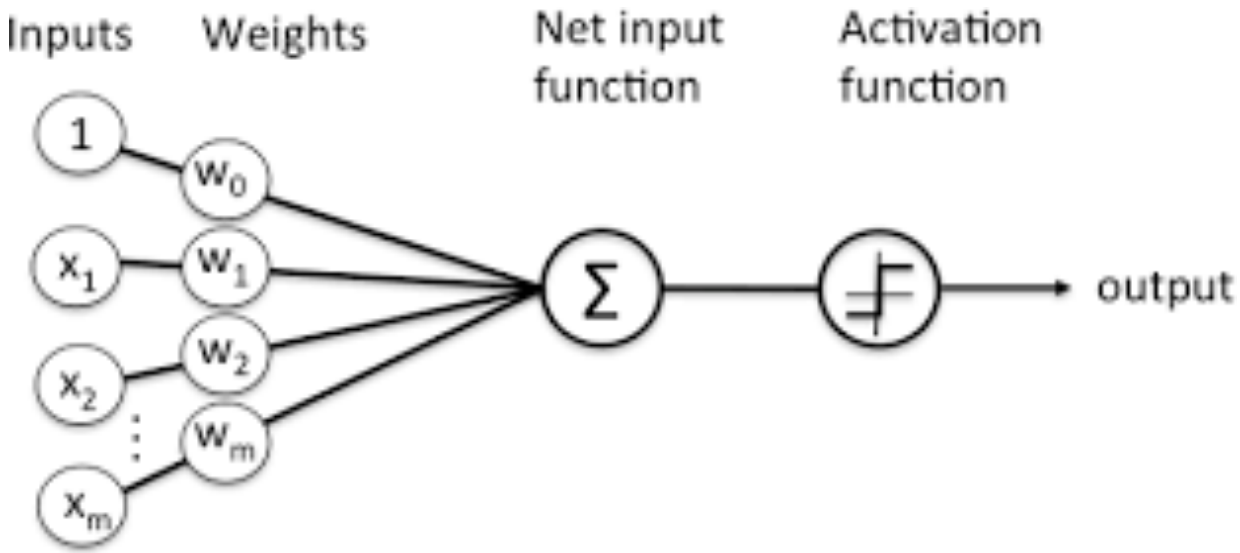
Линейная модель:

$$a(x, w) = \sigma(w_0 + \sigma_{j=1}^{n} w_j x_j),$$

σ – функция активации

Модель нейрона

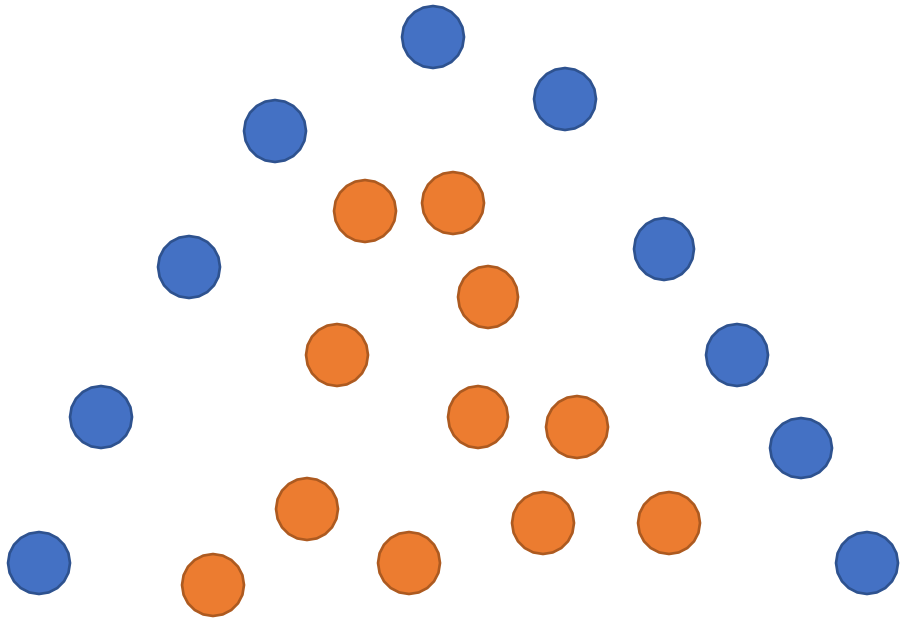
$$a(x,w) = \sigma(w_0 + \sum_{j=1}^{n} w_j x_j)$$



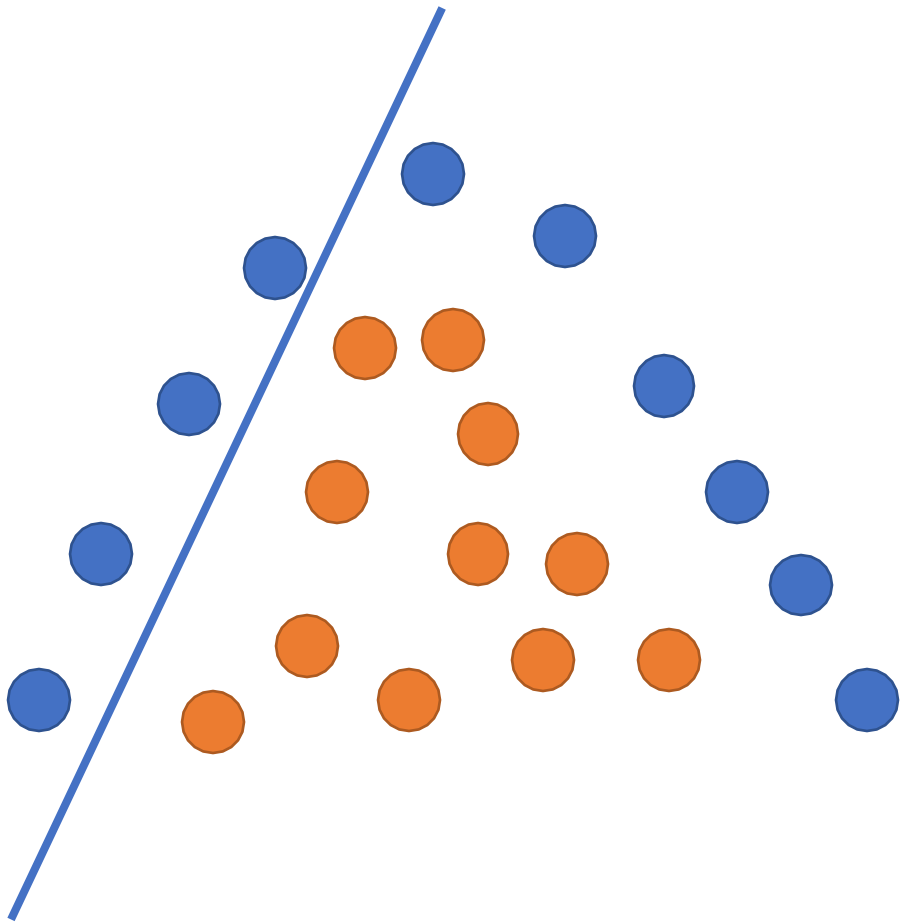
Функции активации:

- $\sigma(z) = sign(z) \Rightarrow a(x,w) = sign[w_0 + \sigma_{j=1}^{n} w_j x_j]$
- $\sigma(z) = \frac{1}{1+\exp(-z)} \Rightarrow a(x,w) = \frac{1}{1+\exp(-\left(w_0+\sigma_{j=1}^{n} w_j x_j\right))}$

