Машинное обучение

Лекция 13

Нейронные сети

Михаил Гущин

mhushchyn@hse.ru

НИУ ВШЭ, 2025

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

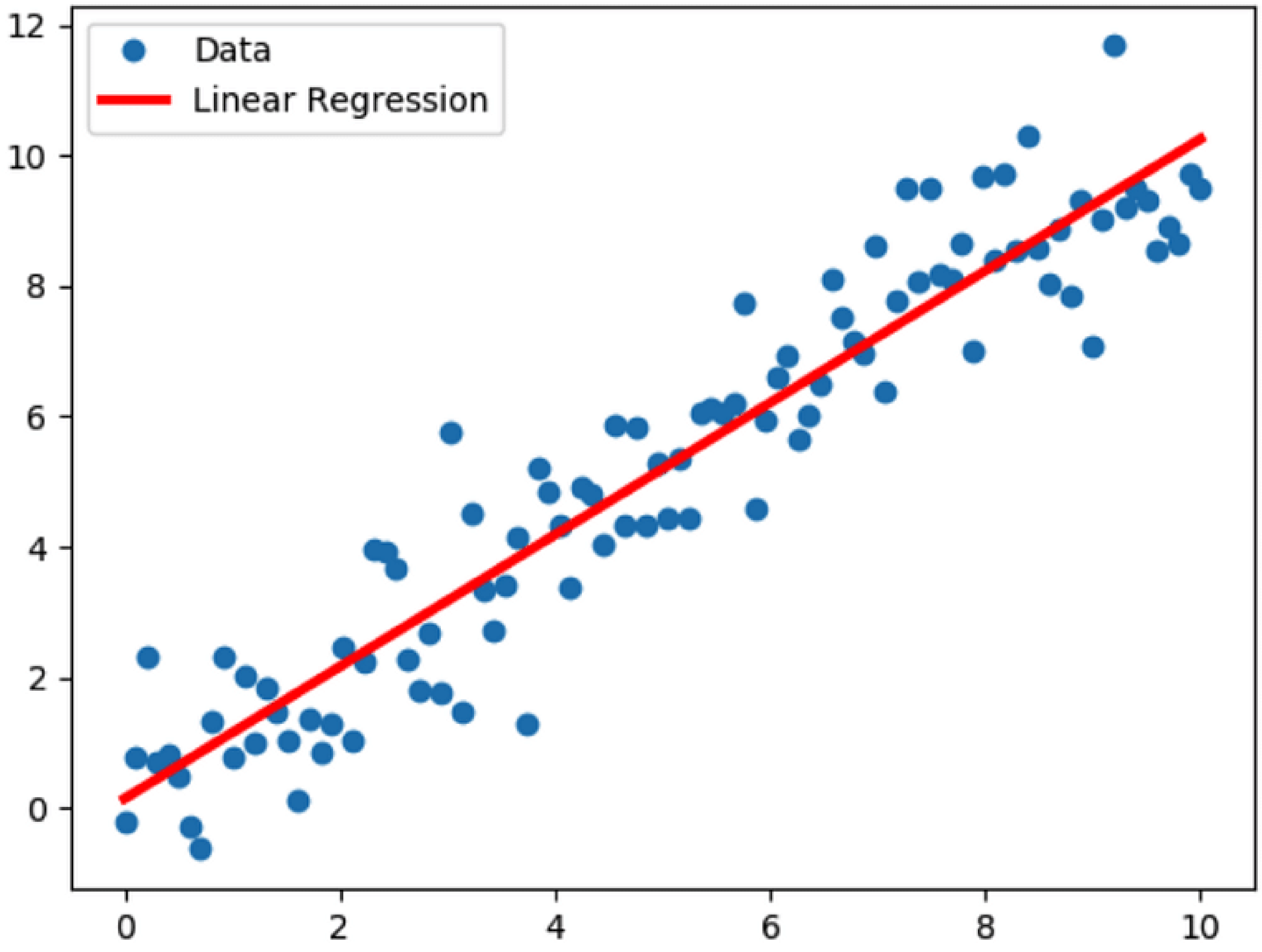
Эволюция нейронных сетей

Линейная регрессия

Логистическая регрессия

Нейронная сеть

Линейная регрессия


Data
Linear Regression
12
10
8
6
4
2
0
0
2
4
6
8
10

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:

