

Машинное обучение

Лекция 11
Нейронные сети

Михаил Гущин
mhushchyn@hse.ru

НИУ ВШЭ, 2023



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

Машинный перевод

Яндекс Браузер

Яндекс.Браузер обновился. Версия 21.8.2

1

Hi! I'm David and I lead the NLP team at Yandex

Закадровый перевод видео с английского

Нейросети Яндекса научились сами переводить и озвучивать видео на английском языке. Пока — не везде, но уже скоро любой ролик на английском можно будет смотреть на русском.

Сразу попробовать новую функцию можно [по ссылке](#).

Как включить перевод видео?

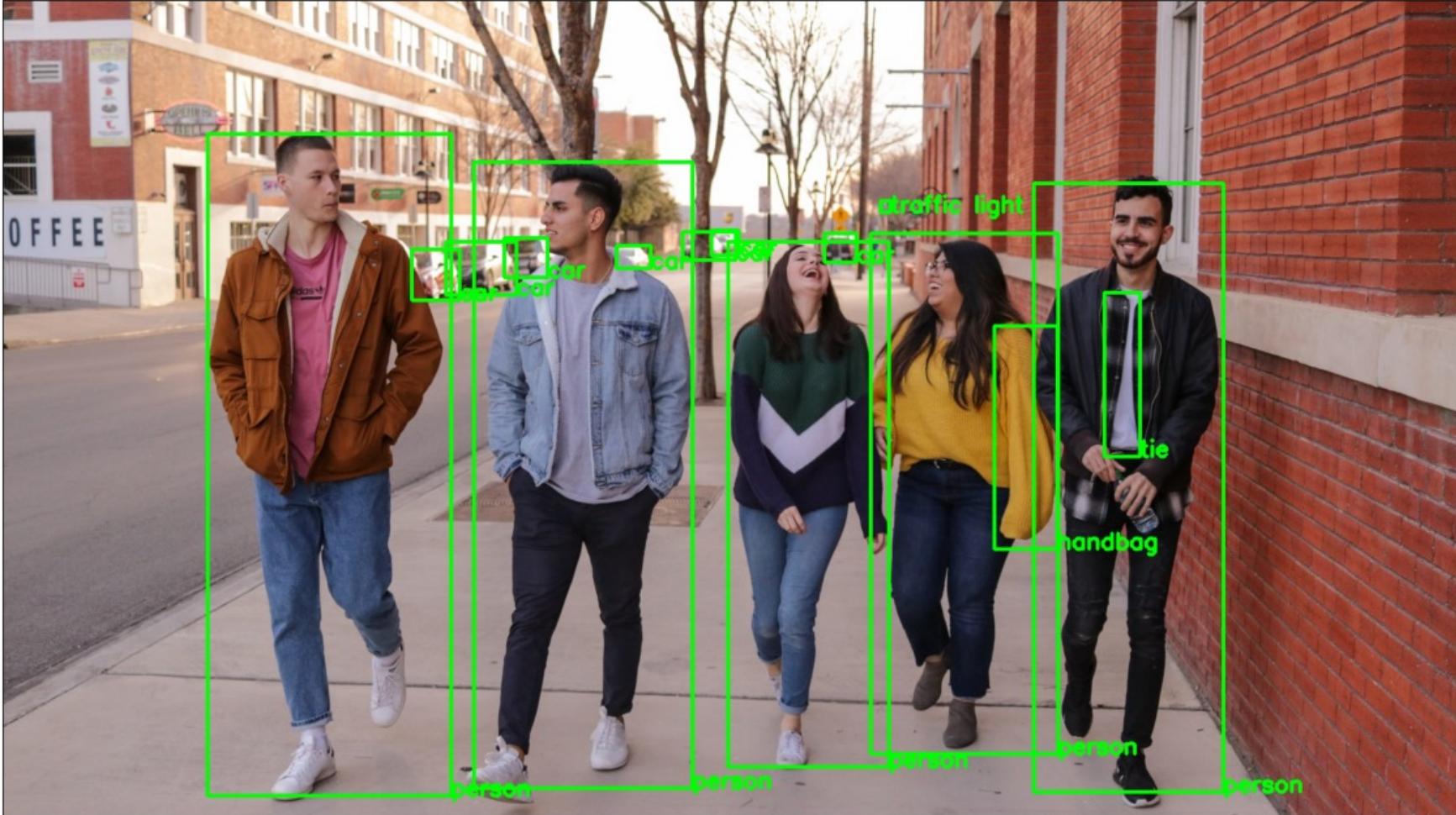
Подписывайтесь на новости Яндекс.Браузера:

Дзен ВКонтакте Твиттер Телеграм

Перевод не доступен для видео, у которых есть технические средства защиты авторских прав (DRM).

2

Анализ видео



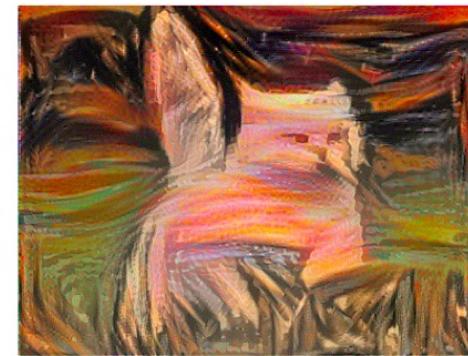
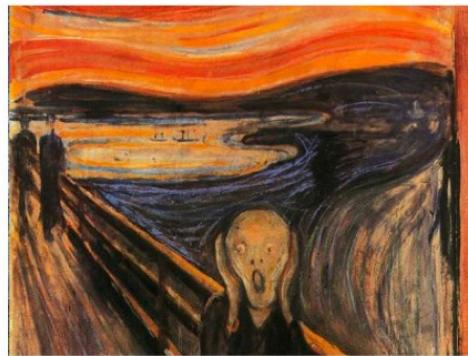
Перенос стиля изображения