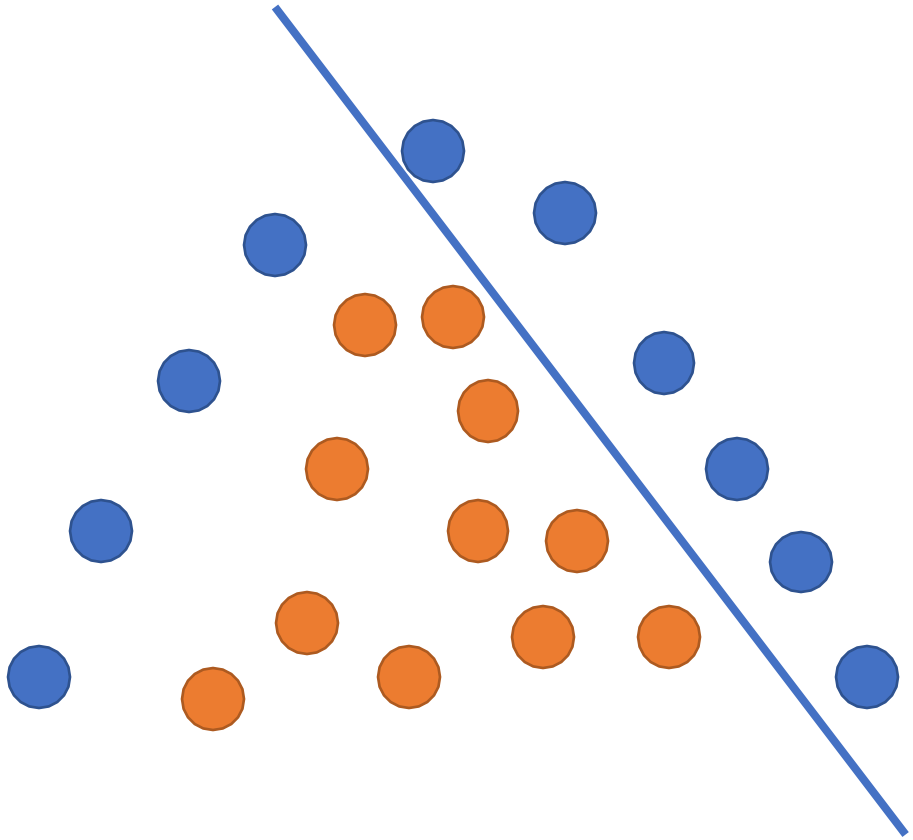
Пример



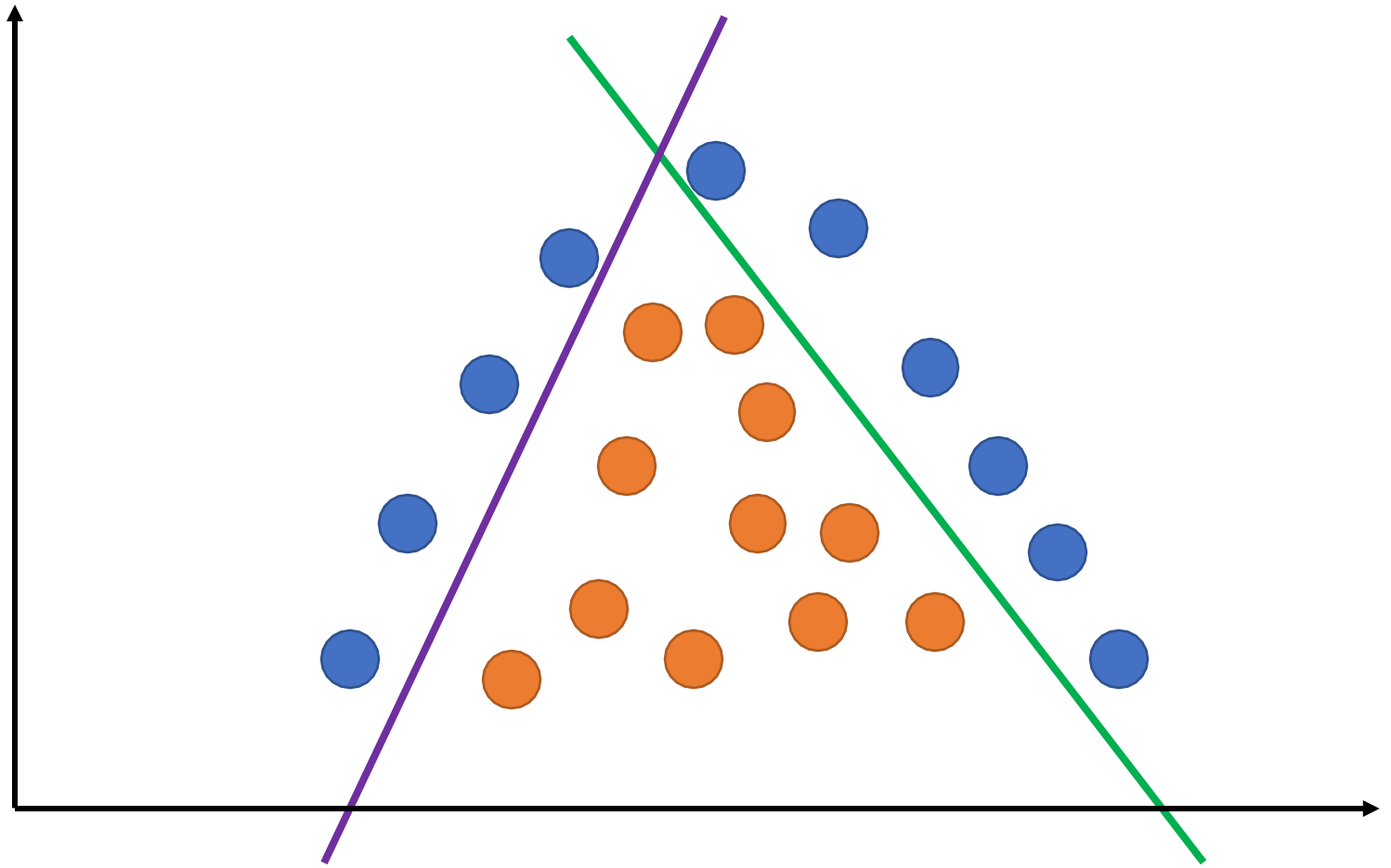
Пример



Пример



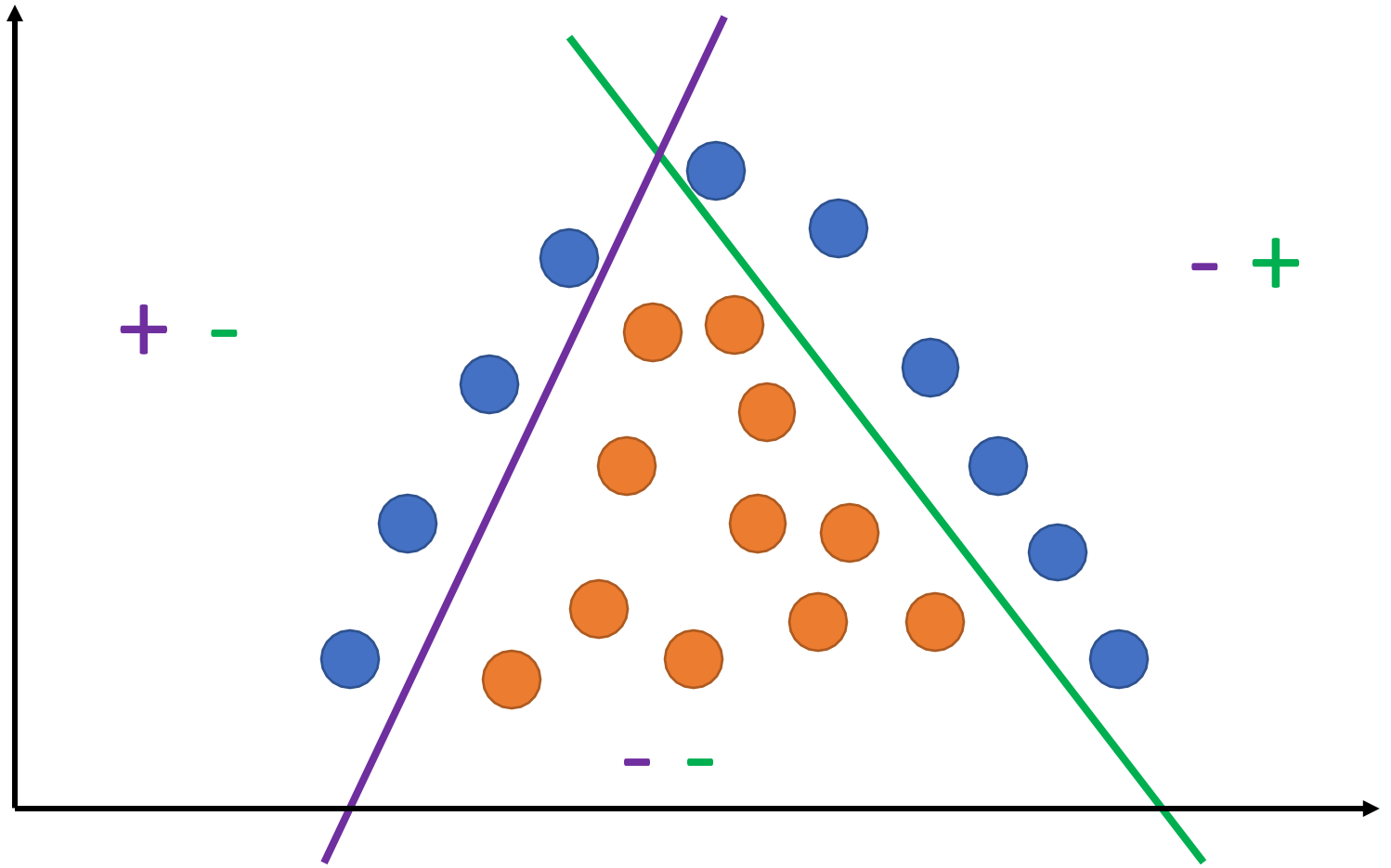
Пример



$b_1(x) = \text{sign}(-2x_1 + x_2 + 5)$

$b_2(x) = \text{sign}(2x_1 + x_2 - 5)$

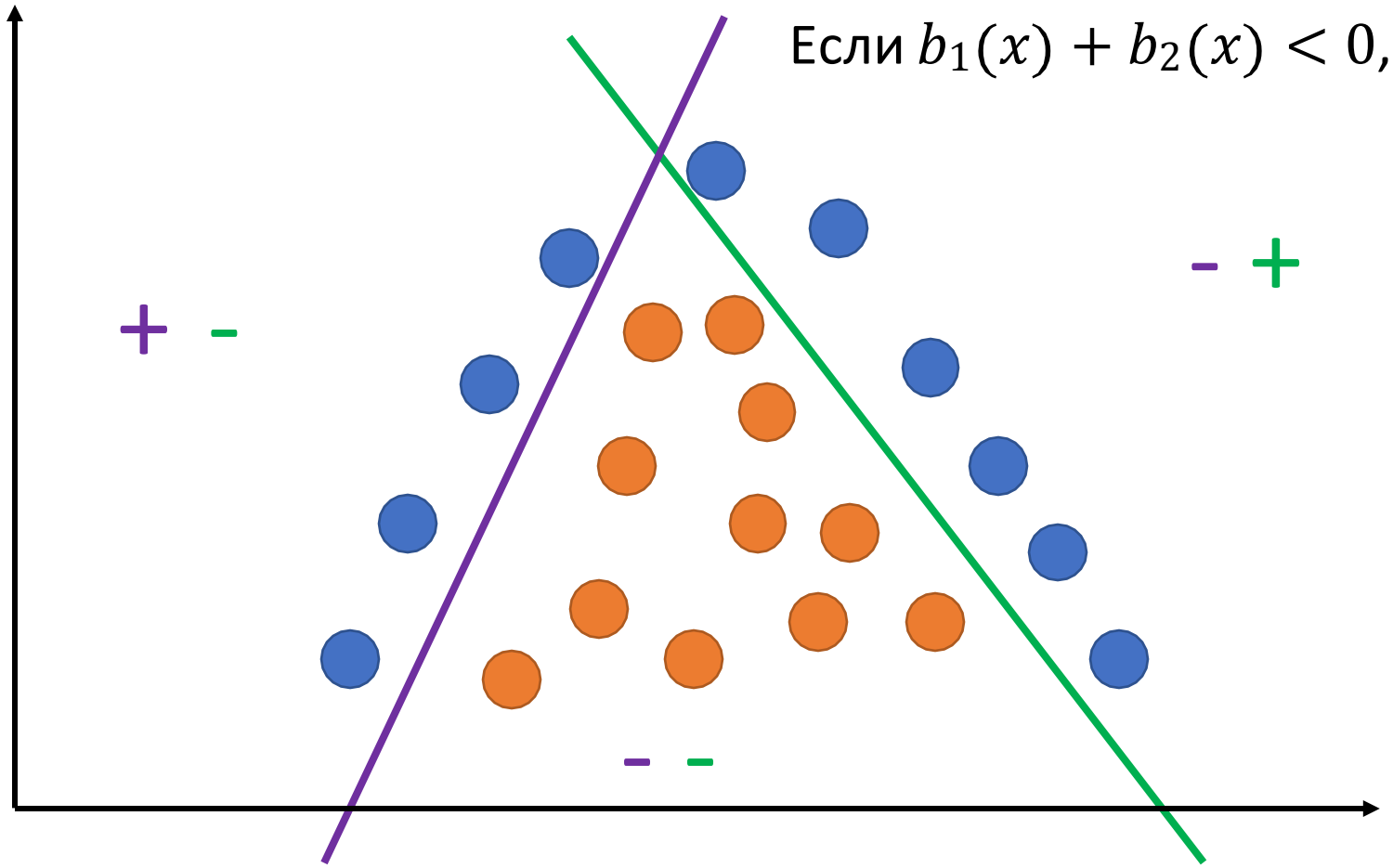
Пример



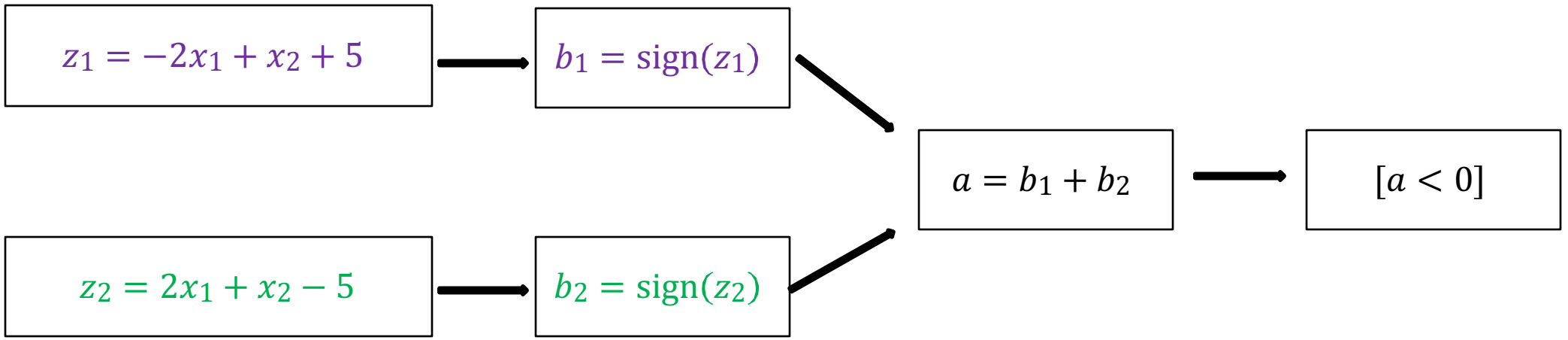
$b_1(x) = \text{sign}(-2x_1 + x_2 + 5)$

$b_2(x) = \text{sign}(2x_1 + x_2 - 5)$

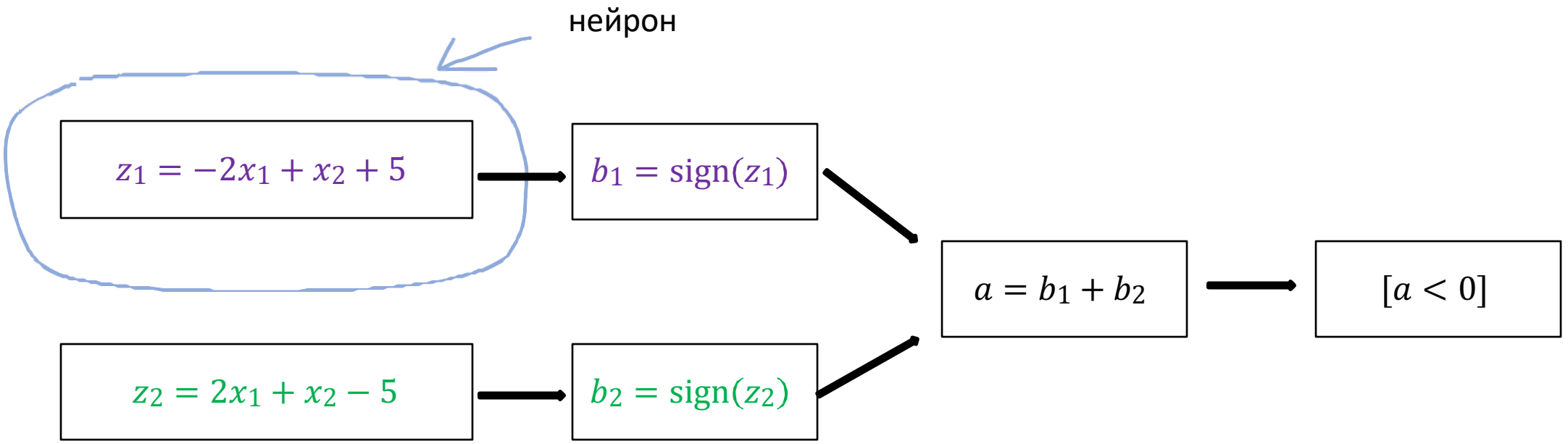
Пример


Если $b_1(x) + b_2(x) < 0$, то оранжевый класс
+ -
- +
- -
$b_1(x) = \text{sign}(-2x_1 + x_2 + 5)$
$b_2(x) = \text{sign}(2x_1 + x_2 - 5)$