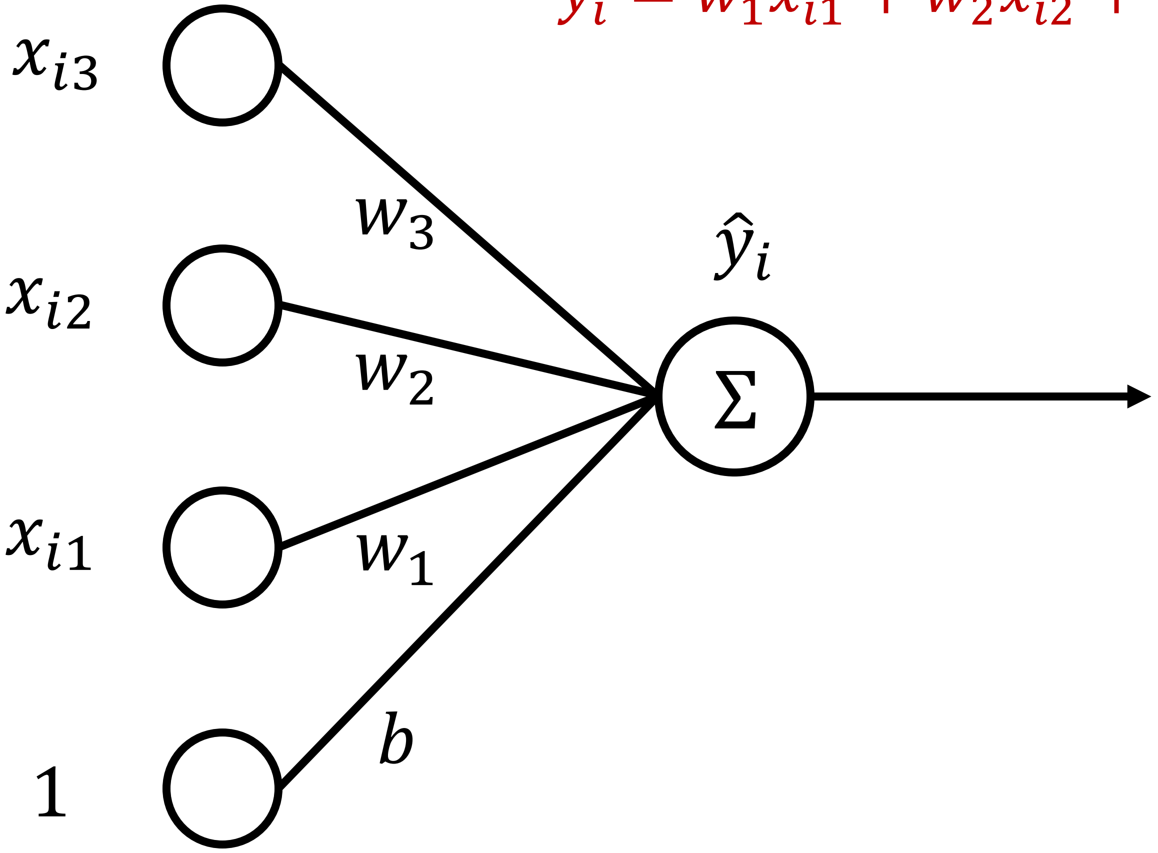
$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + b = x_i^T w + b$$

- Функция потерь:

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

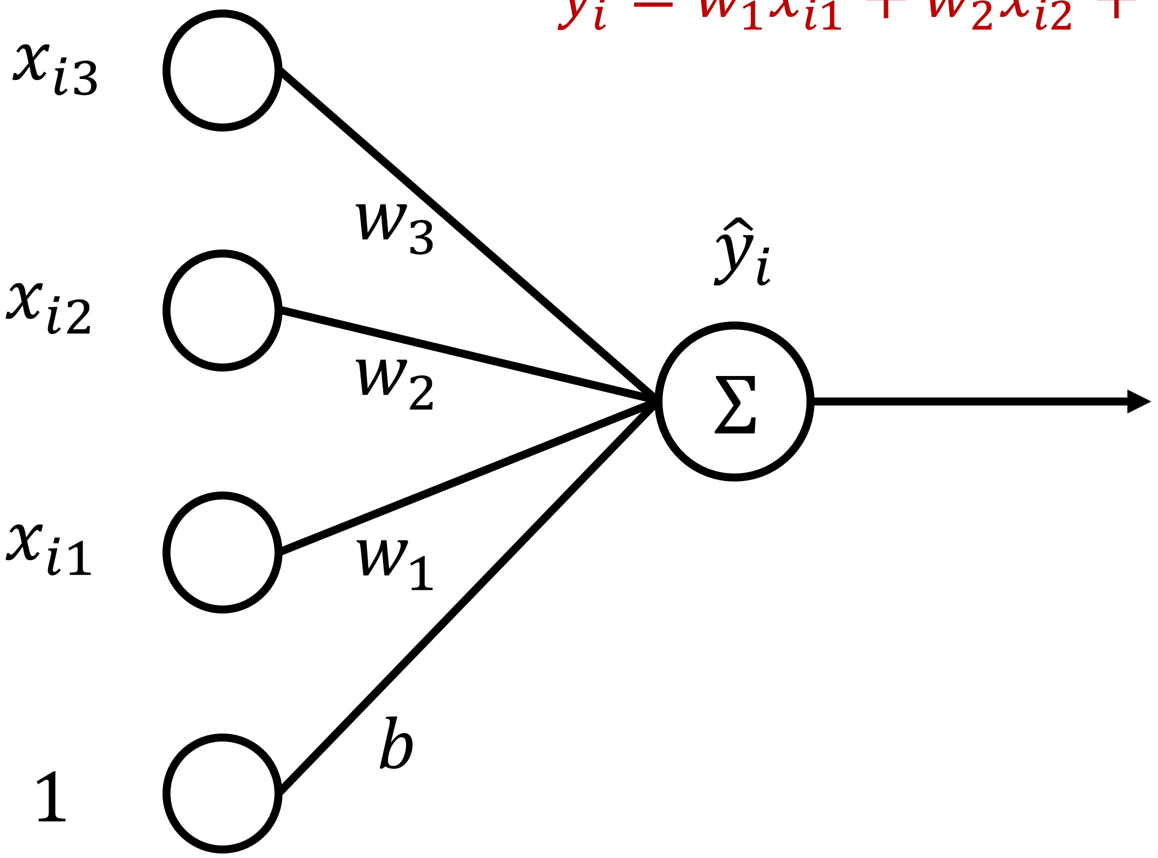
Линейная регрессия как граф вычислений

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



Линейная регрессия как граф вычислений

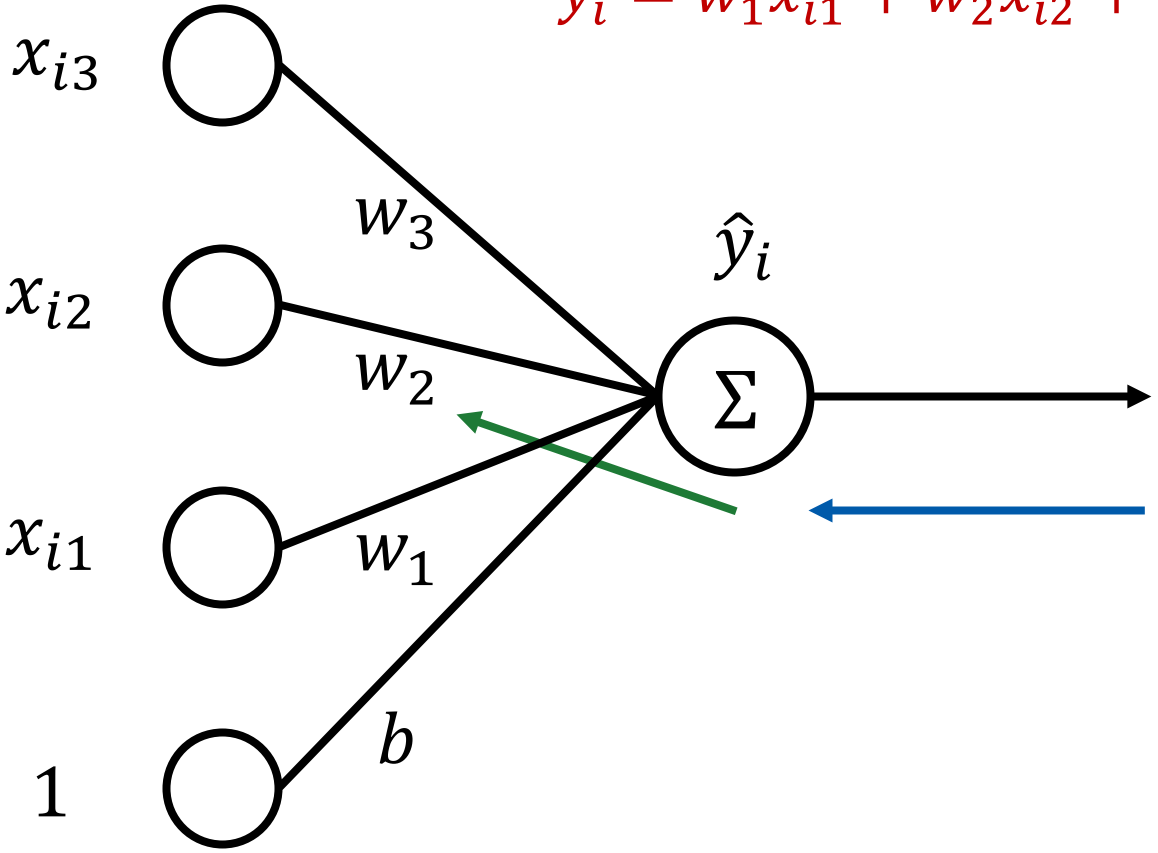
$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + b = x_i^T w + b$$