+



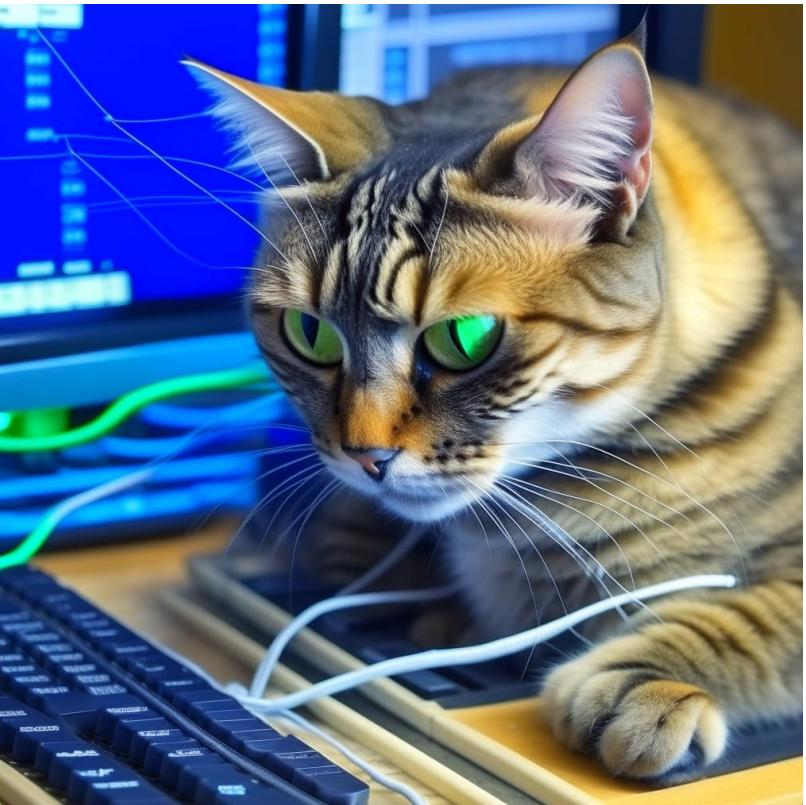
+



+



Генерация изображения по описанию



Кот фиксит баг в обучении
нейронной сети



Розовый фламинго стоит на
одной ноге в воде

Ссылка: <https://www.sberbank.com/promo/kandinsky>

Deepfake



<https://www.youtube.com/c/CtrlShiftFace>

Эволюция нейронных сетей

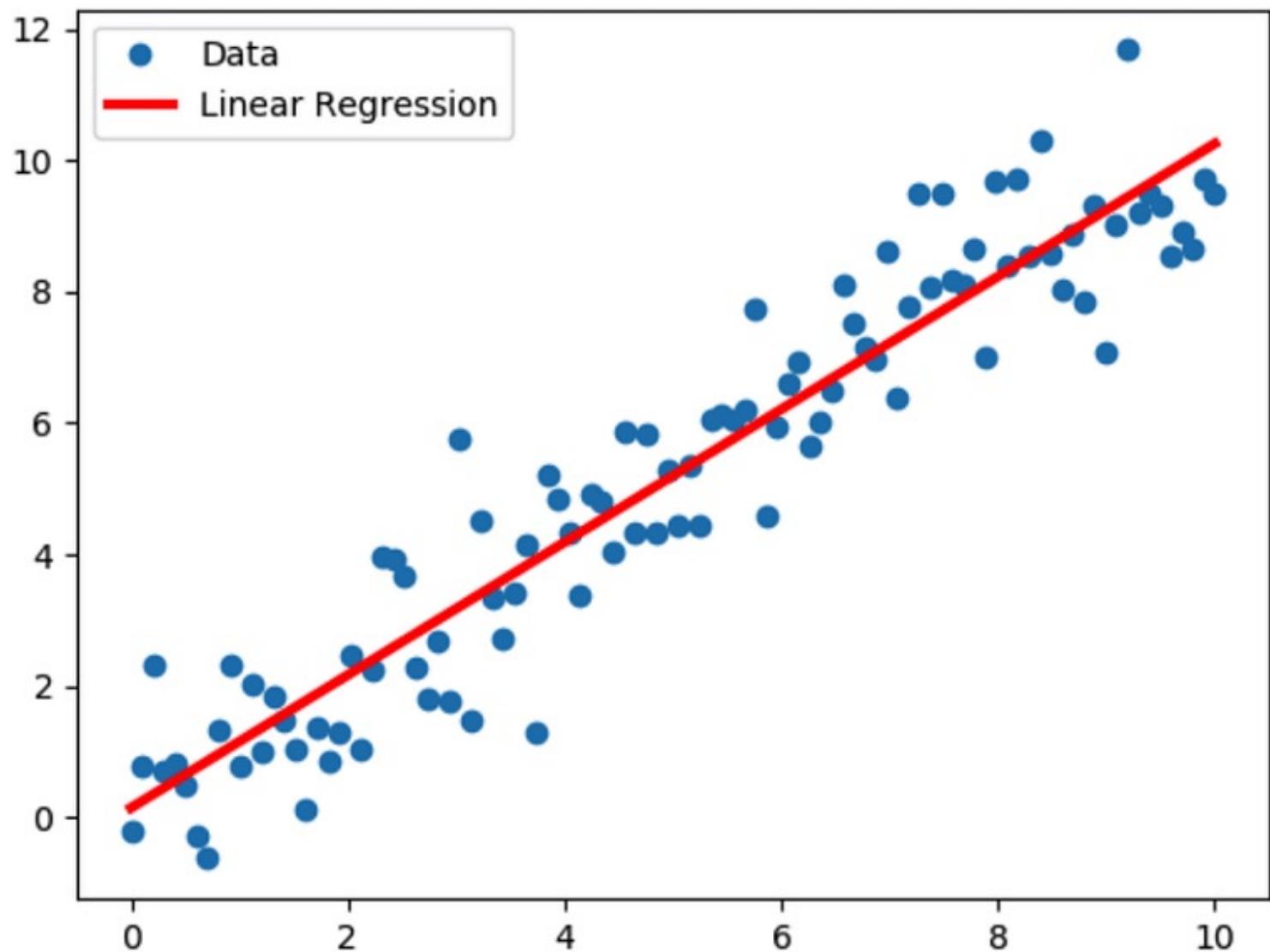


Линейная
регрессия

Логистическая
регрессия

Нейронная сеть

Линейная регрессия



Линейная регрессия

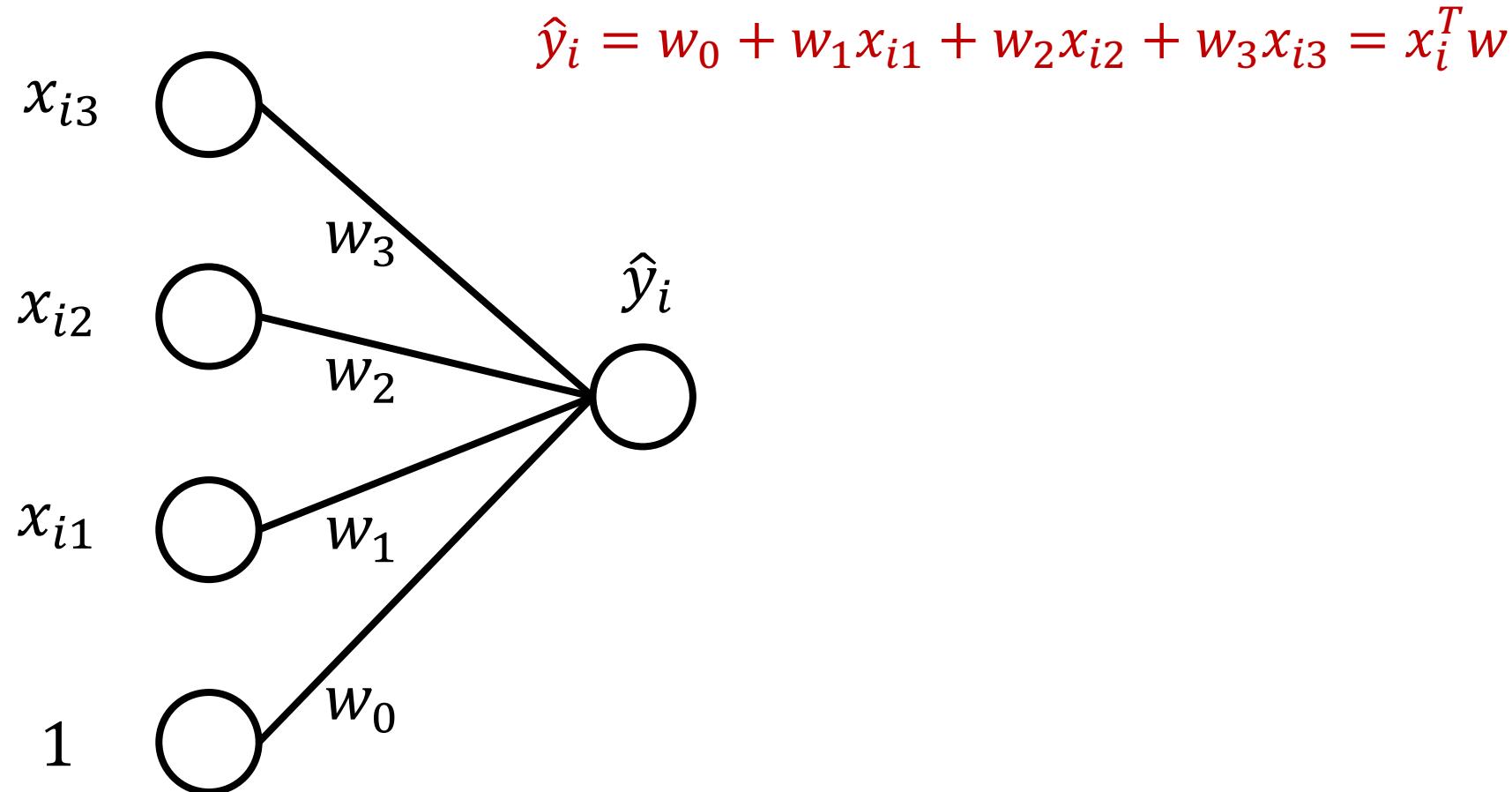
- ▶ Пусть дан набор наблюдений $\{x_i, y_i\}_{i=1}^N$, где $x_i \in R^3$, $y_i \in R$
- ▶ Модель линейной регрессии:

$$\hat{y}_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} = x_i^T w$$

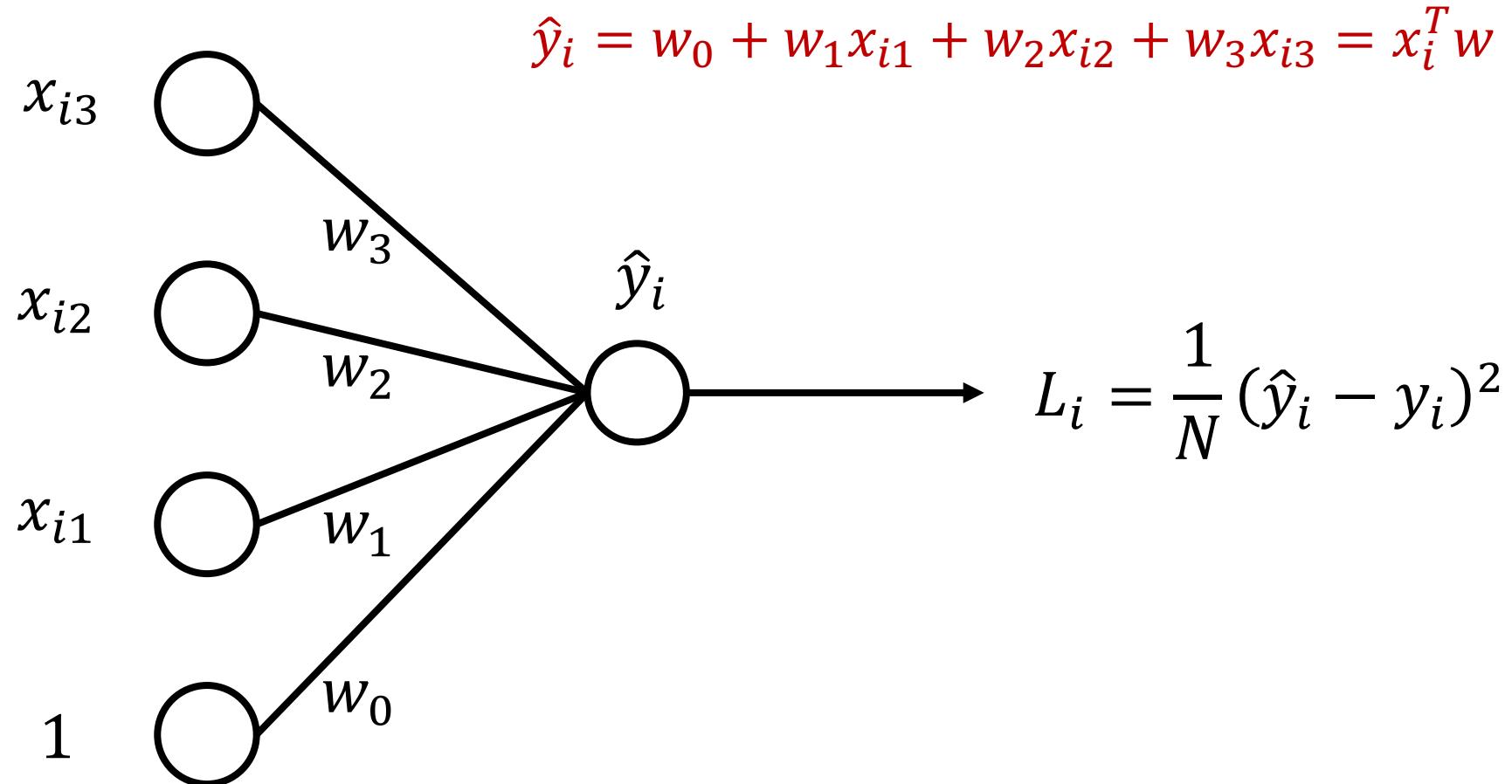
- ▶ Функция потерь:

$$L = \frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2 \rightarrow \min_{w_0, w_1, w_2, w_3}$$