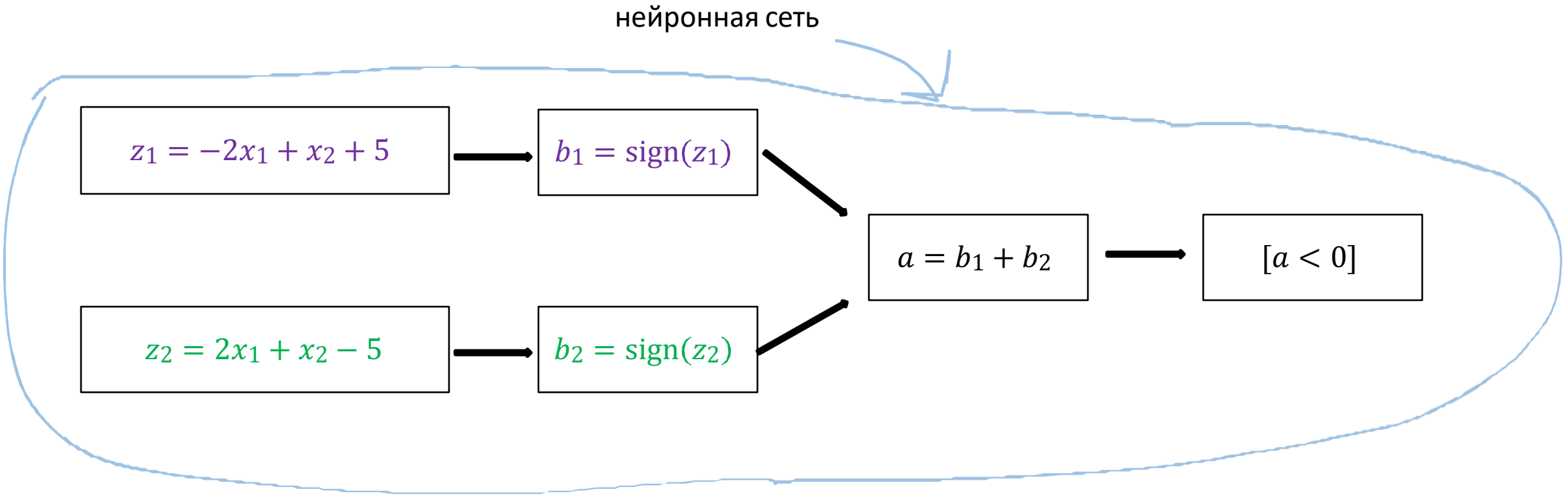
Пример



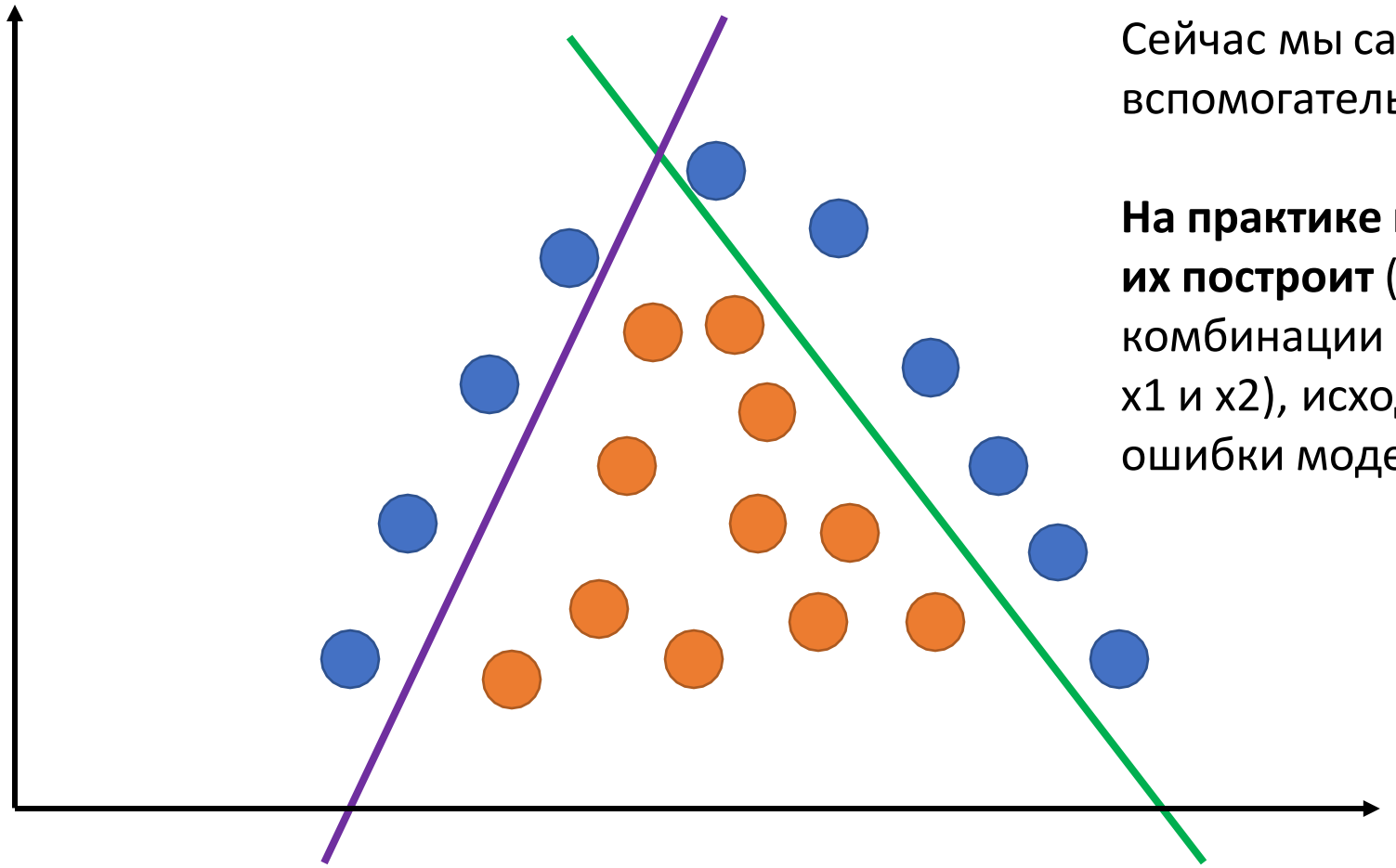
Пример


нейрон
$z_1 = -2x_1 + x_2 + 5$
$b_1 = \text{sign}(z_1)$
$z_2 = 2x_1 + x_2 - 5$
$b_2 = \text{sign}(z_2)$
$a = b_1 + b_2$
$[a < 0]$

Пример


нейронная сеть
$z_1 = -2x_1 + x_2 + 5$
$b_1 = \text{sign}(z_1)$
$z_2 = 2x_1 + x_2 - 5$
$b_2 = \text{sign}(z_2)$
$a = b_1 + b_2$
$[a < 0]$

Пример



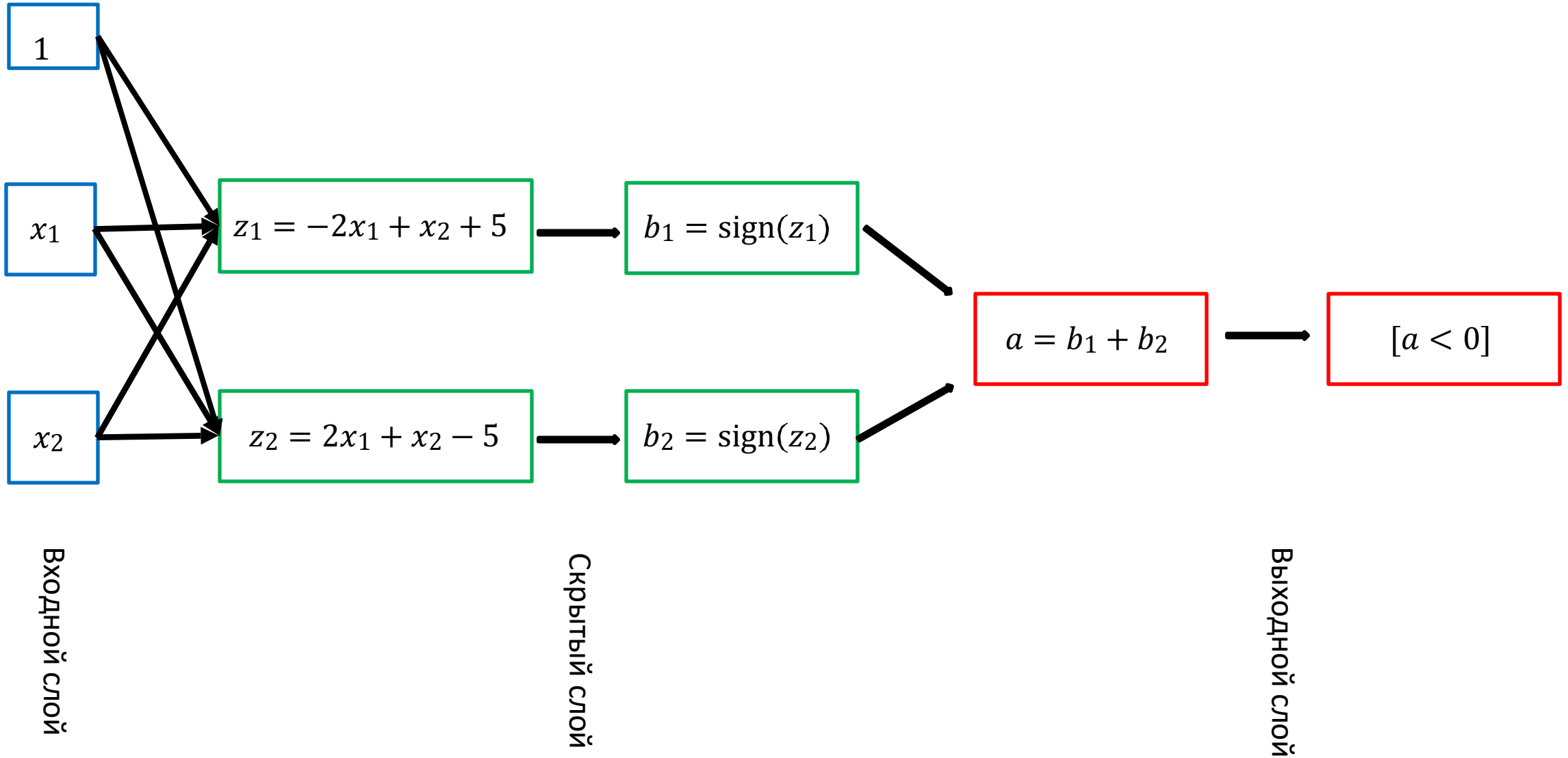
$b_1(x) = \text{sign}(-2x_1 + x_2 + 5)$

$b_2(x) = \text{sign}(2x_1 + x_2 - 5)$

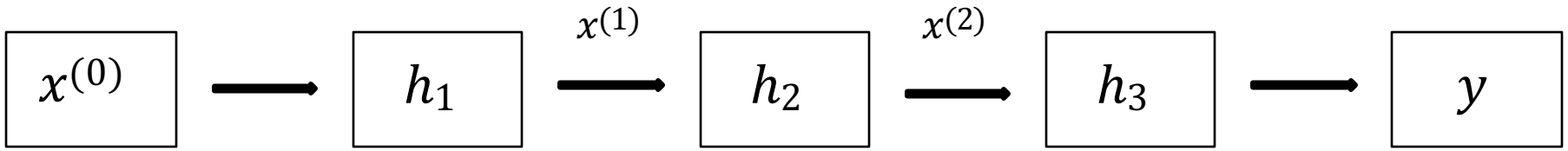
Сейчас мы самостоятельно провели вспомогательные прямые.

На практике нейронная сеть сама их построит (как линейные комбинации исходных признаков x1 и x2), исходя из минимизации ошибки модели.