$$L_i = \frac{1}{N}(\hat{y}_i - y_i)^2$$

Градиент функции потерь

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

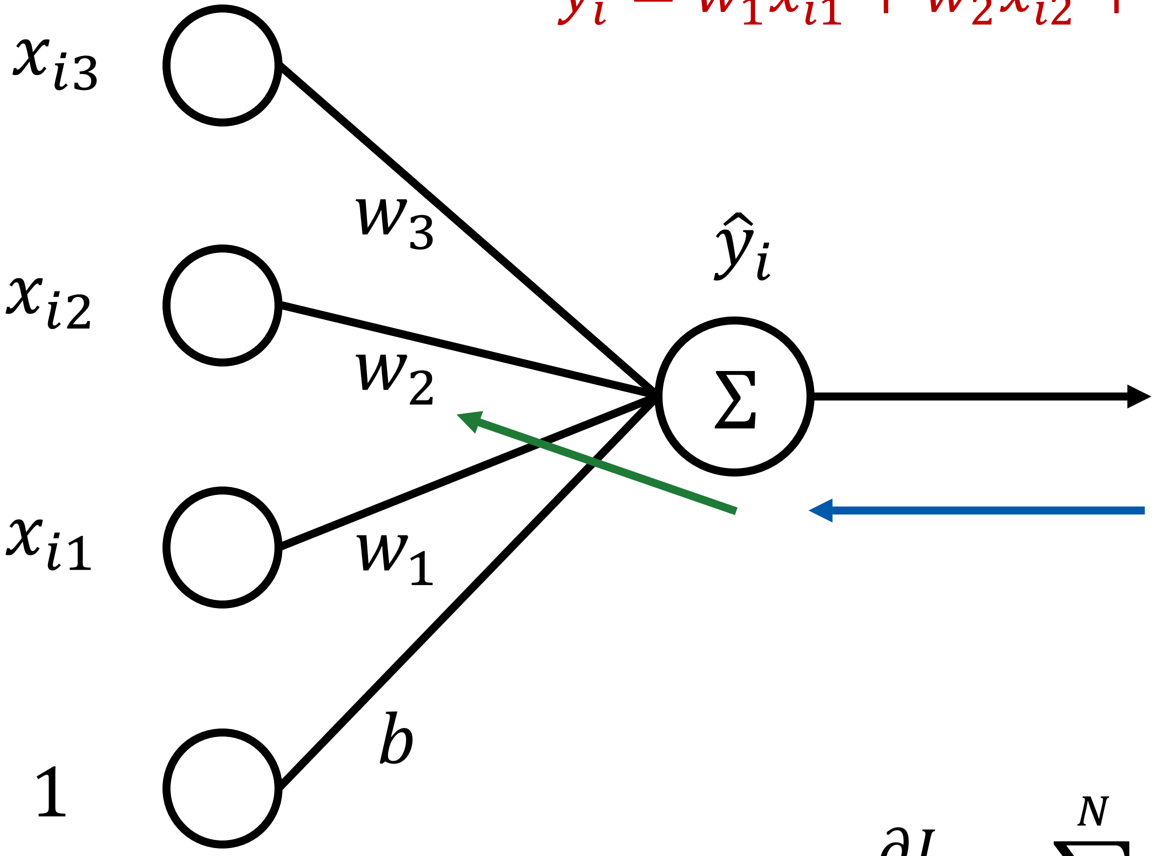


$$L_i = \frac{1}{N}(\hat{y}_i - y_i)^2$$