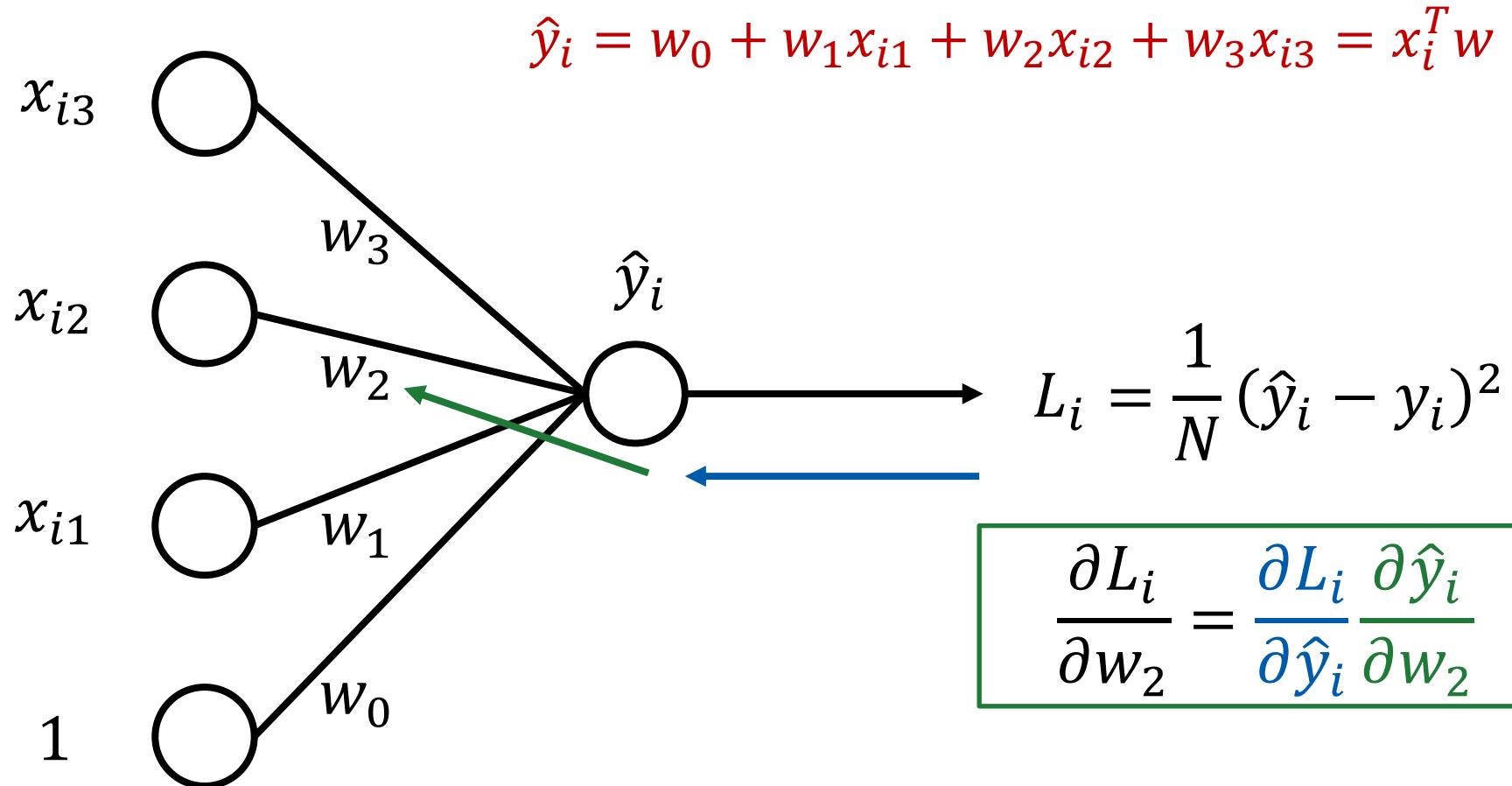
Линейная регрессия как граф вычислений



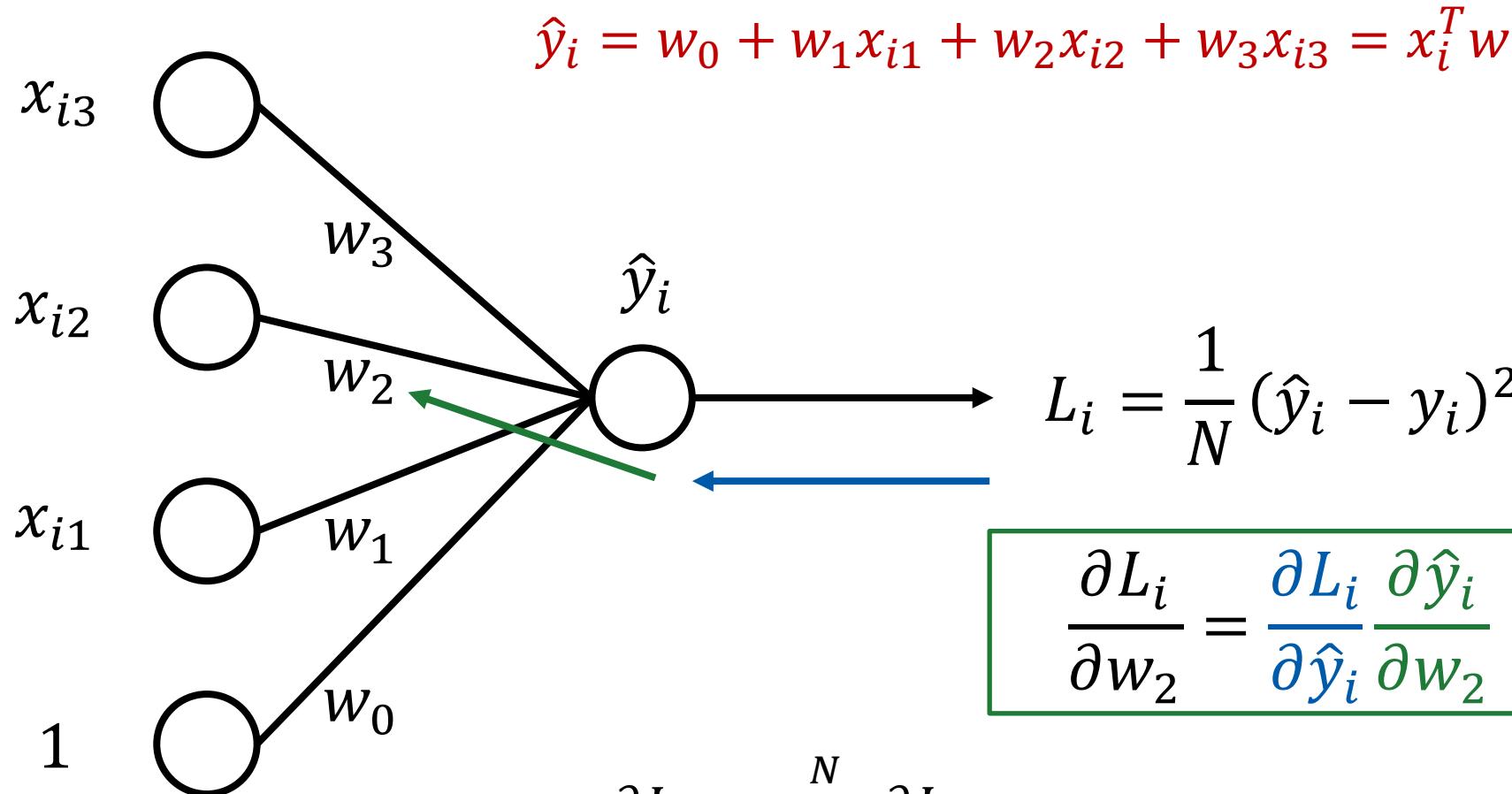
Линейная регрессия как граф вычислений



Градиент функции потерь



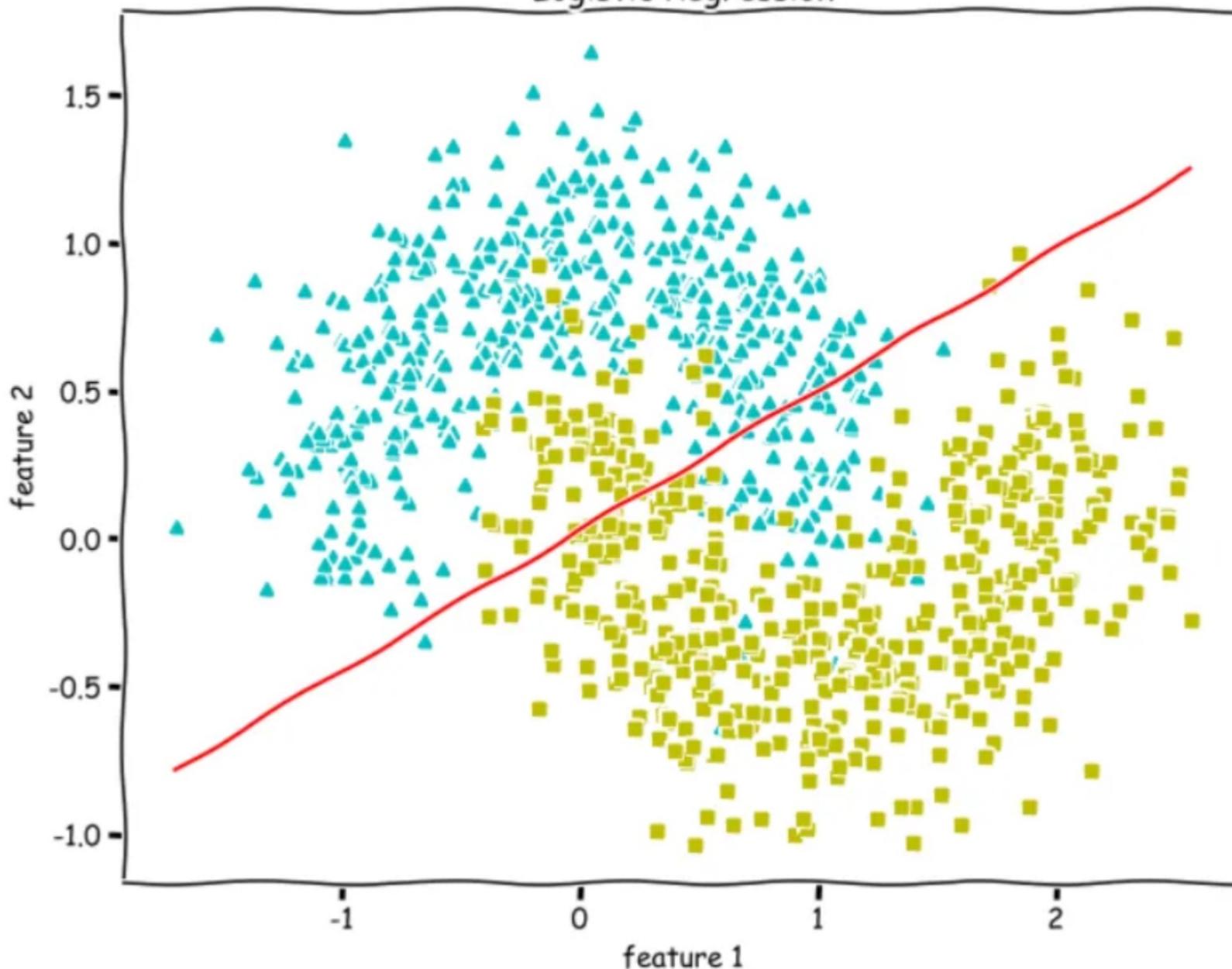
Градиентный спуск



$$w_2^{(t+1)} = w_2^{(t)} - \alpha \frac{\partial L}{\partial w_2}$$

Логистическая регрессия

Logistic Regression



Логистическая регрессия

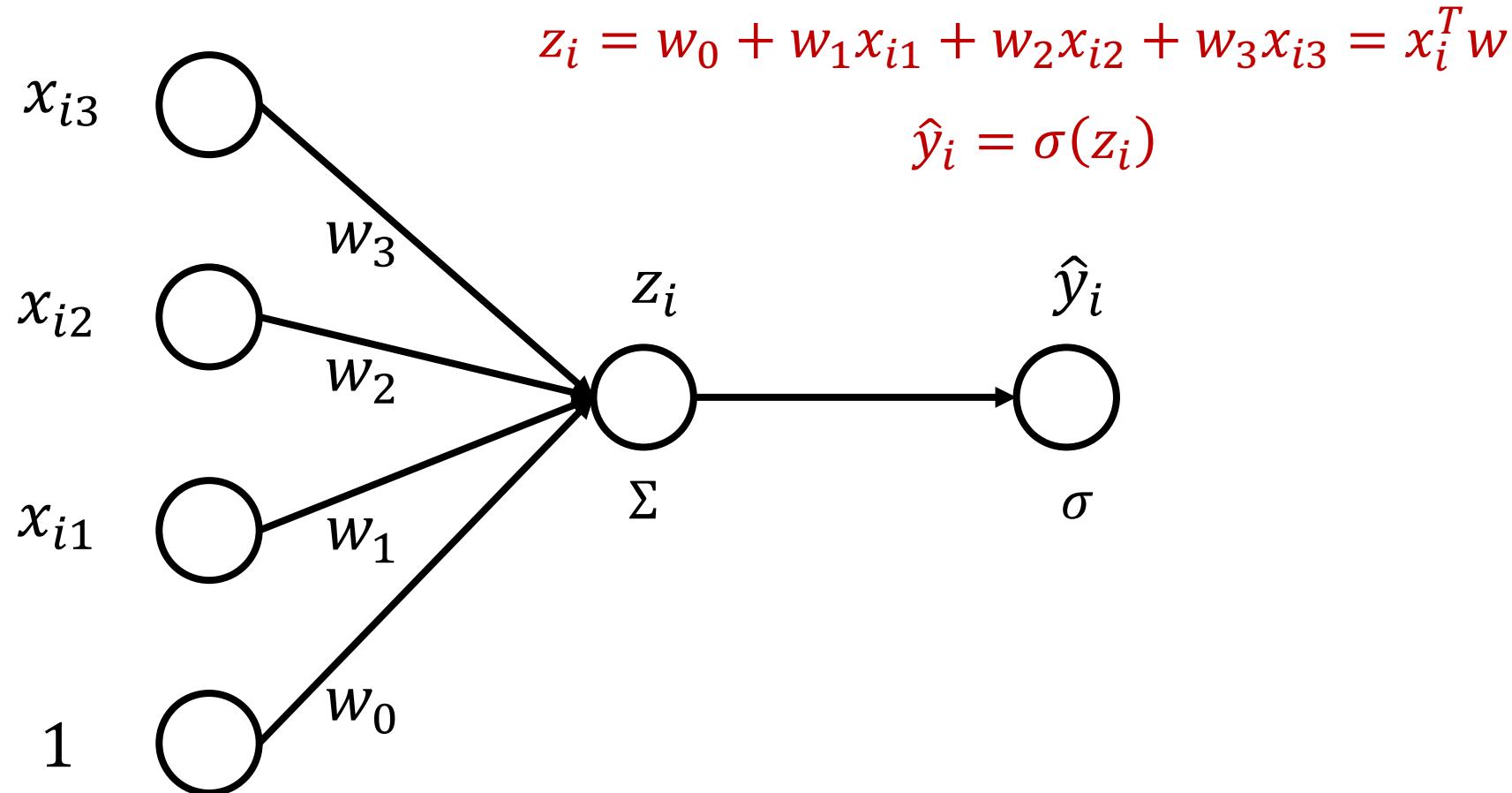
- ▶ Пусть дан набор наблюдений $\{x_i, y_i\}_{i=1}^N$, где $x_i \in R^3$, $y_i \in \{0, 1\}$
- ▶ Модель логистической регрессии:

$$\hat{y}_i = \sigma(w_0 + w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3}) = \sigma(x_i^T w)$$

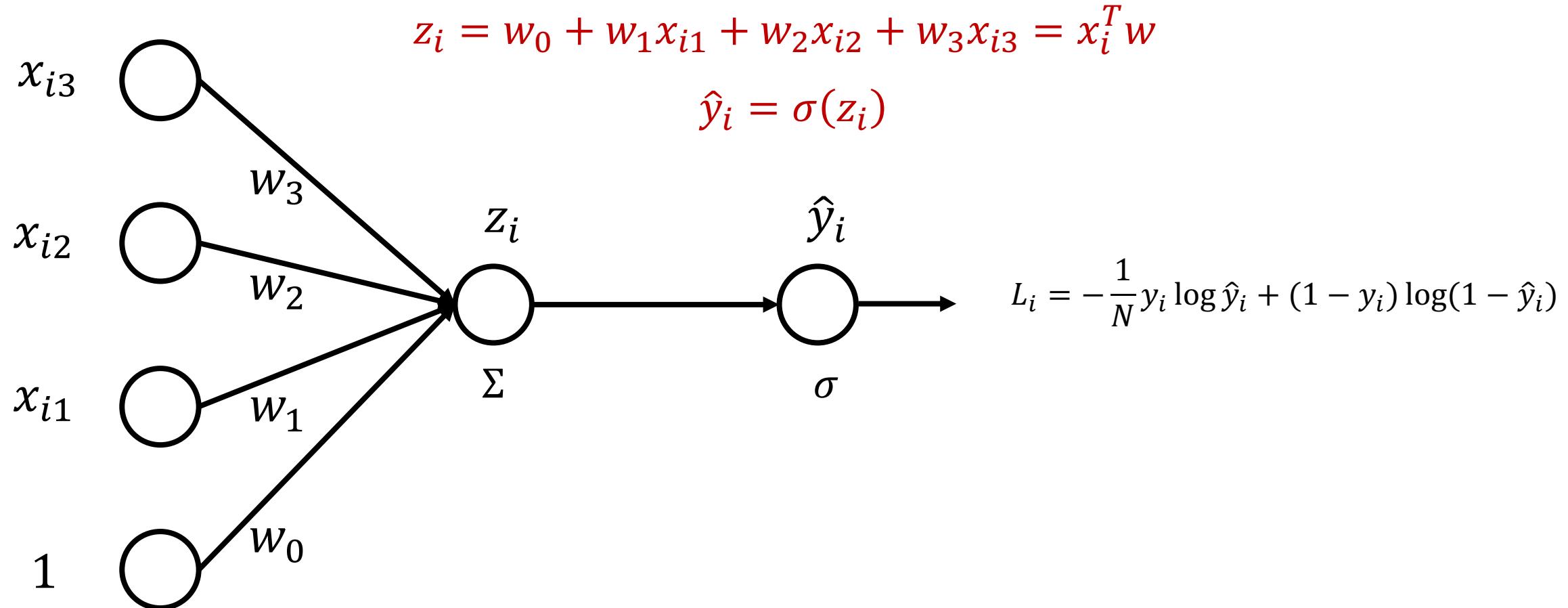
- ▶ Функция потерь:

$$L = -\frac{1}{N} \sum_{i=1}^N y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \rightarrow \min_{w_0, w_1, w_2, w_3}$$