Пример

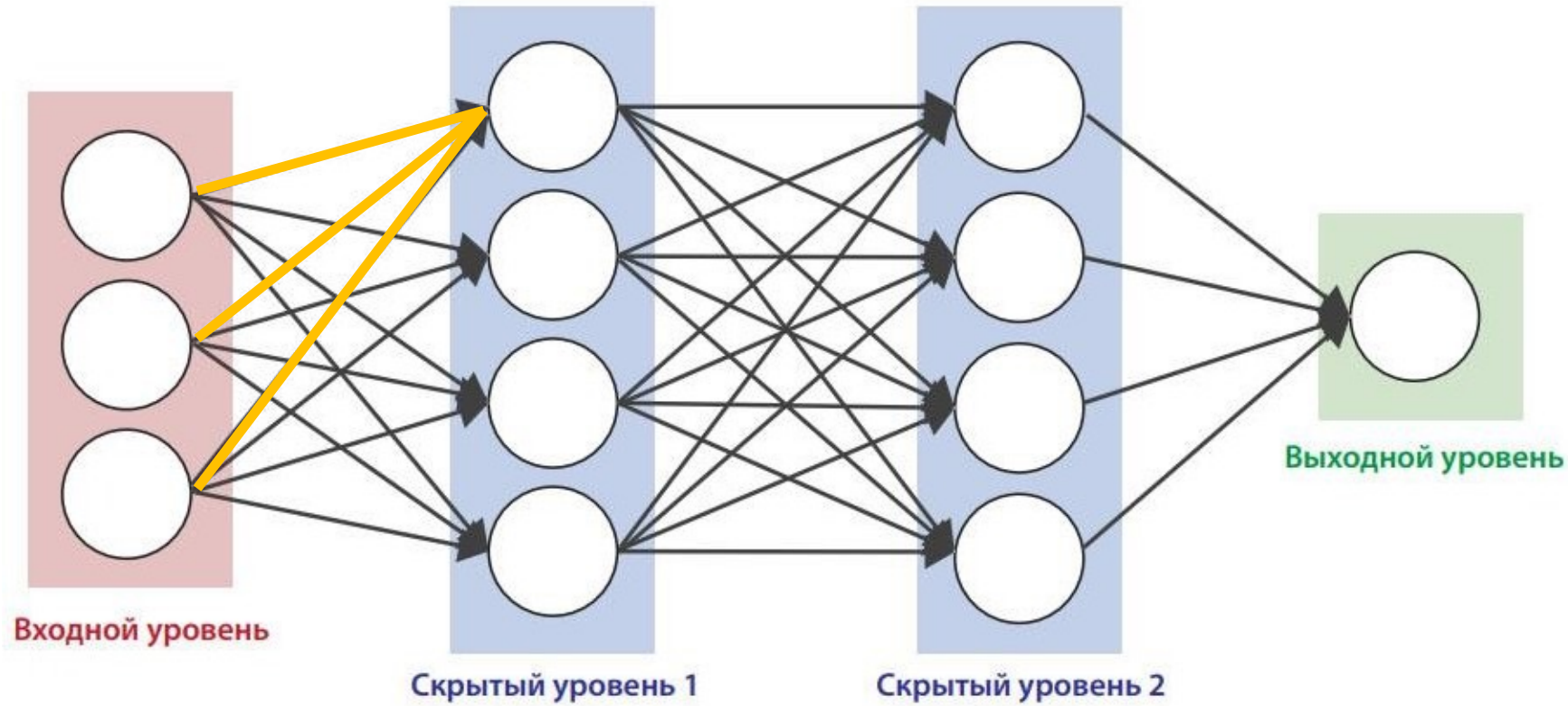

1
x_1
x_2
$z_1 = -2x_1 + x_2 + 5$
$z_2 = 2x_1 + x_2 - 5$
$b_1 = \text{sign}(z_1)$
$b_2 = \text{sign}(z_2)$
$a = b_1 + b_2$
$[a < 0]$
Входной слой
Скрытый слой
Выходной слой

Граф вычислений (нейронная сеть)



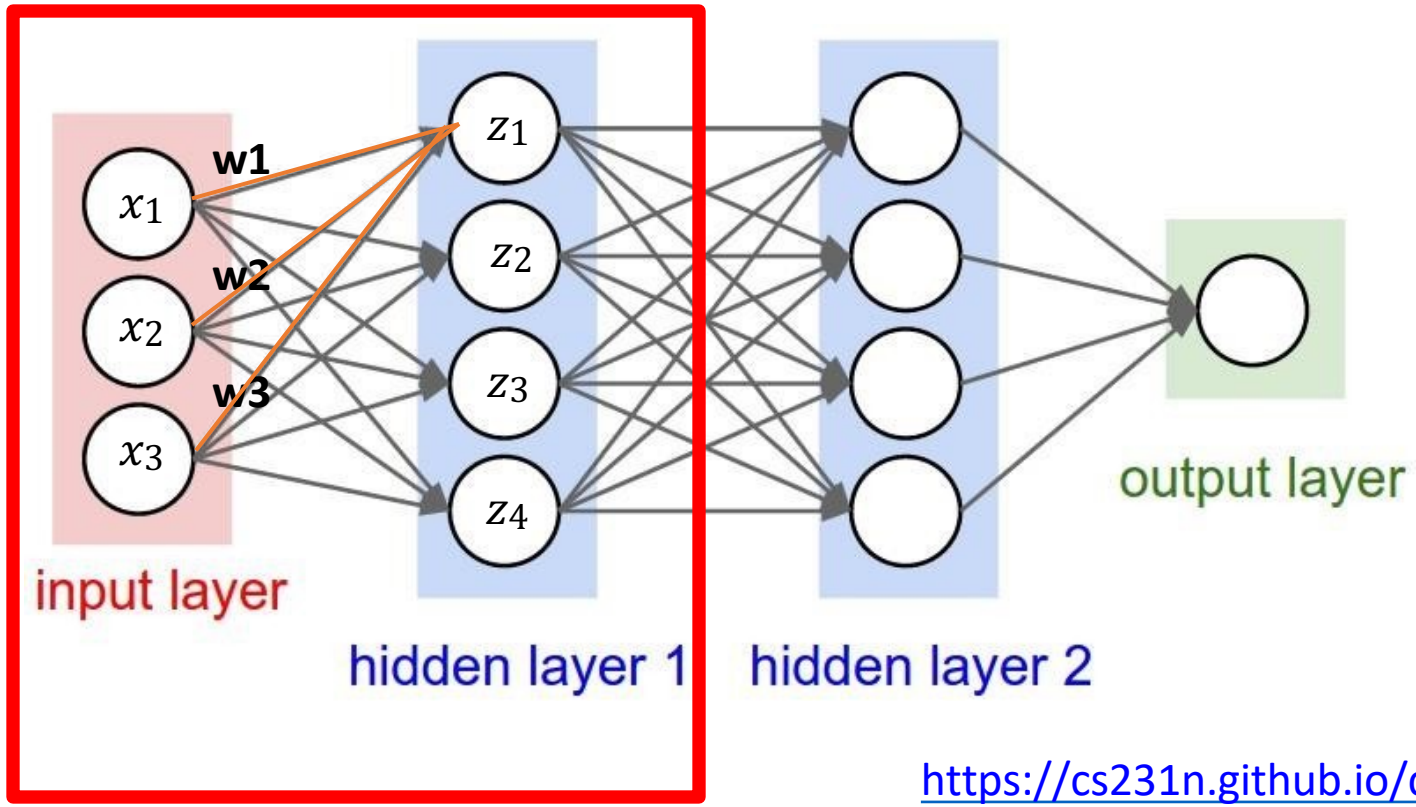
Полносвязные слои

Полносвязная нейронная сеть



Полносвязный слой (fully connected)

$$z_1 = w_0 + x_1 w_1 + x_2 w_2 + x_3 w_3$$



Полносвязный слой нейронной сети

https://cs231n.github.io/convolutional-networks/

Важный вопрос в DL

Что такое композиция нескольких линейных моделей?

Важный вопрос в DL

Что такое композиция нескольких линейных моделей?

Линейная модель ☹

Нелинейность

- Нужно добавлять **нелинейную функцию** после полносвязного слоя

$$z_j = \boldsymbol{f}\left(\sum_{i=1}^{n} w_{ji} x_i + b_j\right)$$

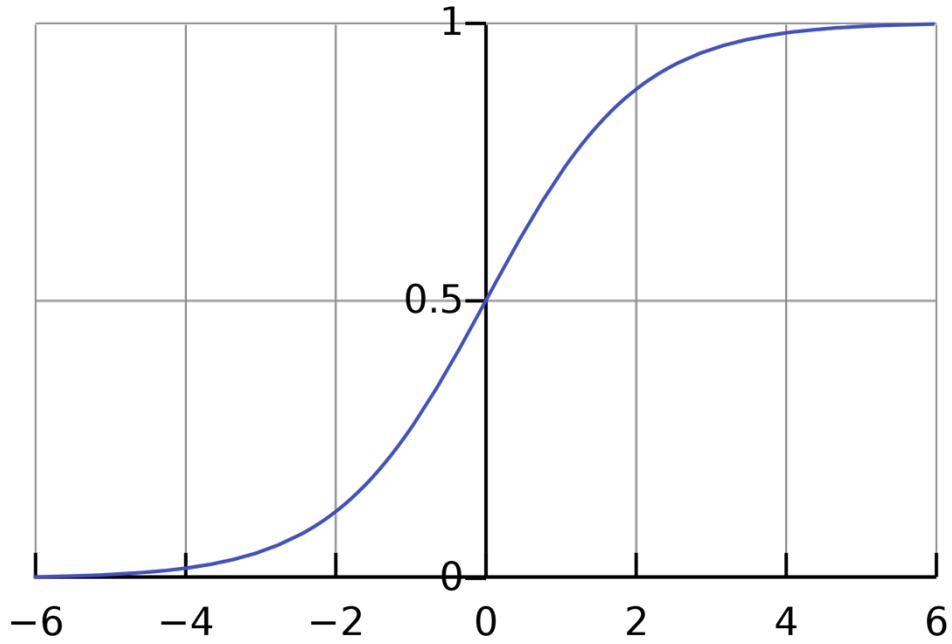
- Функция $\boldsymbol{f}$ называется **функцией активации**.

Функция активации

$$z_j = \boldsymbol{f}\left(\sum_{i=1}^{n} w_{ji}x_i + b_j\right)$$

Вариант 1: $\boldsymbol{f(x) = \frac{1}{1+\exp(-x)}}$

(сигмоида)

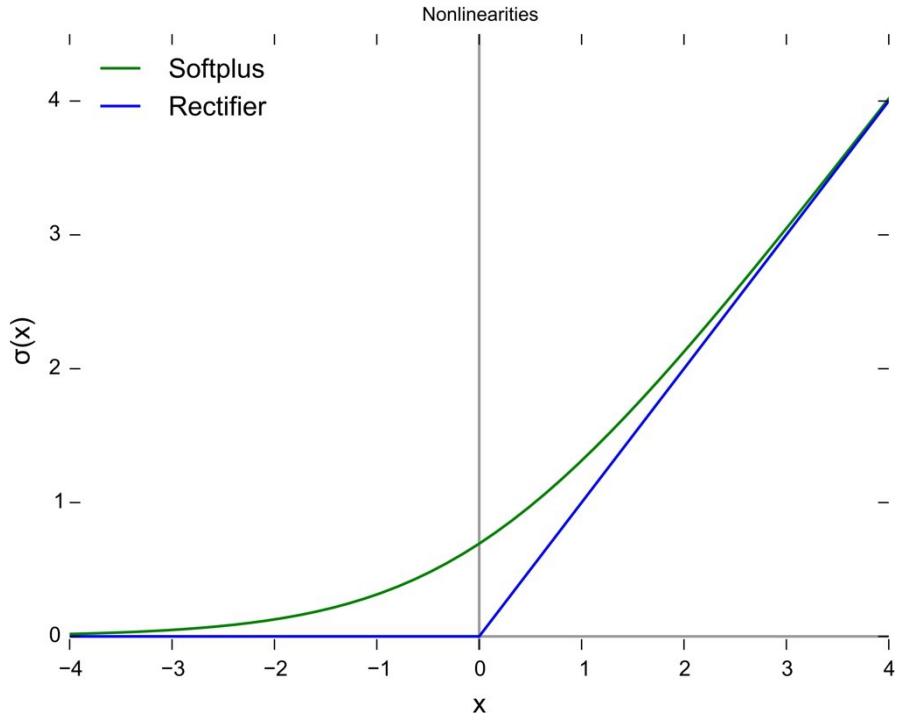


Функция активации

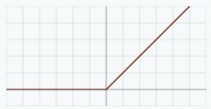
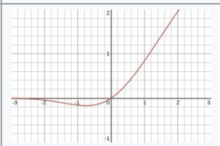
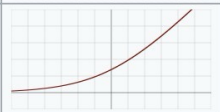
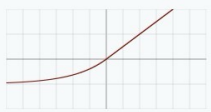
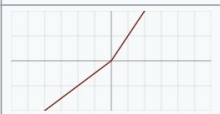
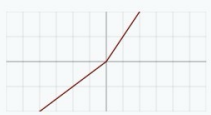
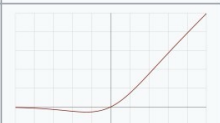
$$z_j = \boldsymbol{f}\left(\sum_{i=1}^{n} w_{ji}x_i + b_j\right)$$

Вариант 2: $\boldsymbol{f(x) = \max(0, x)}$

(ReLU, Rectified Linear Unit)