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

Градиентный спуск

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



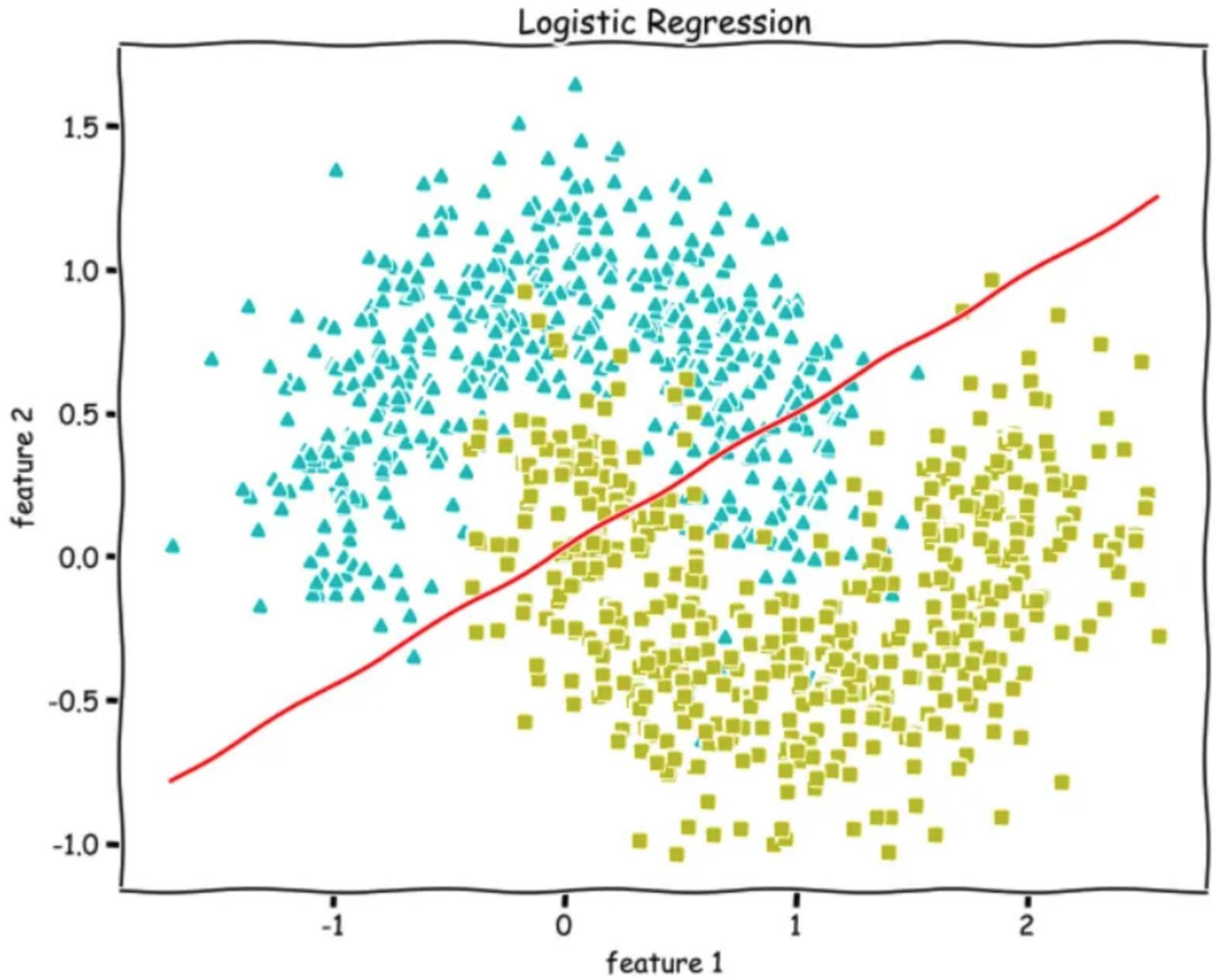
$$L_i = \frac{1}{N}(\hat{y}_i - y_i)^2$$