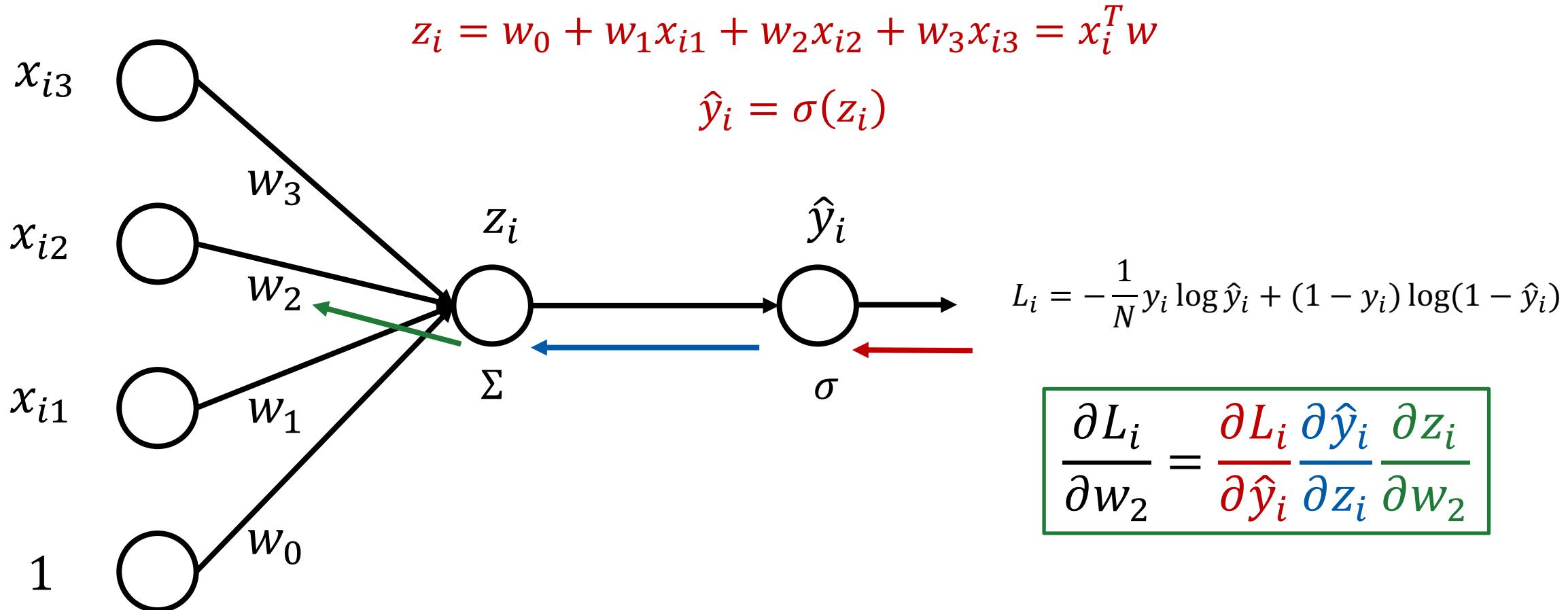
Логистическая регрессия как граф вычислений



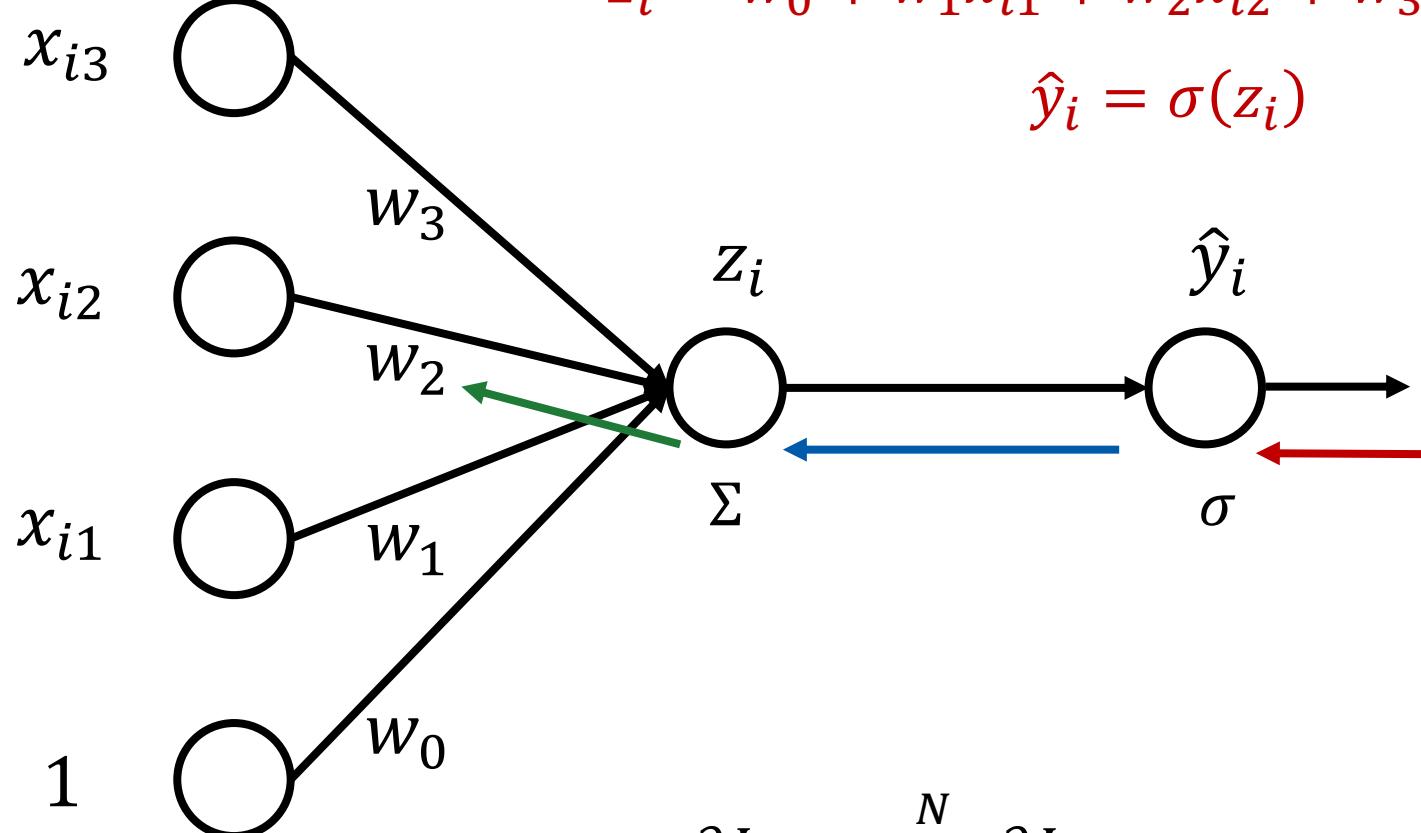
Логистическая регрессия как граф вычислений



Градиент функции потерь



Градиентный спуск



$$z_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} = x_i^T w$$

$$\hat{y}_i = \sigma(z_i)$$

$$L_i = -\frac{1}{N} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

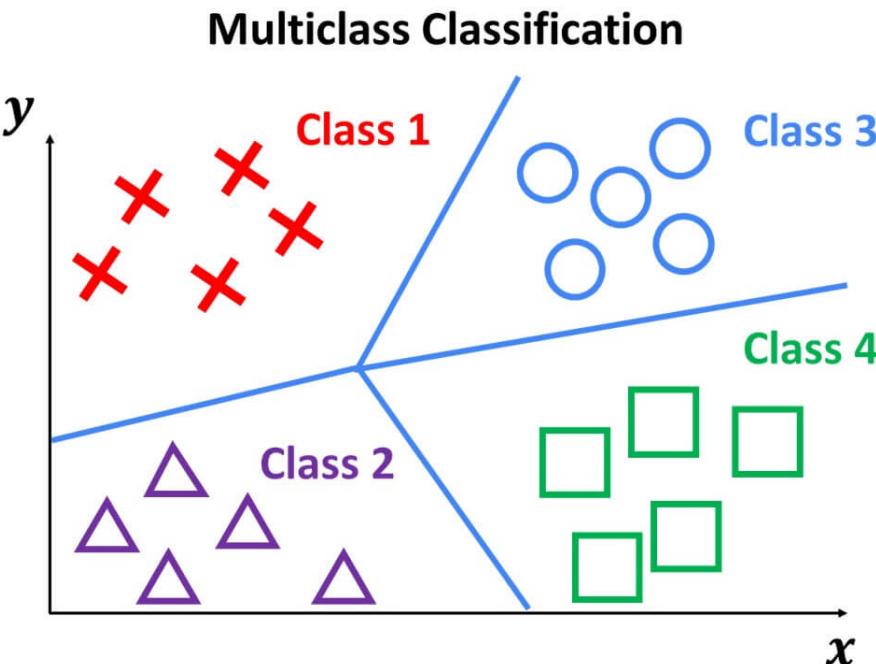
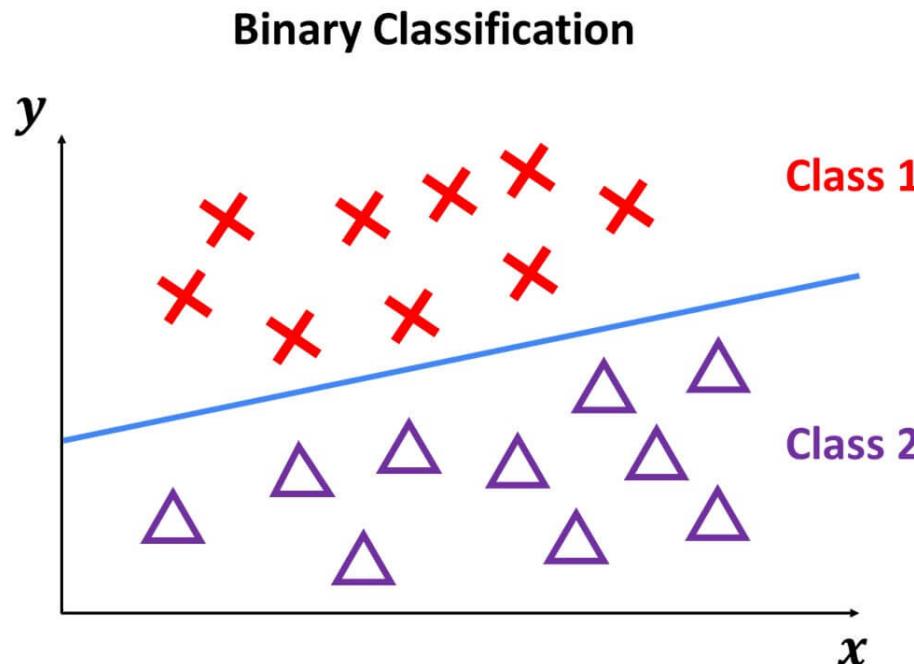
$$\boxed{\frac{\partial L_i}{\partial w_2} = \frac{\partial L_i}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_i} \frac{\partial z_i}{\partial w_2}}$$

$$\frac{\partial L}{\partial w_2} = \sum_{i=1}^N \frac{\partial L_i}{\partial w_2}$$

$$w_2^{(t+1)} = w_2^{(t)} - \alpha \frac{\partial L}{\partial w_2}$$

А что дальше?

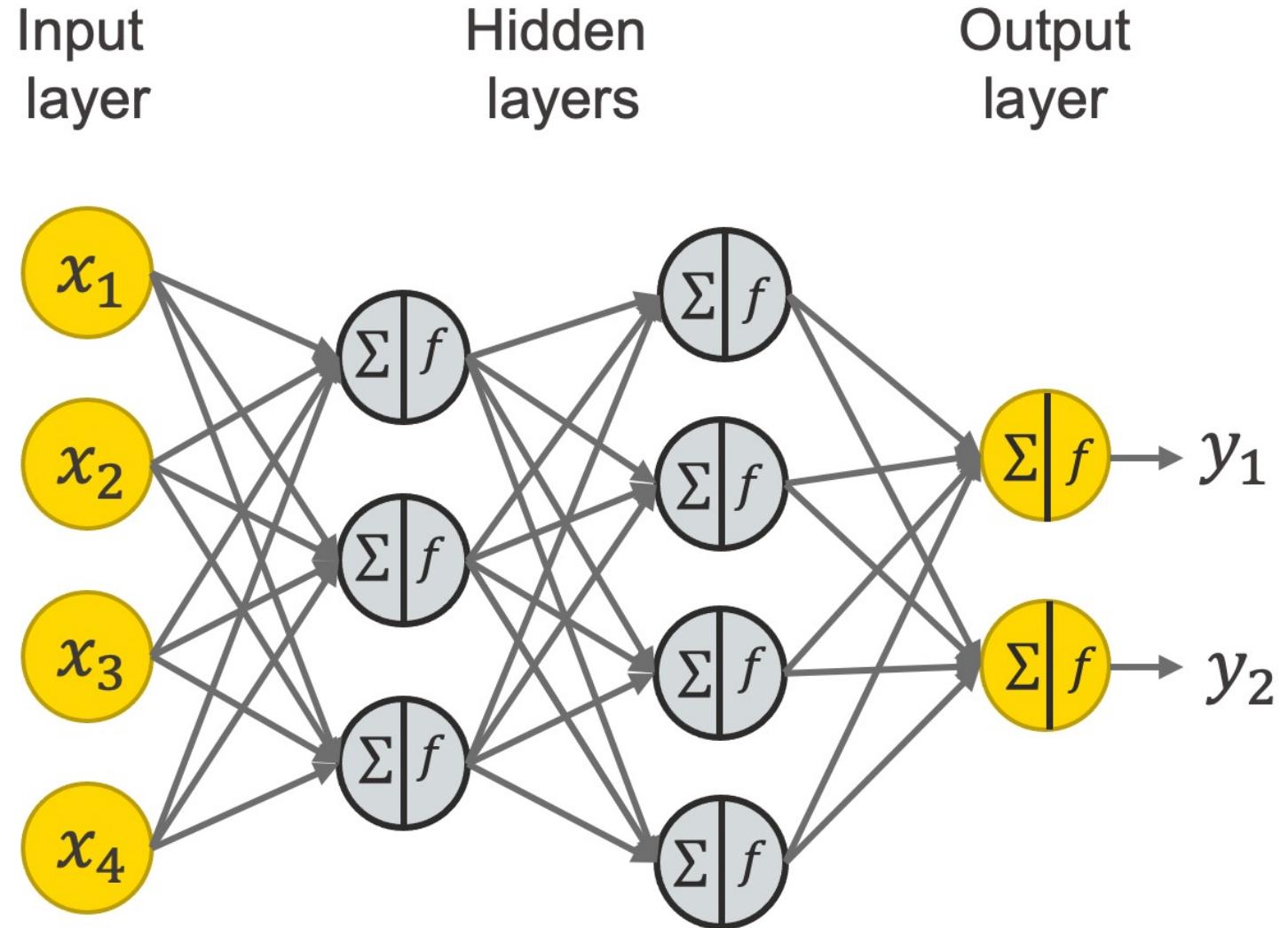
- ▶ Что делать, если у меня больше, чем два класса в данных?
- ▶ Что делать, если классы не разделяются линейной функцией?