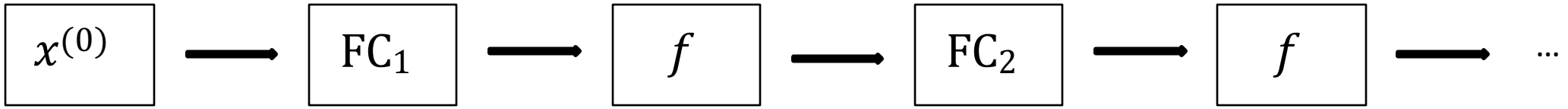


Функция активации

Rectified linear unit (ReLU)[9]		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x\mathbf{1}_{x>0}$
Gaussian Error Linear Unit (GELU)[4]		$\frac{1}{2}x\left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right)$ $= x\Phi(x)$
Softplus[10]		$\ln(1 + e^x)$
Exponential linear unit (ELU)[11]		$\begin{cases} \alpha\left(e^x - 1\right) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α
Scaled exponential linear unit (SELU)[12]		$\lambda\begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameters $\lambda = 1.0507$ and $\alpha = 1.67326$
Leaky rectified linear unit (Leaky ReLU)[13]		$\begin{cases} 0.01x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$
Parameteric rectified linear unit (PReLU)[14]		$\begin{cases} \alpha x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$ with parameter α
Sigmoid linear unit (SiLU,[4] Sigmoid shrinkage,[15] SiL,[16] or Swish-1[17])		$\frac{x}{1 + e^{-x}}$

https://en.wikipedia.org/wiki/Activation_function

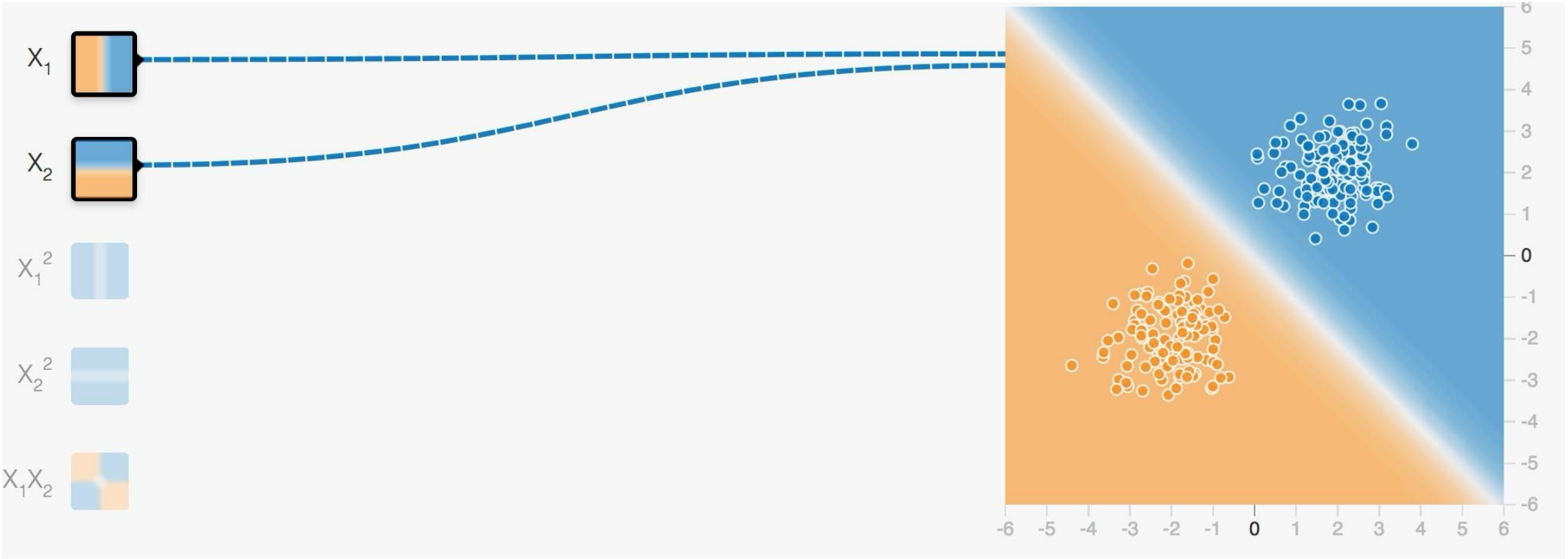
Структура полносвязной сети



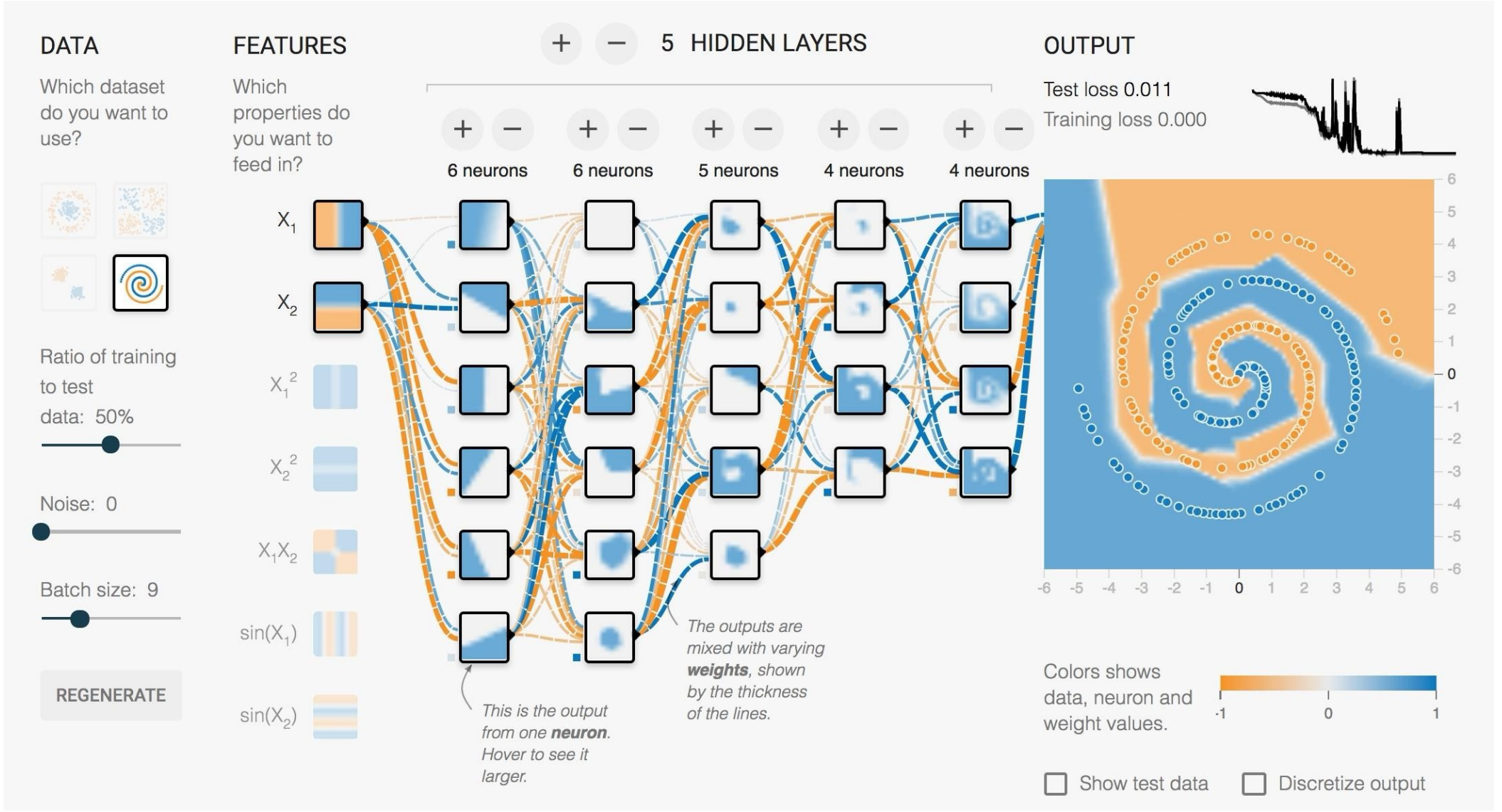
- На входе признаки
- В последнем слое выходов столько, сколько целевых переменных мы предсказываем

Демо

- http://playground.tensorflow.org



Решает сложные задачи

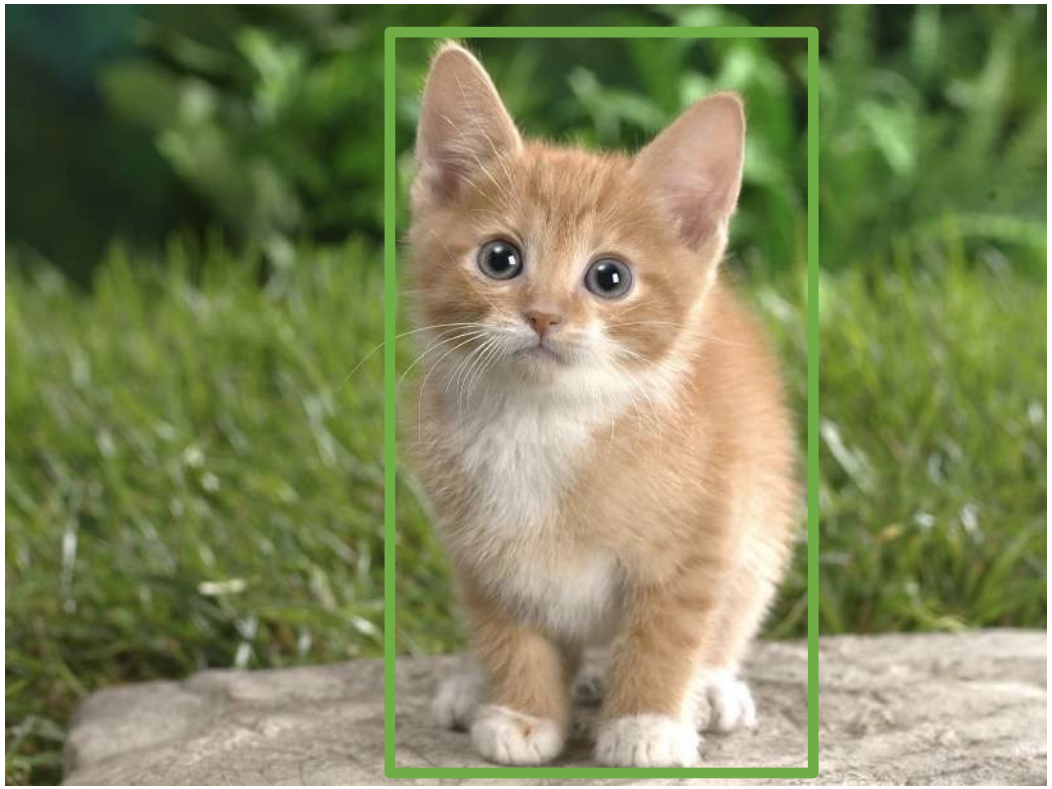


Компьютерное зрение

Классификация изображений

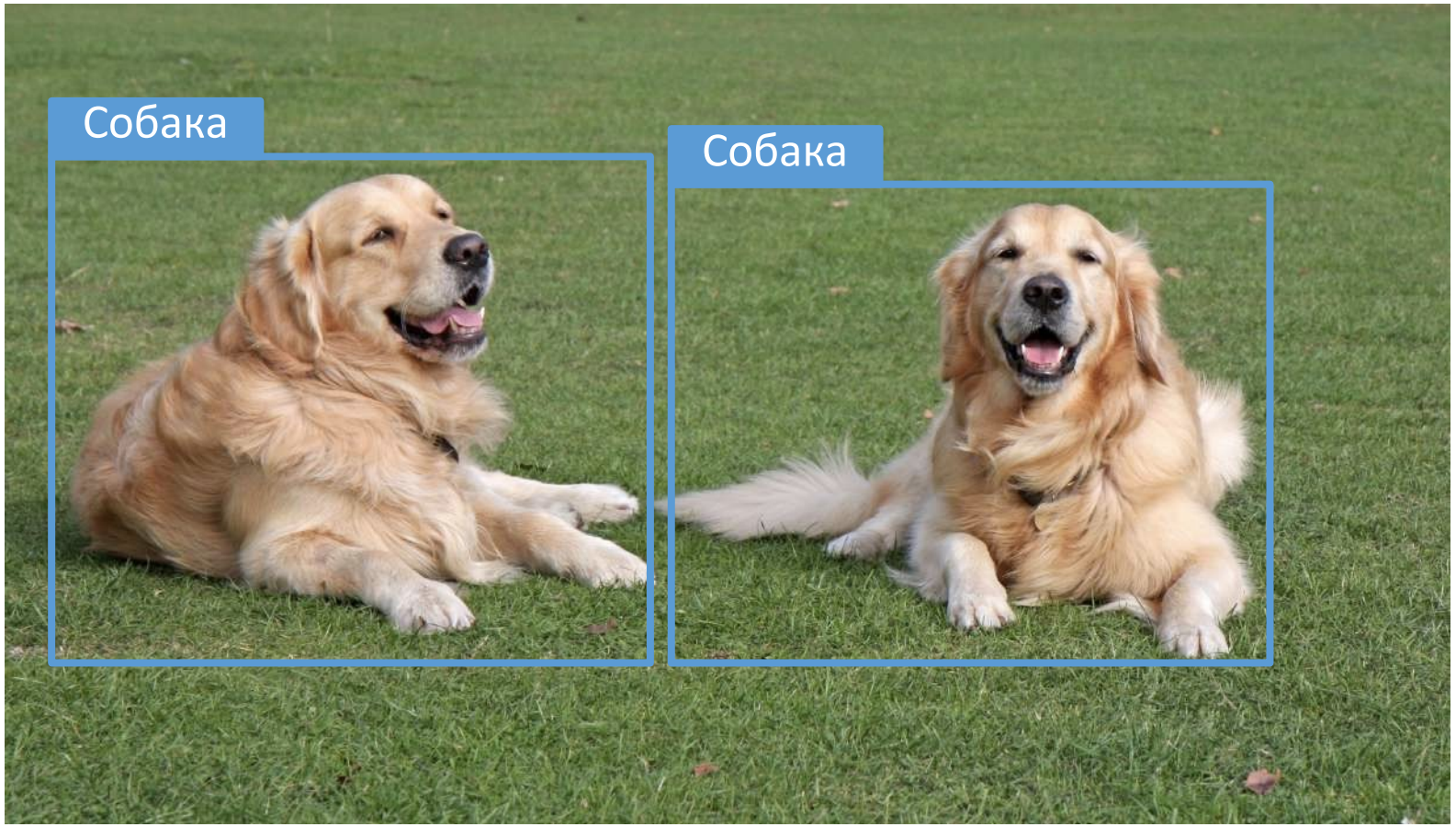
«Кот»

Классификация и локализация



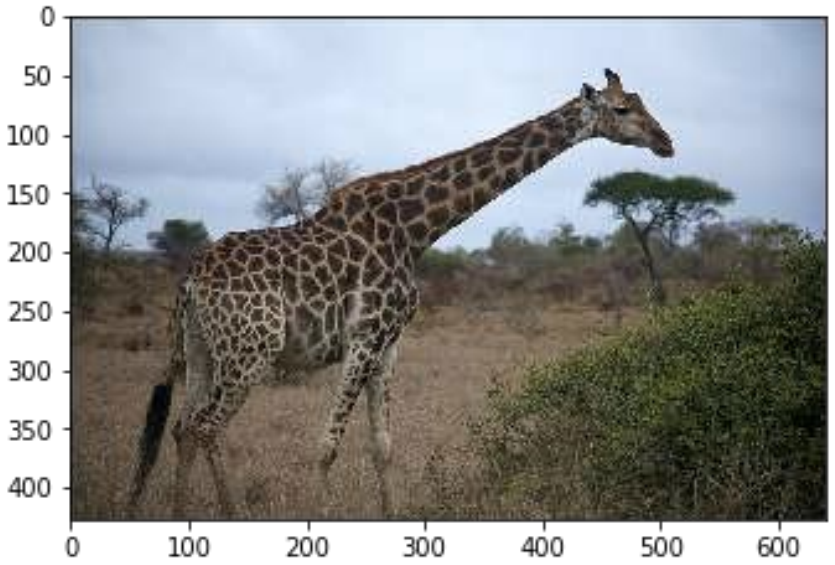
«Кот»

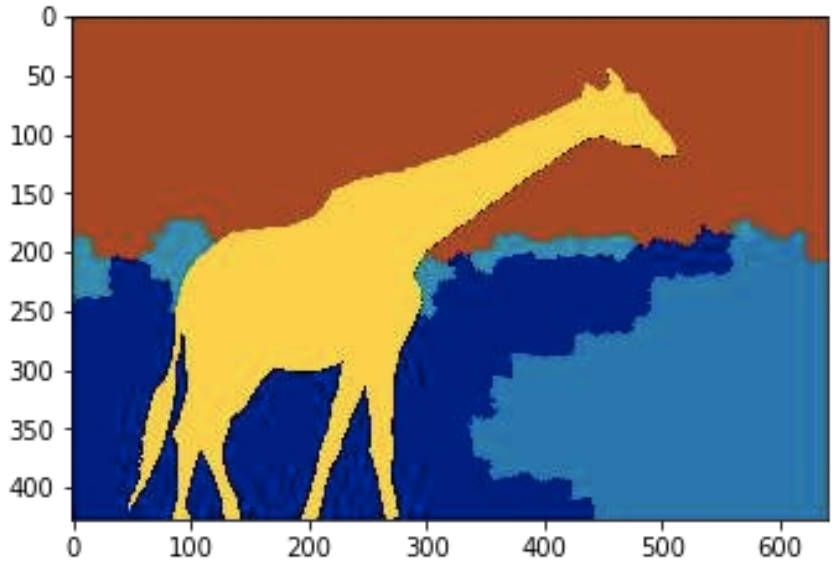
Детектирование объектов



publicdomainpictures.net

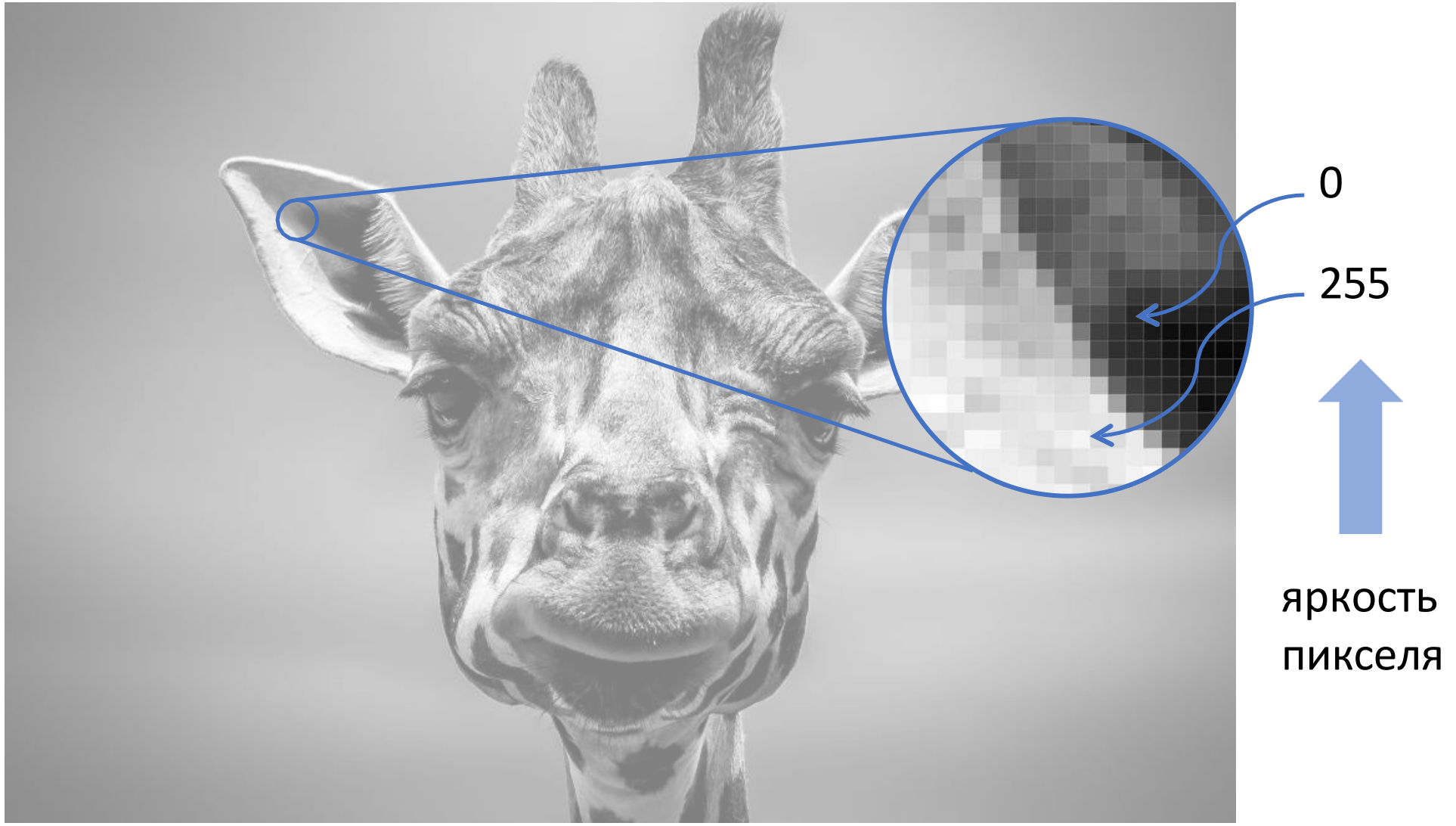
Семантическая сегментация





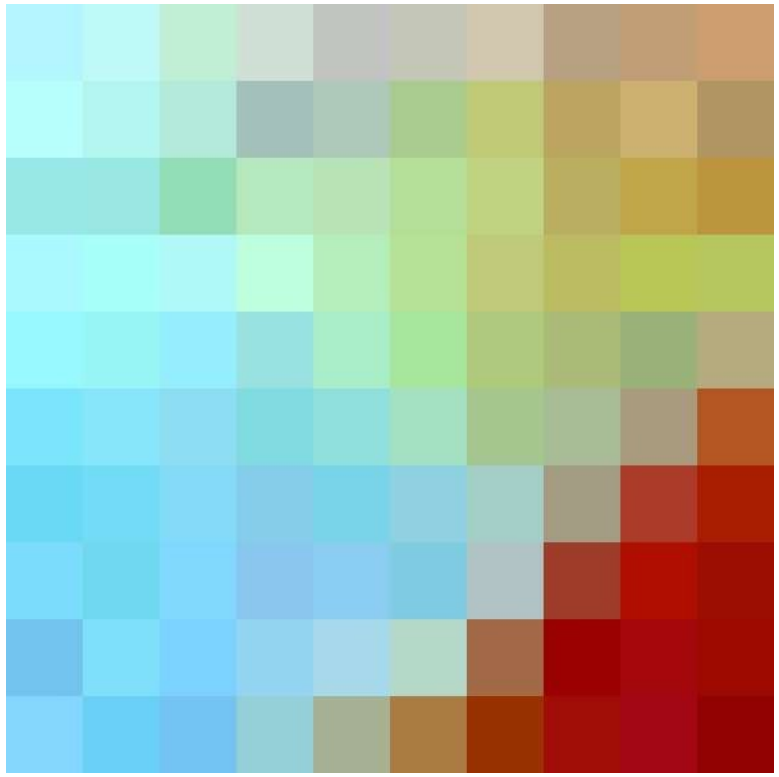
Как мы видим картинку

Как компьютер видит картинку



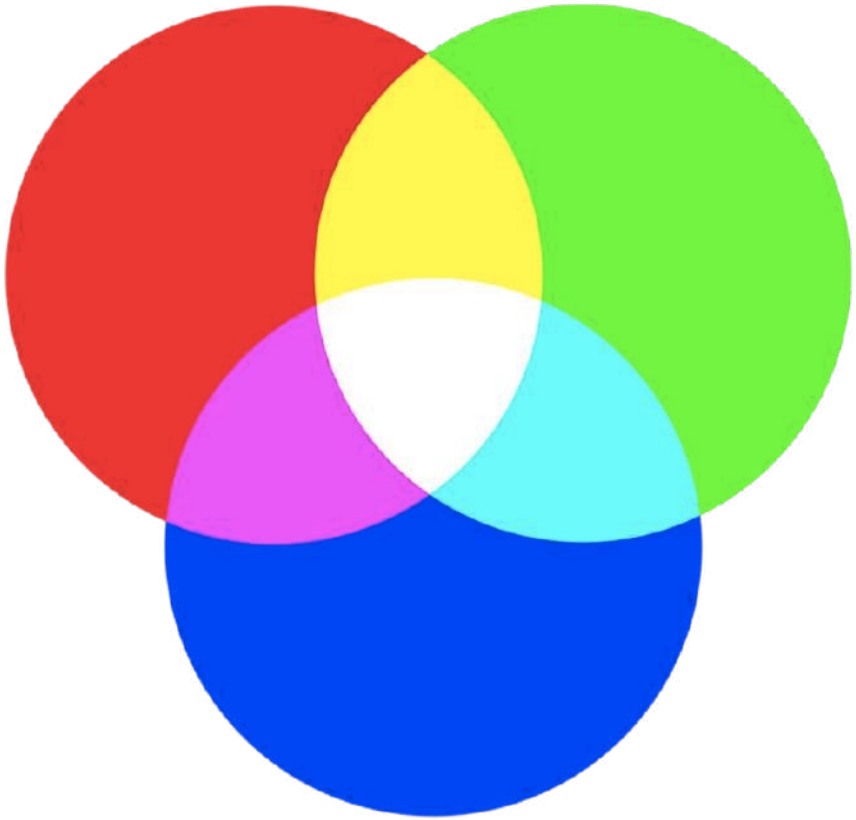
Изображение

Пиксели



- Изображение состоит из матрицы пикселей
- Каждый пиксель имеет свой цвет

Цветовая модель



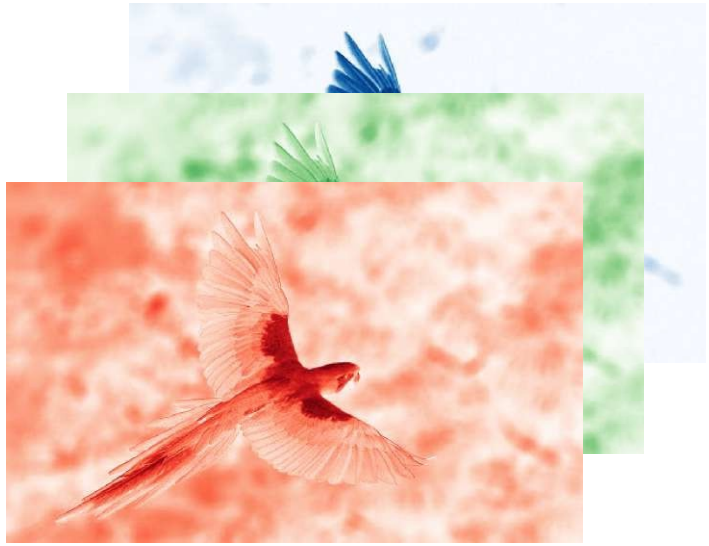
- Цвета пикселей представляются в виде комбинации красного (**R**), зеленого (**G**) и синего (**B**) цветов
- Такой способ называется RGB цветовой моделью