Нейронные сети



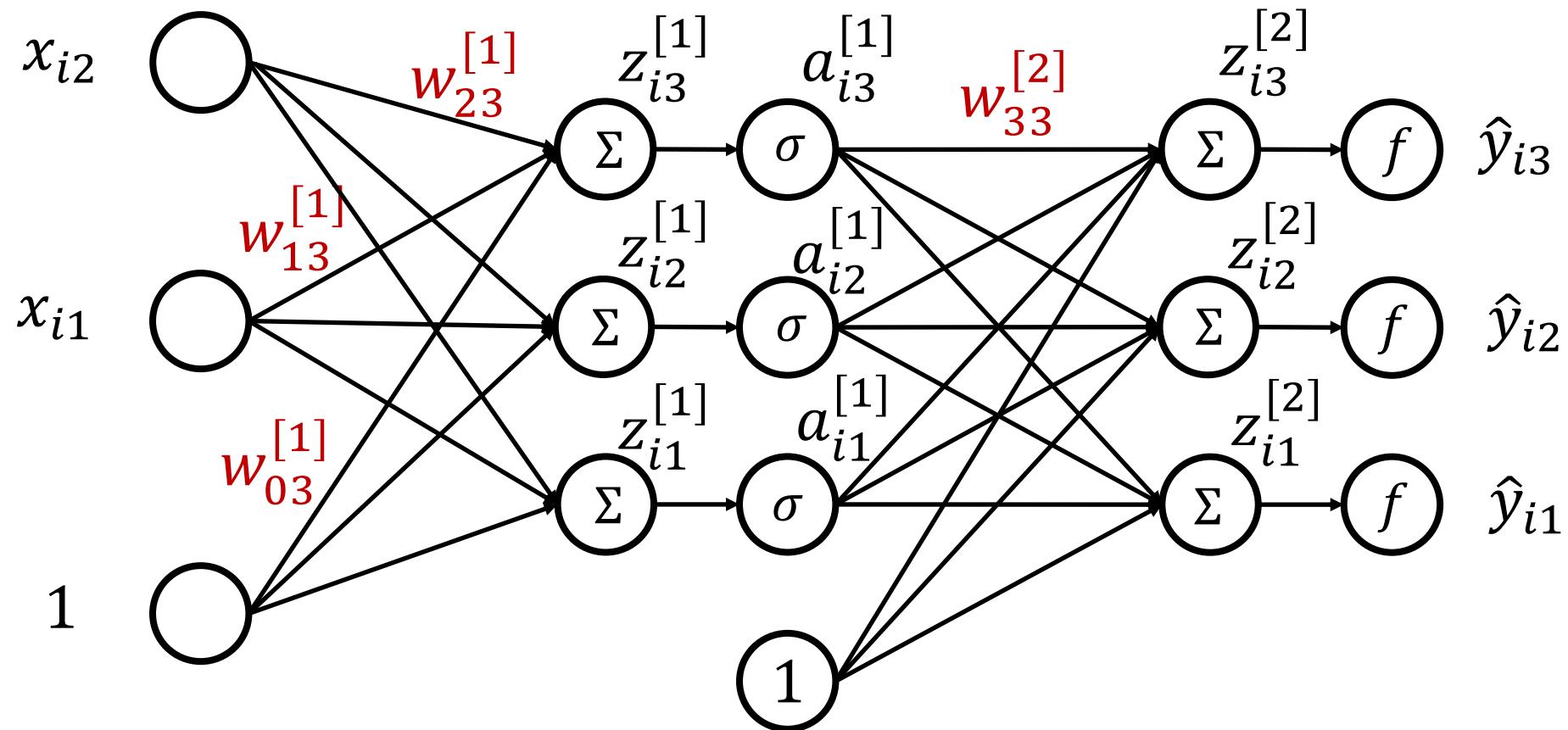
Нейронная сеть



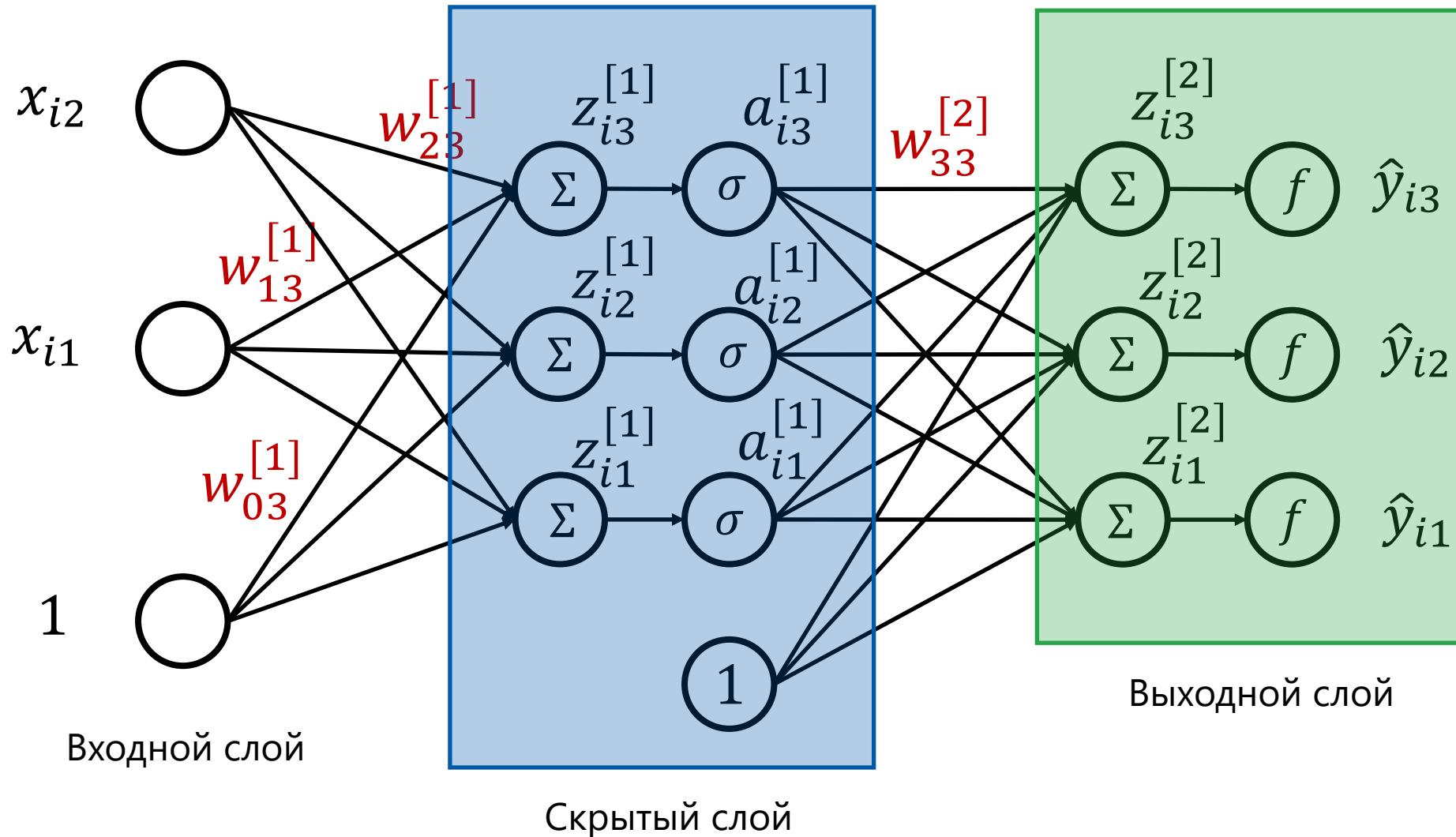
Классификация на три класса

- ▶ Пусть дан набор наблюдений $\{x_i, y_i\}_{i=1}^N$, где $x_i \in R^2$, $y_i \in \{0, 1, 2\}$
- ▶ Построим нейронную сеть

Нейронная сеть

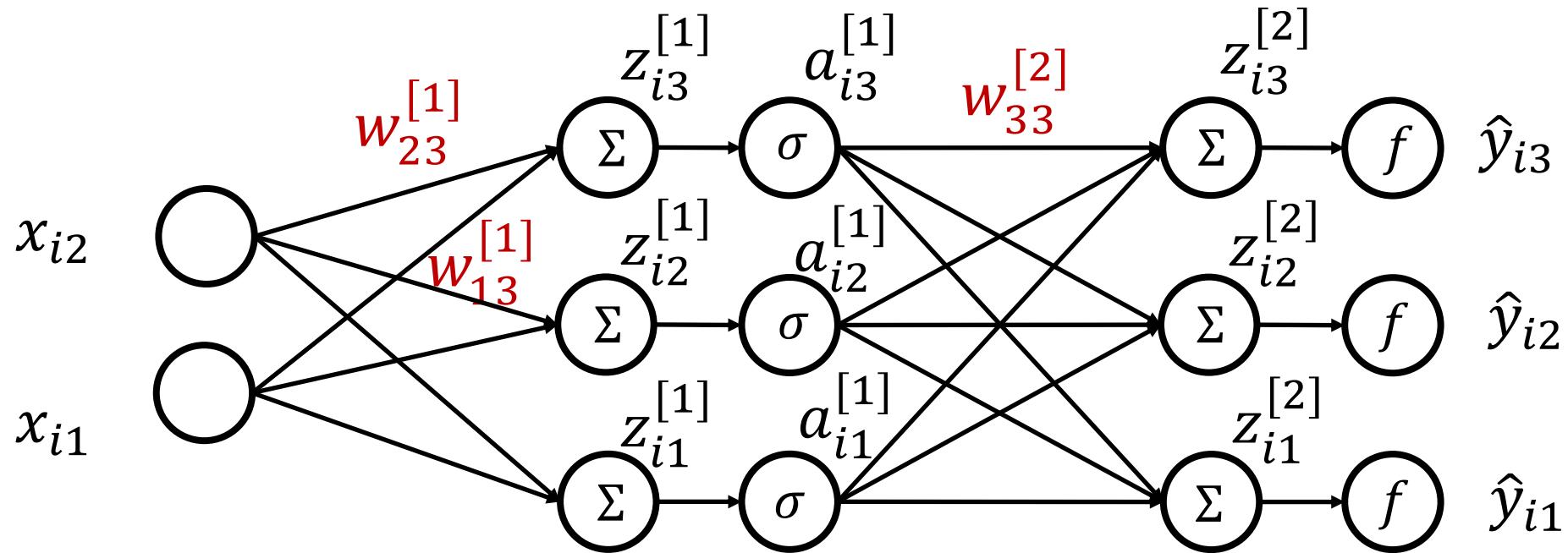


Нейронная сеть



Нейронная сеть

$$\hat{y}_i = f(\sigma^T(x_i^T W^{[1]}) W^{[2]})$$



Нейронная сеть

- ▶ Пусть дан набор наблюдений $\{x_i, y_i\}_{i=1}^N$, где $x_i \in R^2$, $y_i \in \{0, 1, 2\}$
- ▶ Нейронная сеть

$$\hat{y}_i = f(\sigma^T(x_i^T W^{[1]}) W^{[2]})$$

$W^{[1]} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{pmatrix}$ - веса нейронов первого скрытого слоя;

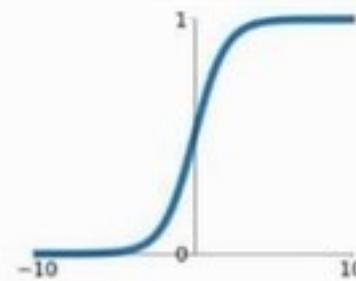
$W^{[2]} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix}$ - веса второго выходного (скрытого) слоя.

f, σ – функции активации.

ФУНКЦИИ АКТИВАЦИИ σ

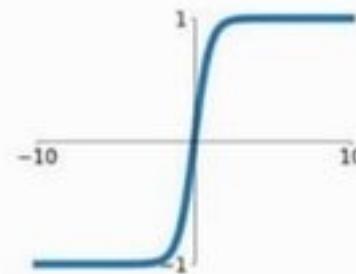
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



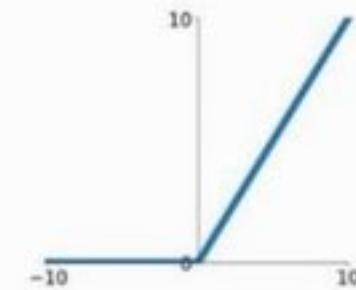
tanh

$$\tanh(x)$$



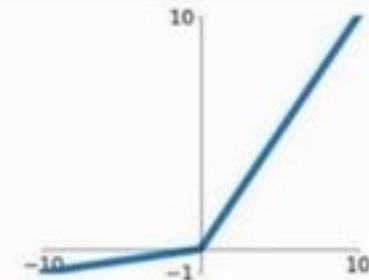
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

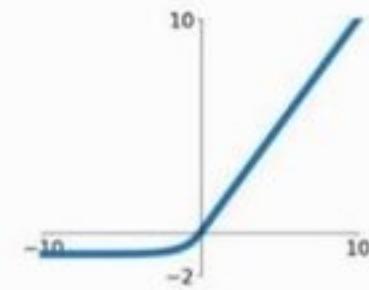


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

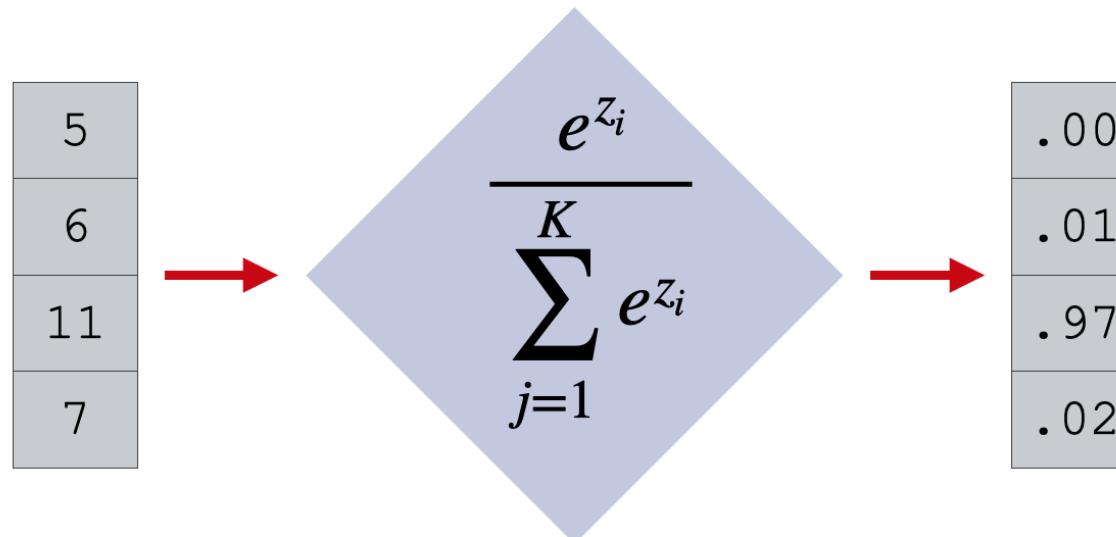
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Функции активации softmax

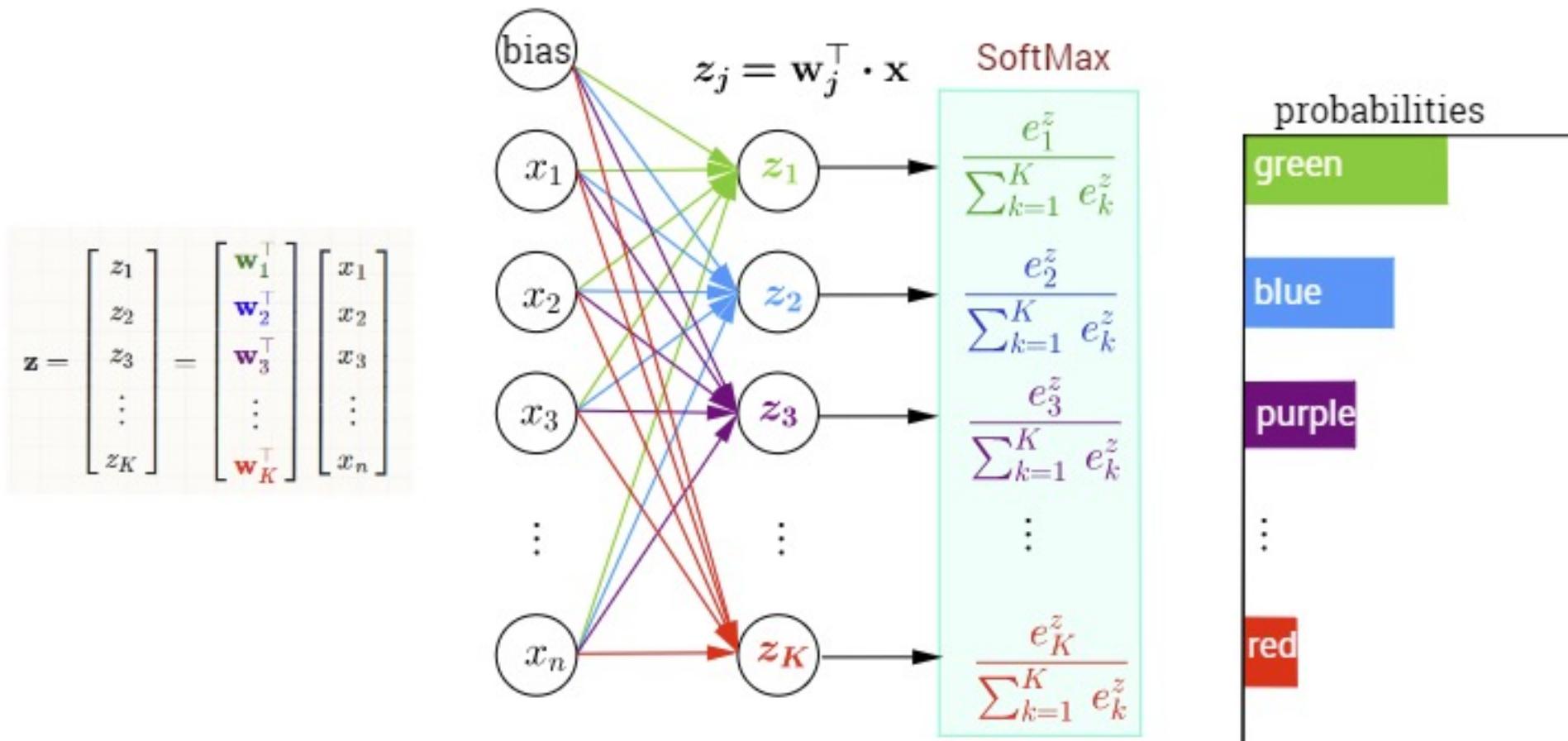
- ▶ Дан вектор z
- ▶ Хотим получить вектор вероятностей p , чтобы сумма равнялась 1.
- ▶ Используем функцию softmax:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$



ФУНКЦИИ АКТИВАЦИИ softmax

Multi-Class Classification with NN and SoftMax Function



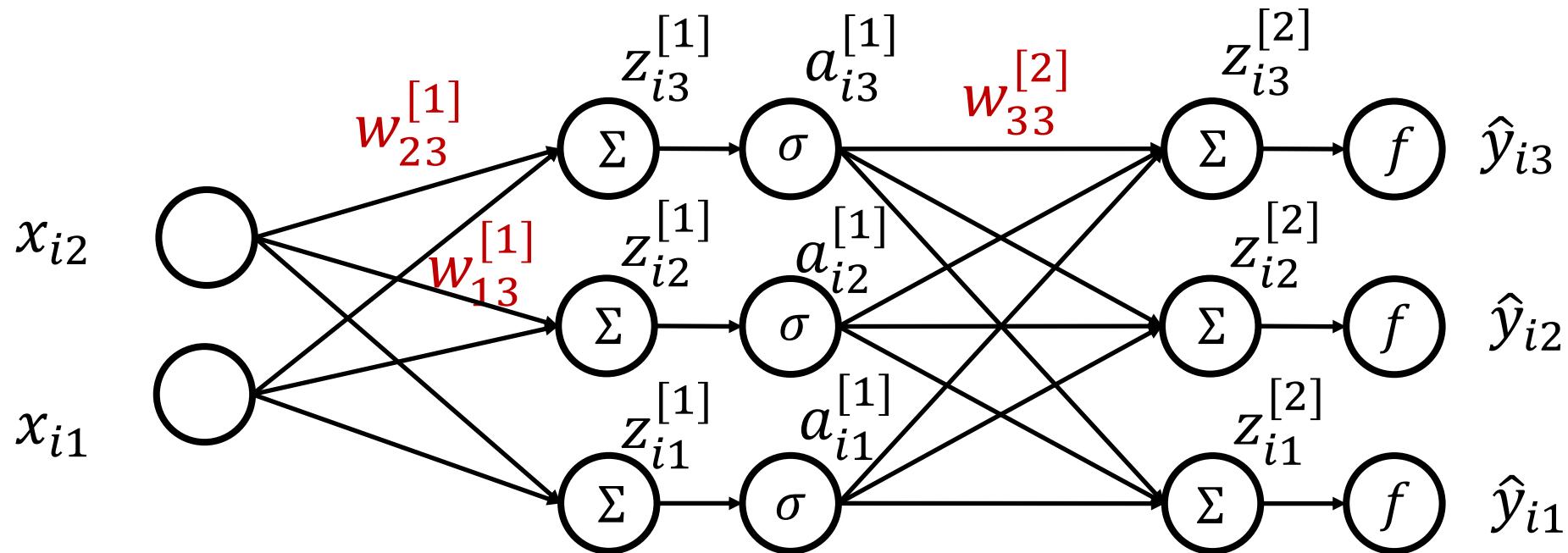
Источник: <https://rinterested.github.io/statistics/softmax.html>

Нейронная сеть

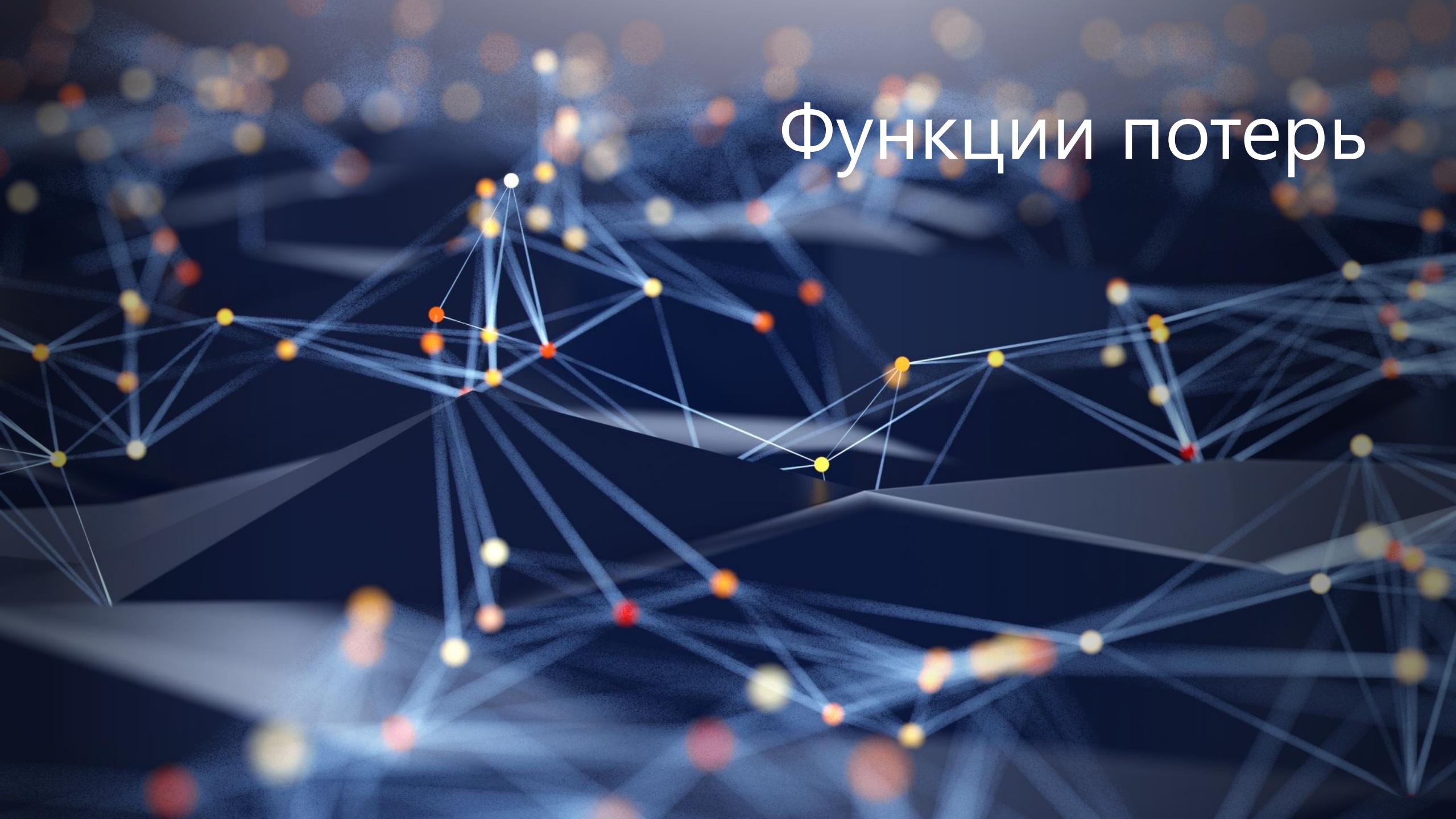
$$\hat{y}_i = f(\sigma^T(x_i^T W^{[1]}) W^{[2]})$$