Примеры

(0, 0, 0)	(150, 150, 150)	(255, 255, 255)
(255, 0, 0)	(0, 255, 0)	(0, 0, 255)
(255, 64, 255)	(255, 252, 121)	(148, 23, 81)

Разложение изображений

Каналы изображений



=

88	31	1	22
42	195	13	28
130	85	43	203
18	75	122	187
241	61	7	91
65	27	155	101

- Изображение представляется в виде трех матриц
- Каждая матрица соответствует одному из цветовых каналов

Свёртки изображений

Пример

- Связь между ближними пикселями больше
- Связь между далекими пикселями меньше

Пример

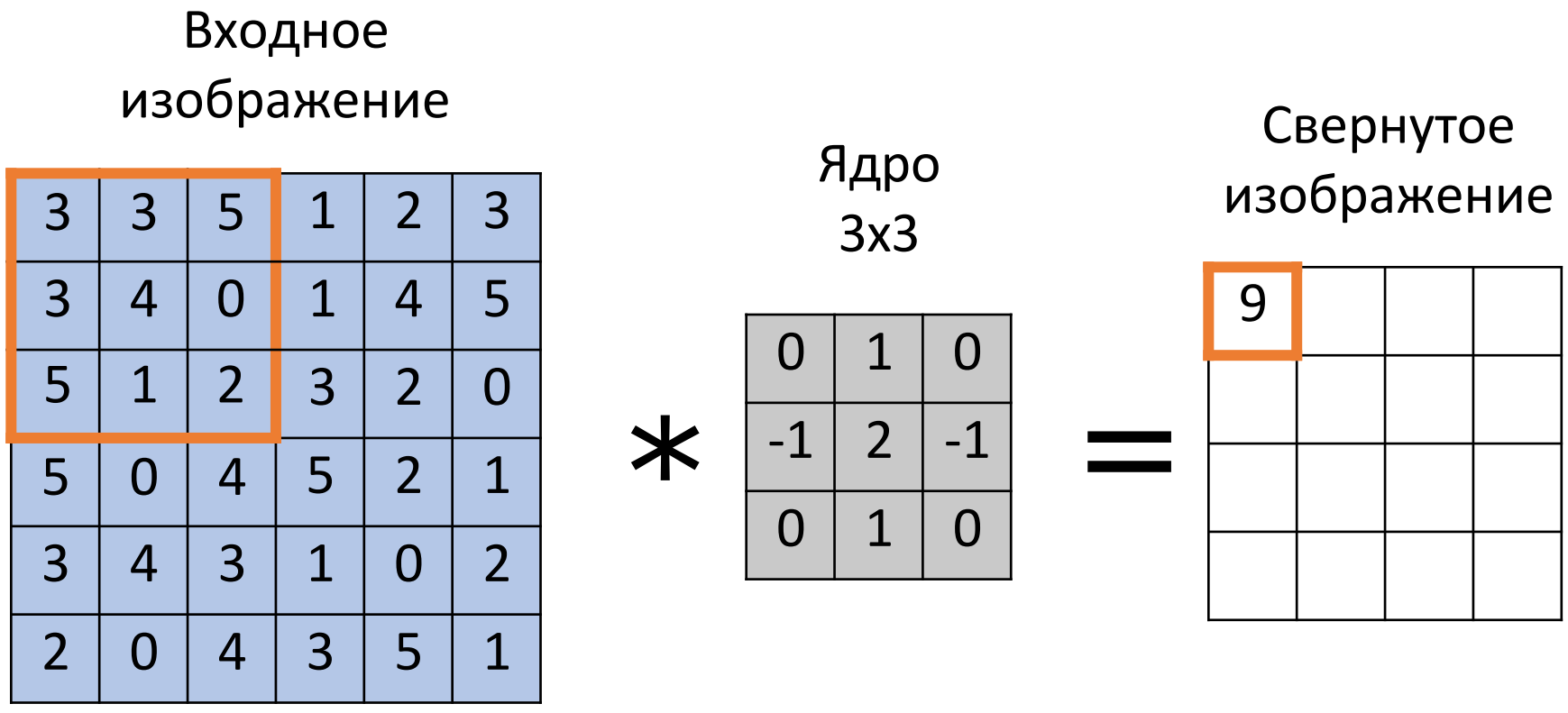
Рассмотрим изображение с одним каналом

=

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

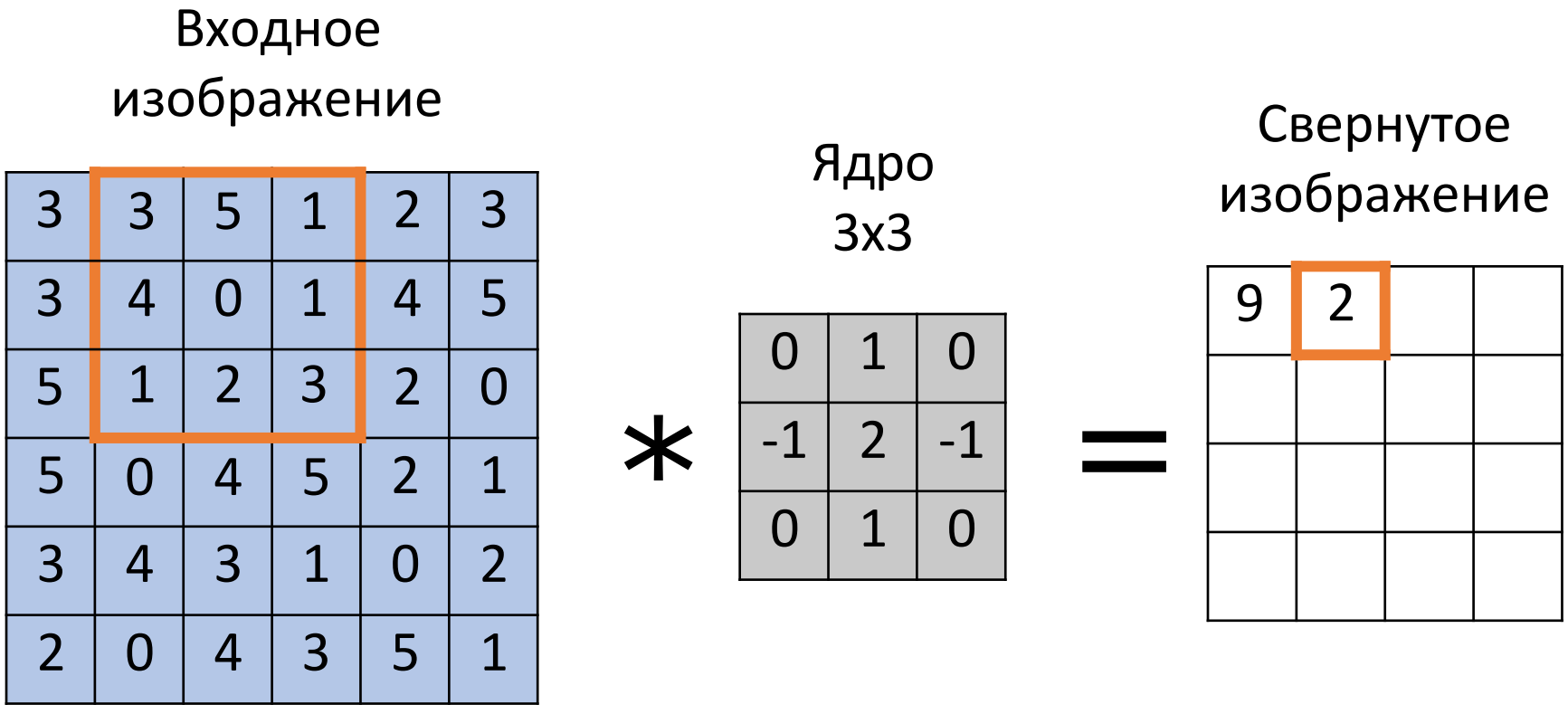
Как можем учесть локальную связь пикселей?

Операция свертки



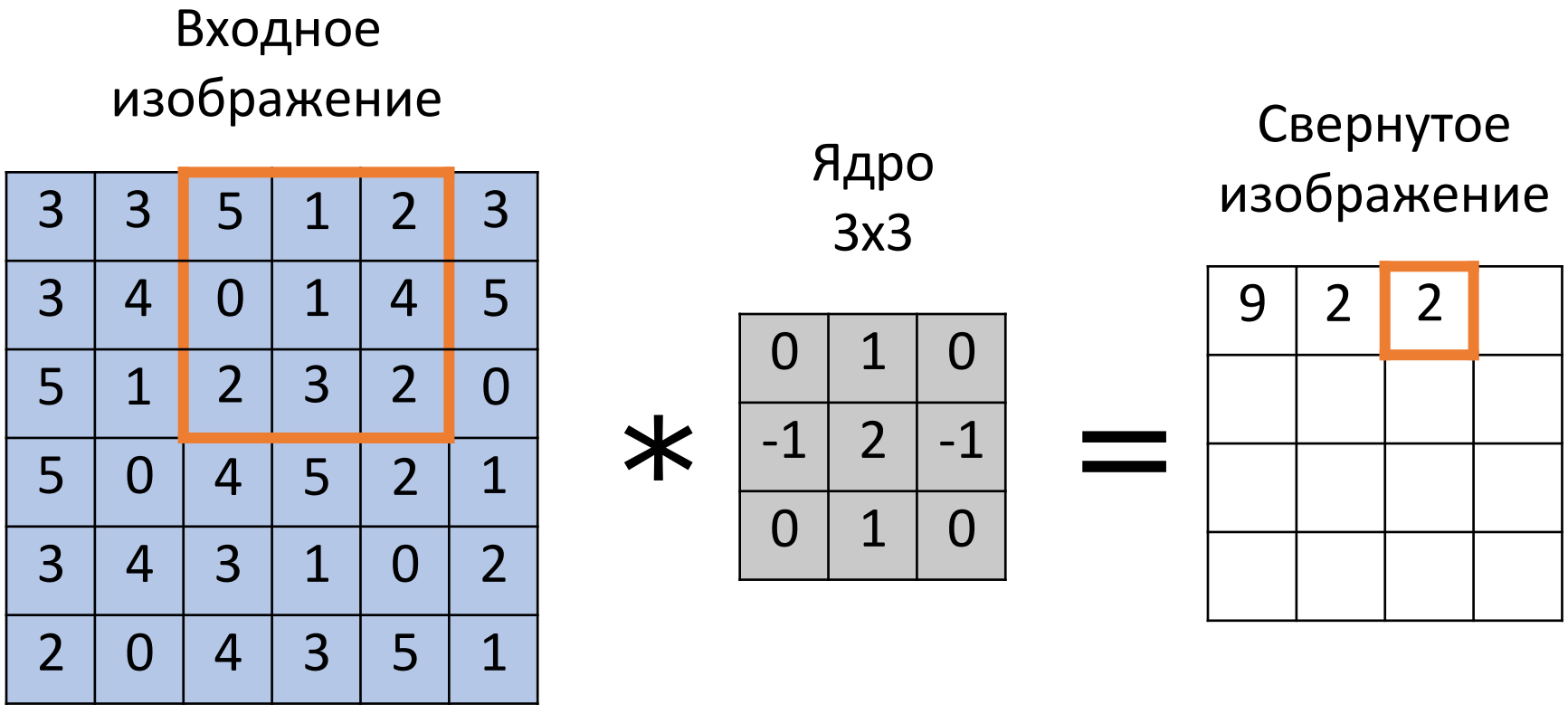
$$3 * 0 + 3 * 1 + 5 * 0 - 3 * 1 + 4 * 2 - 0 * 1 + 5 * 0 + 1 * 1 + 2 * 0 = 9$$