σ – сигмоида,

f – softmax для классификации



Функции потерь



Функция потерь для классификации

- ▶ Функция потерь для 2 классов, где $y_i \in \{0, 1\}$:

$$L = -\frac{1}{N} \sum_{i=1}^N y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

- ▶ Функция потерь для K классов, где $y_i \in \{0, 1, 2, \dots, K\}$:

$$L = -\frac{1}{N} \sum_{i=1}^N \sum_{k=1}^K [y_i == k] \log \hat{y}_{ik}$$

\hat{y}_{ik} - прогноз вероятности для класса k

Функция потерь для классификации

Предсказания нейронной сети			
	пес	кот	жук
1	0.7	0.1	0.2
2	0.8	0.15	0.05
3	0.04	0.9	0.06
4	0.6	0.1	0.3
5	0.75	0.2	0.05

Правильные ответы Y	
	Y
1	кот
2	жук
3	кот
4	пес
5	кот

Кросс-энтропийный функционал

$$Q(w) = - \sum_{n=1}^N \log P(y_n | f(x_n, w))$$

$$\begin{aligned} & -\log 0.1 - \log 0.05 - \log \\ & 0.9 - \log 0.6 - \log 0.2 \end{aligned}$$

ФУНКЦИИ ПОТЕРЬ

- ▶ Классификация на 2 класса:

$$L = -\frac{1}{N} \sum_{i=1}^N y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

- ▶ Классификация на K классов, где $y_i \in \{0, 1, 2, \dots, K\}$:

$$L = -\frac{1}{N} \sum_{i=1}^N \sum_{k=1}^K [y_i == k] \log \hat{y}_{ik}$$

- ▶ Регрессия для y_i любой размерности:

$$L = \frac{1}{N} \sum_{i=1}^N \sum_{k=1}^K (\hat{y}_{ik} - y_{ik})^2$$

Обучение нейронных сетей

Градиентный спуск

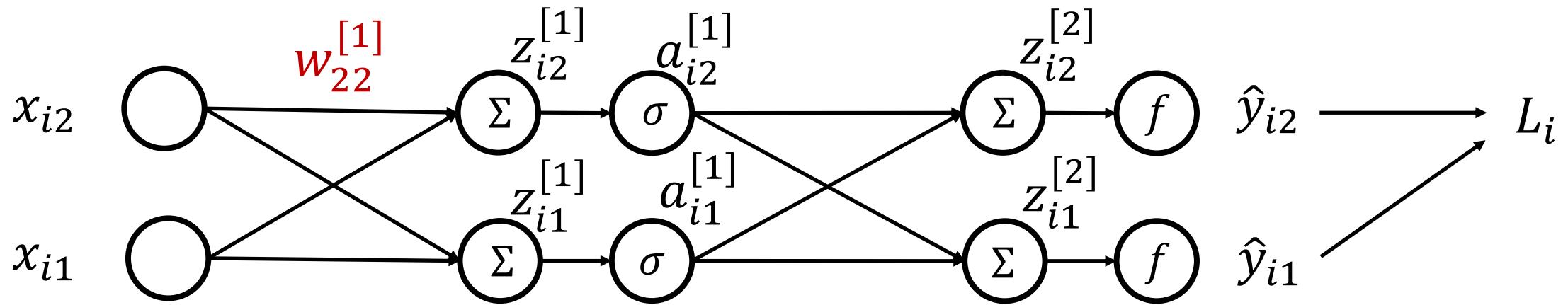
- ▶ Нейронные сети обучаем с помощью градиентного спуска

$$w = w - \alpha \frac{\partial L}{\partial w}$$

- ▶ Как посчитать градиент функции потерь по нужному весу сети?
- ▶ Будем использовать правило дифференцирования сложной функции
- ▶ Этот прием также называют обратным распространением ошибки (backpropagation)

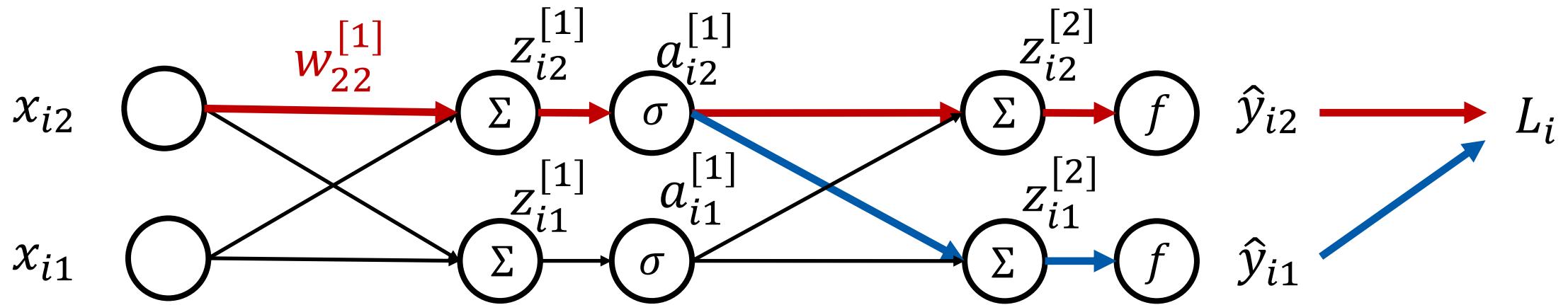
Нейронная сеть

$$\hat{y}_i = f(\sigma^T(x_i^T W^{[1]}) W^{[2]})$$



Нейронная сеть

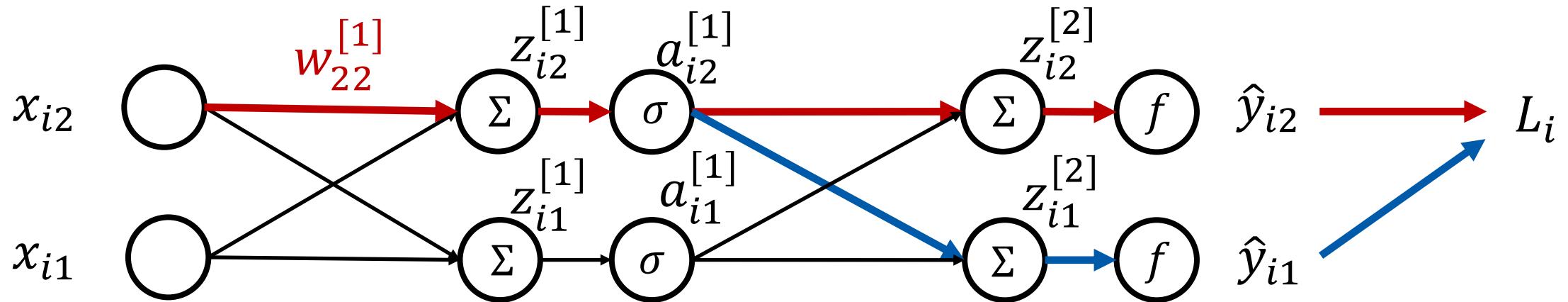
$$\hat{y}_i = f(\sigma^T(x_i^T W^{[1]}) W^{[2]})$$



$$\frac{\partial L_i}{\partial w_{22}^{[1]}} = ?$$

Обратное распространение ошибки

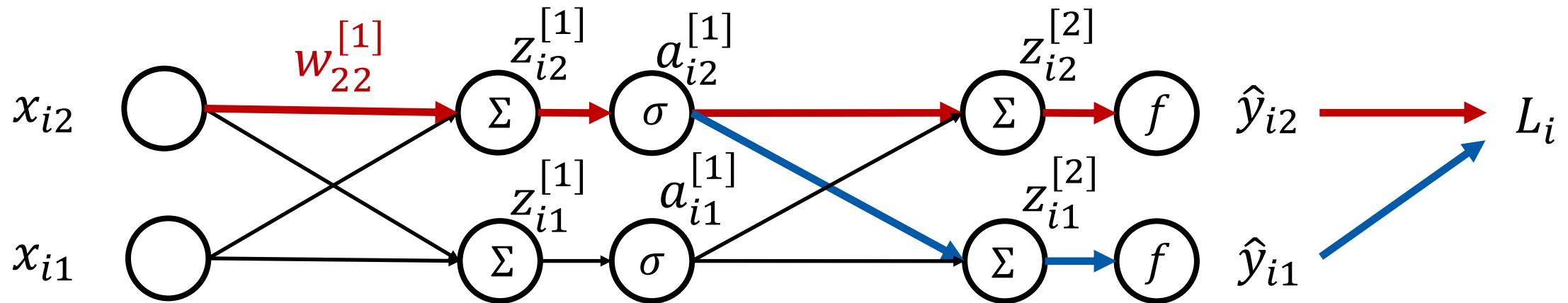
$$\hat{y}_i = f(\sigma^T(x_i^T W^{[1]}) W^{[2]})$$



$$\frac{\partial L_i}{\partial w_{22}^{[1]}} = \frac{\partial L_i}{\partial \hat{y}_{i1}} \frac{\partial \hat{y}_{i1}}{\partial z_{i1}^{[2]}} \frac{\partial z_{i1}^{[2]}}{\partial a_{i1}^{[1]}} \frac{\partial a_{i1}^{[1]}}{\partial w_{22}^{[1]}} + \frac{\partial L_i}{\partial \hat{y}_{i2}} \frac{\partial \hat{y}_{i2}}{\partial z_{i2}^{[2]}} \frac{\partial z_{i2}^{[2]}}{\partial a_{i2}^{[1]}} \frac{\partial a_{i2}^{[1]}}{\partial w_{22}^{[1]}}$$

Обратное распространение ошибки

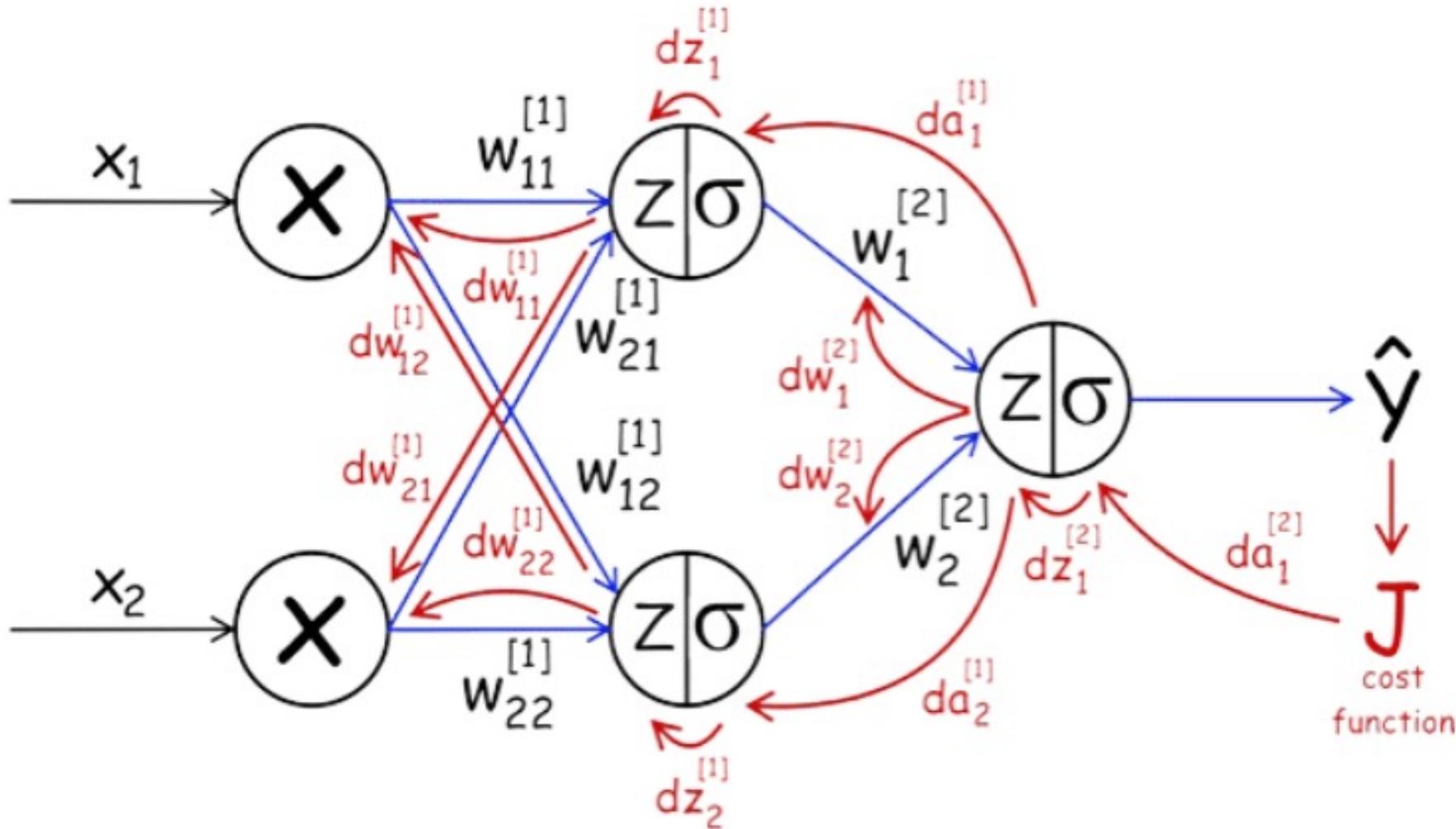
$$\hat{y}_i = f(\sigma^T(x_i^T W^{[1]}) W^{[2]})$$



$$\frac{\partial L_i}{\partial w_{22}^{[1]}} = \frac{\partial L_i}{\partial \hat{y}_{i1}} \frac{\partial \hat{y}_{i1}}{\partial z_{i1}^{[2]}} \frac{\partial z_{i1}^{[2]}}{\partial a_{i1}^{[1]}} \frac{\partial a_{i1}^{[1]}}{\partial w_{22}^{[1]}} + \frac{\partial L_i}{\partial \hat{y}_{i2}} \frac{\partial \hat{y}_{i2}}{\partial z_{i2}^{[2]}} \frac{\partial z_{i2}^{[2]}}{\partial a_{i2}^{[1]}} \frac{\partial a_{i2}^{[1]}}{\partial w_{22}^{[1]}}$$

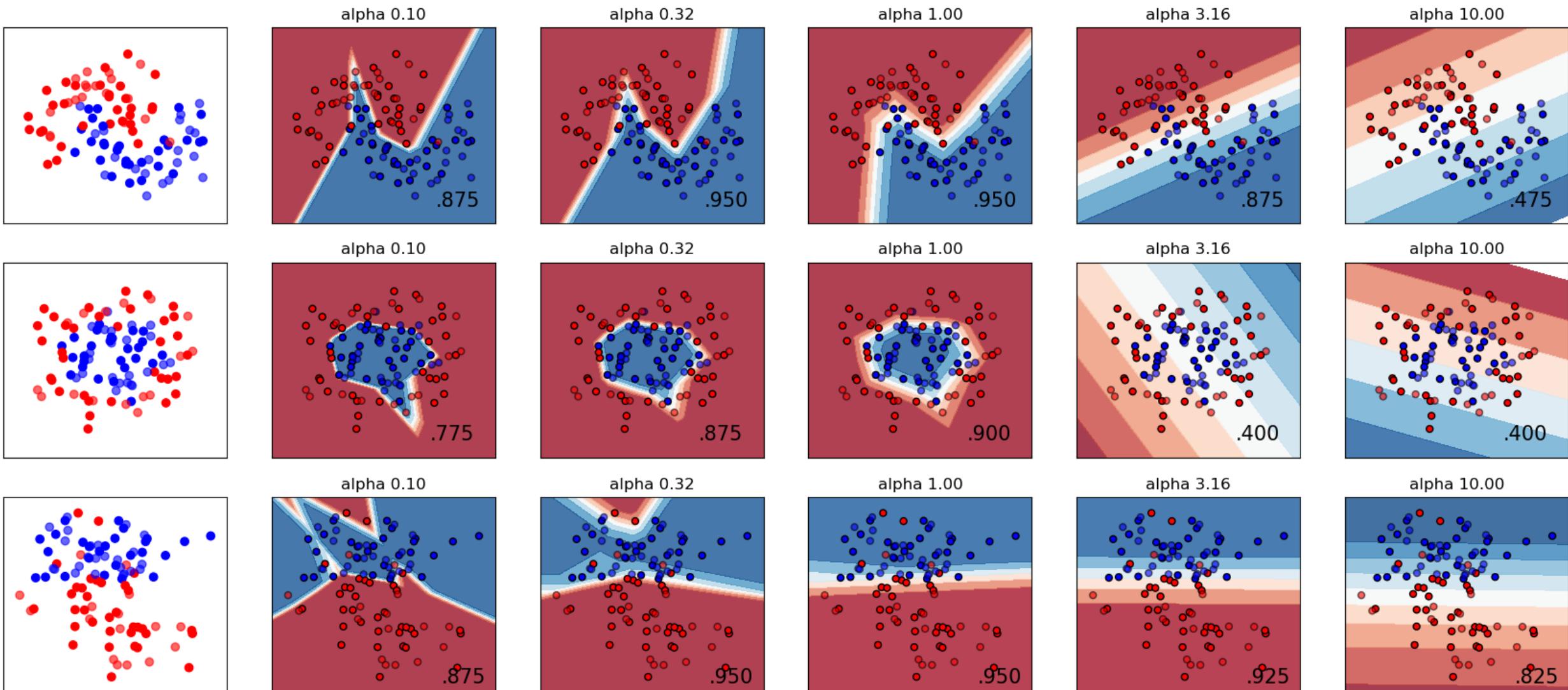
$$w_{22}^{[1]} = w_{22}^{[1]} - \alpha \frac{\partial L_i}{\partial w_{22}^{[1]}}$$

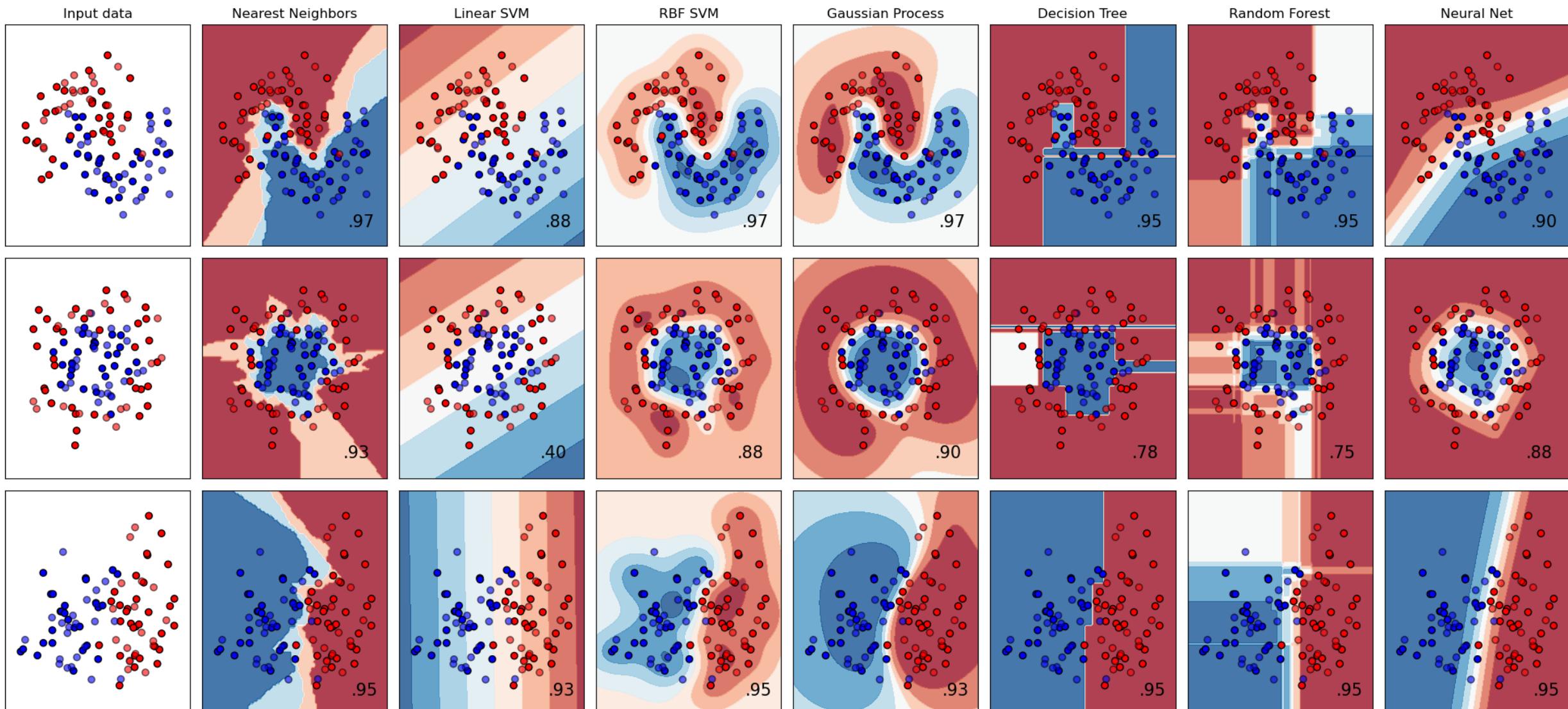
Обратное распространение ошибки



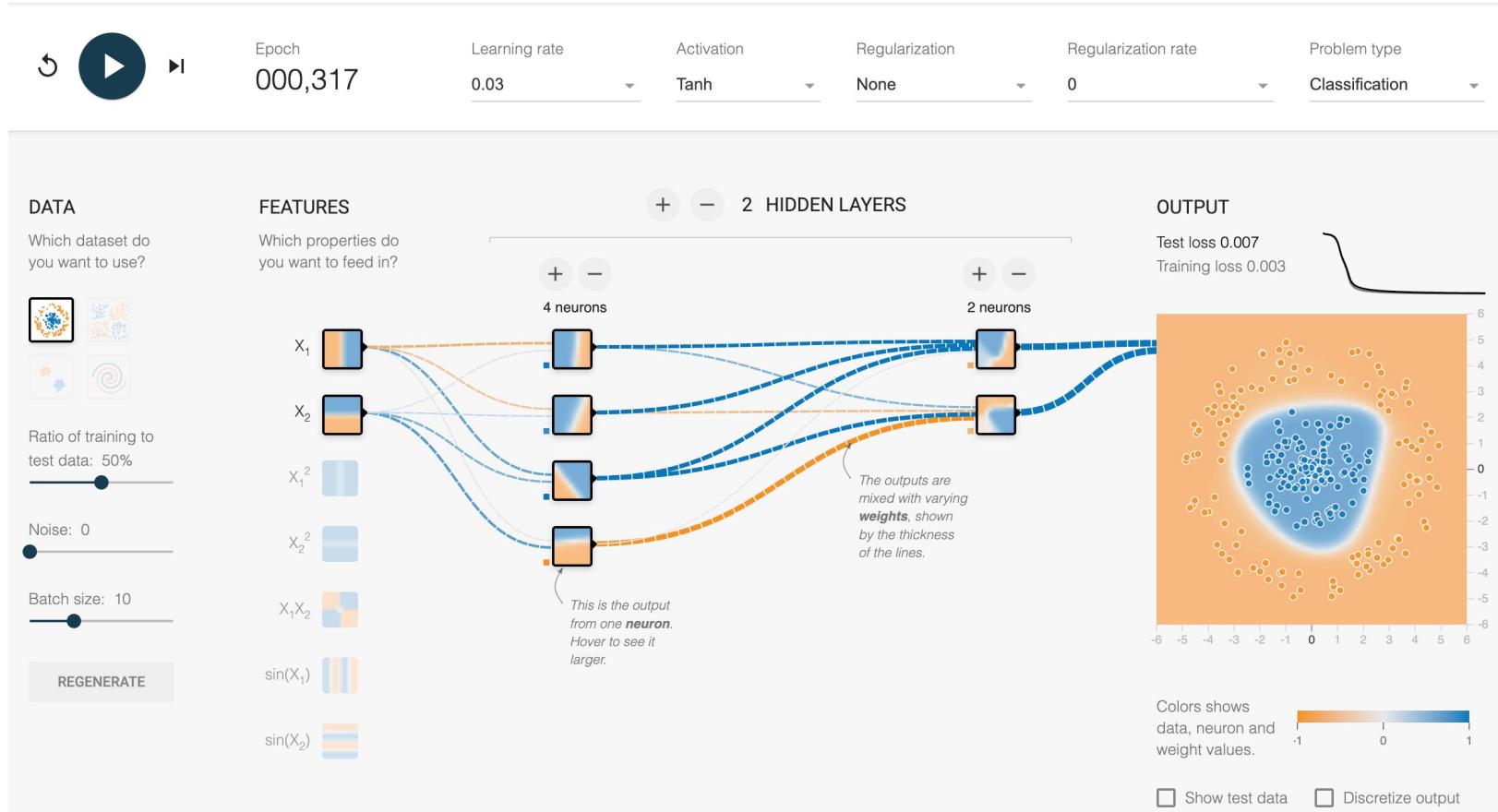
Примеры

Sklearn





Демонстрация



<https://playground.tensorflow.org>