Операция свертки



$$3 * 0 + 5 * 1 + 1 * 0 - 4 * 1 + 0 * 2 - 1 * 1 + 1 * 0 + 2 * 1 + 3 * 0 = 2$$

Операция свертки



$$5 * 0 + 1 * 1 + 2 * 0 - 0 * 1 + 1 * 2 - 4 * 1 + 2 * 0 + 3 * 1 + 2 * 0 = 2$$

Операция свертки

Входное изображение

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

$*$

Ядро 3x3

0	1	0
-1	2	-1
0	1	0

$=$

Свернутое изображение

9	2	2	6
-1	4	8	7
-4	8	8	0
2	9	7	4

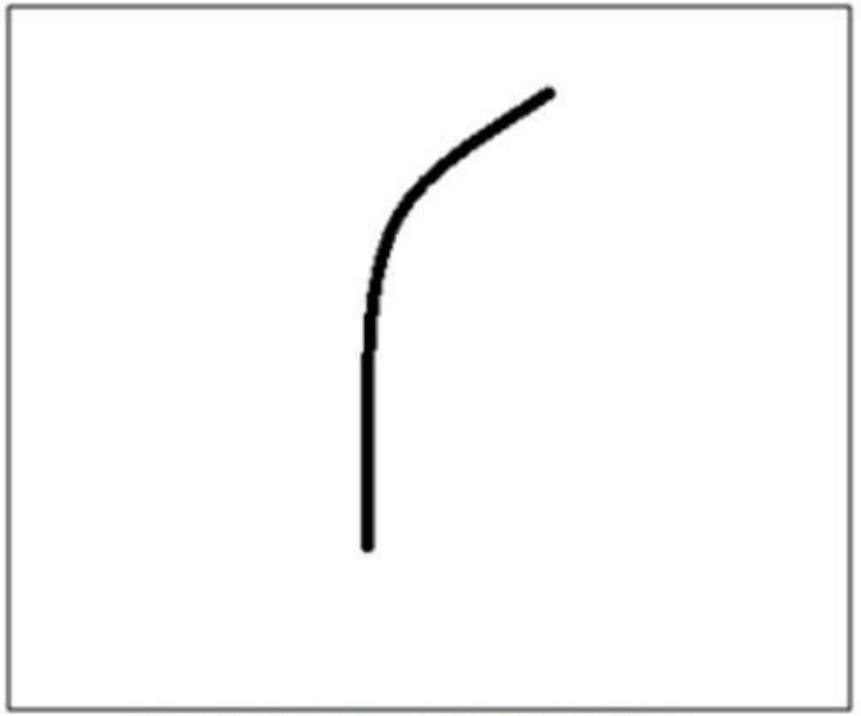
Свертка — это процесс сложения соседних элементов изображения, взвешиваемых ядром

Свёртка для поиска паттернов

- Свёртка – это фильтр, накладываемый на изображение. Фильтр помогает детектировать участки изображения с некоторыми свойствами.

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

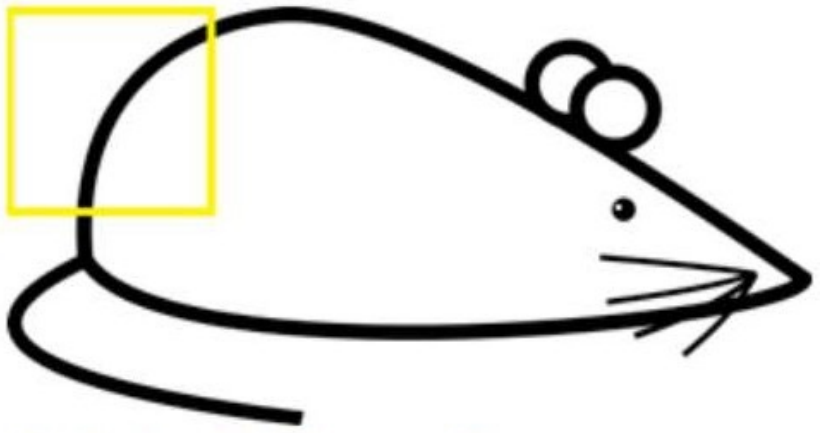
Pixel representation of filter



Visualization of a curve detector filter

Свёртка для поиска паттернов

Original image



Visualization of the filter on the image

https://habr.com/post/309508/

Свёртка для поиска паттернов

Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

Visualization of the filter on the image

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

Pixel representation of receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = 0

Свёртка для преобразования изображений

Фильтр

-1	-1	-1
-1	8	-1
-1	-1	-1

* =

Поиск краев

Входная картинка

На однотонной заливке дает ноль (черный)

Свёртка для преобразования изображений

Фильтр

-1	-1	-1
-1	8	-1
-1	-1	-1

*

=

Поиск краев

Входная картинка

0	-1	0
-1	5	-1
0	-1	0

*

=

Повышение резкости

Добавляет к картинке края

Свёртка для преобразования изображений

Входная картинка

Фильтр

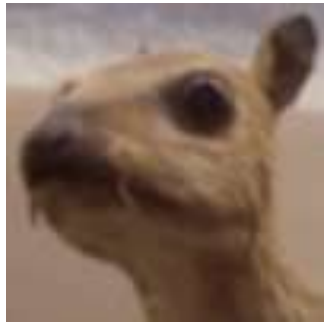
$$* \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$

Поиск краев

$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$

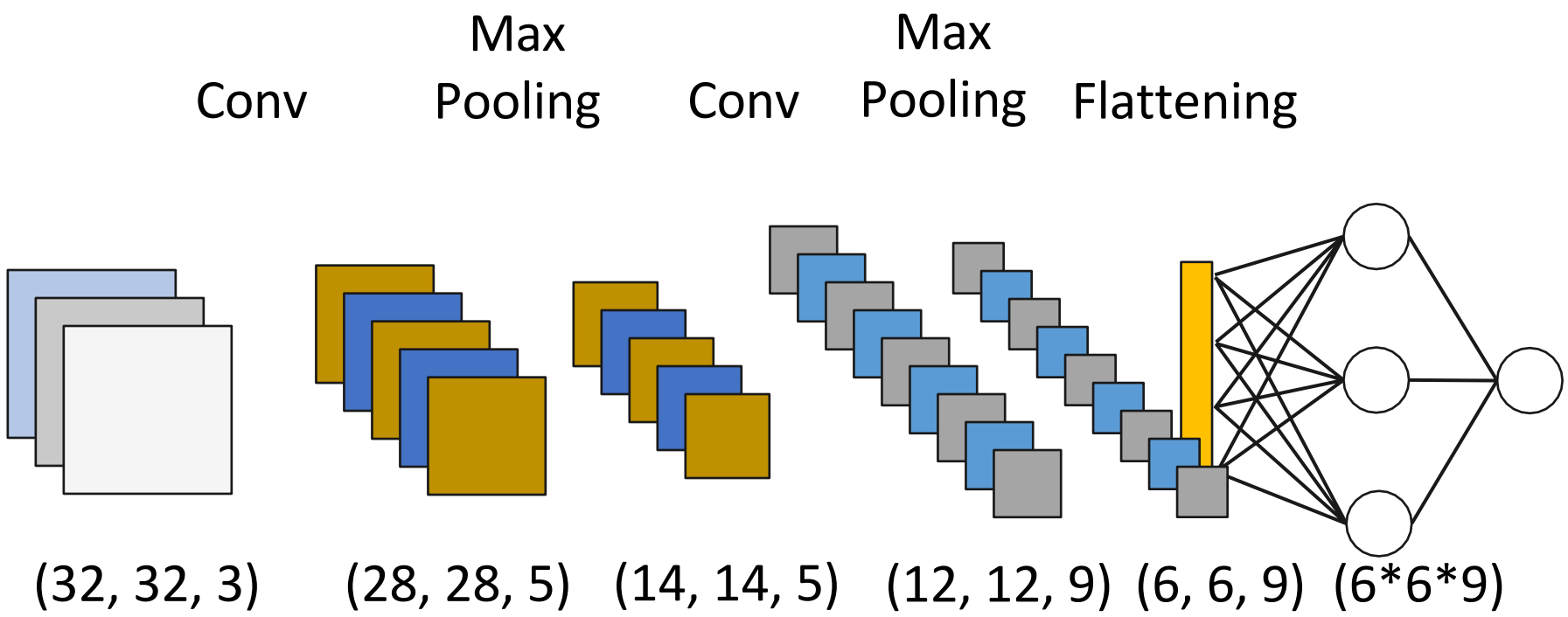
Повышение резкости

$$* \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$



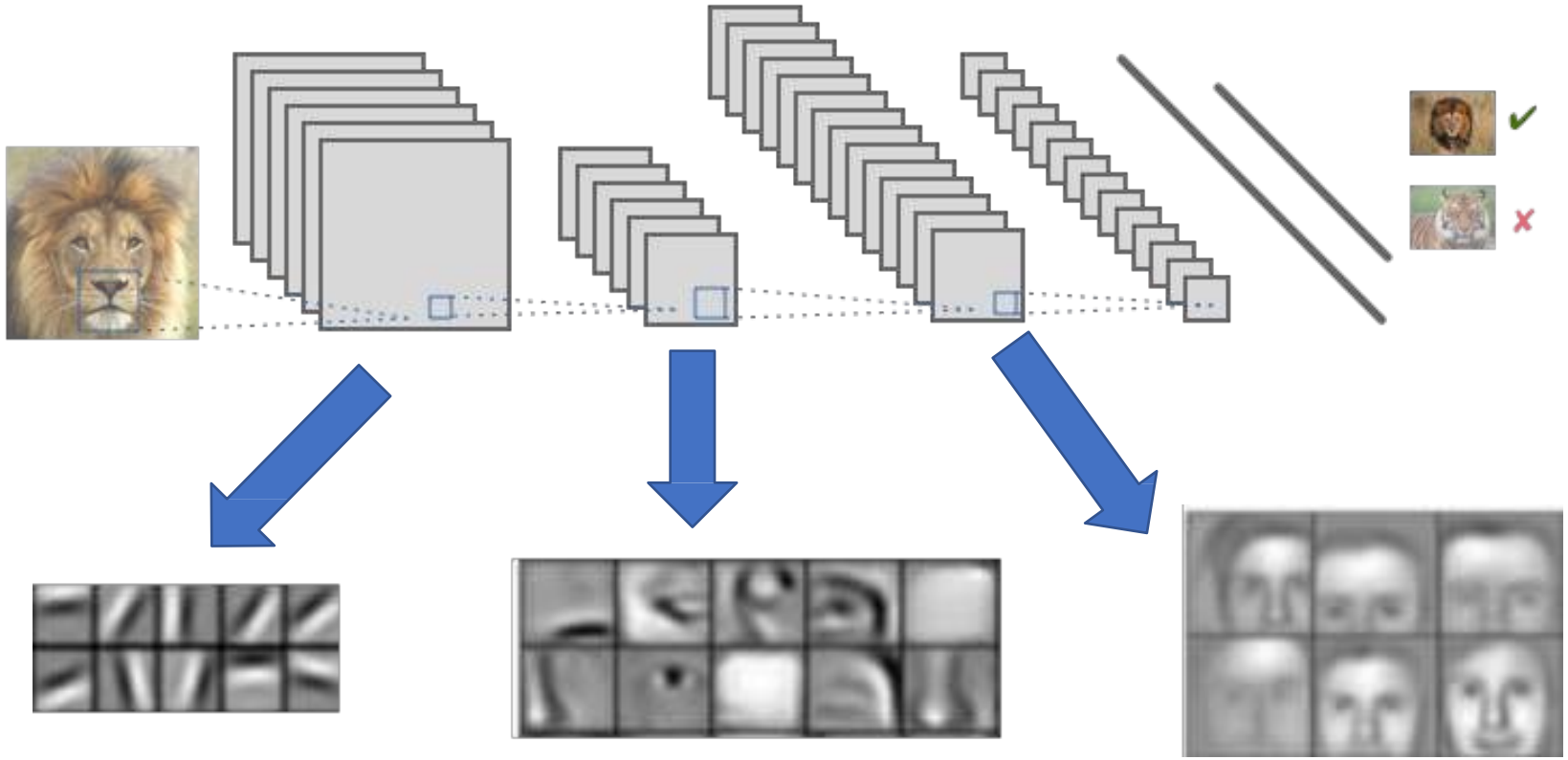
Размытие

Сверточные нейронные сети



Иерархические шаблоны

Глубокая нейронная сеть:



Имея достаточное количество обучающих примеров машина сама найдет все эти шаблоны в данных.

Рисуем лица

https://petapixel.com/2017/11/07/ai-creates-photo-realistic-faces-people-dont-exist/

Перенос стиля

Картинка | Стиль | Перенос стиля

https://github.com/luanfujun/deep-photo-styletransfer