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

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

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

Логистическая регрессия


Logistic Regression
1.5
1.0
0.5
0.0
-0.5
-1.0
feature 2
-1
0
1
2
feature 1

Логистическая регрессия

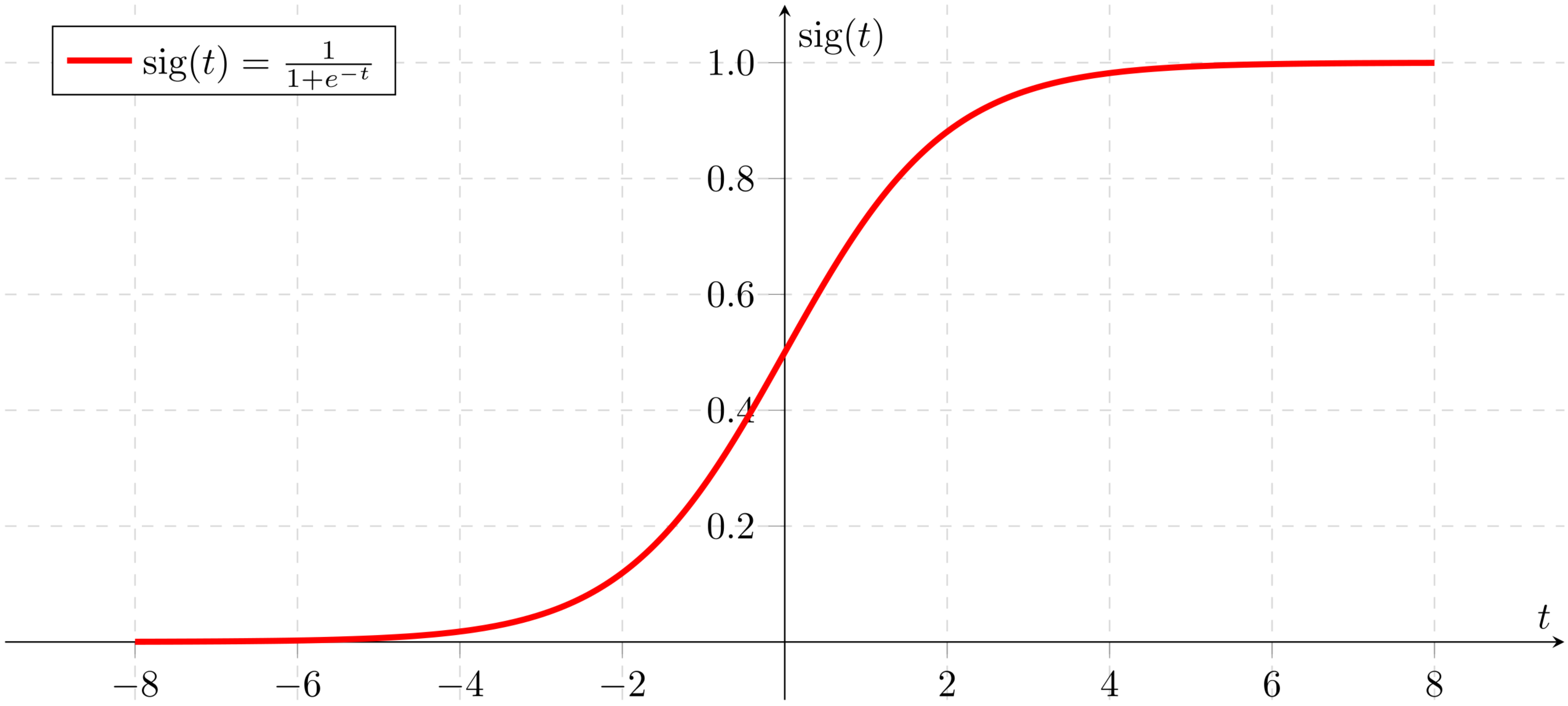
- Пусть дан набор наблюдений $\{x_i, y_i\}_{i=1}^N$, где $x_i \in R^3$, $y_i \in \{0, 1\}$
- Модель логистической регрессии:

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

- Функция потерь:

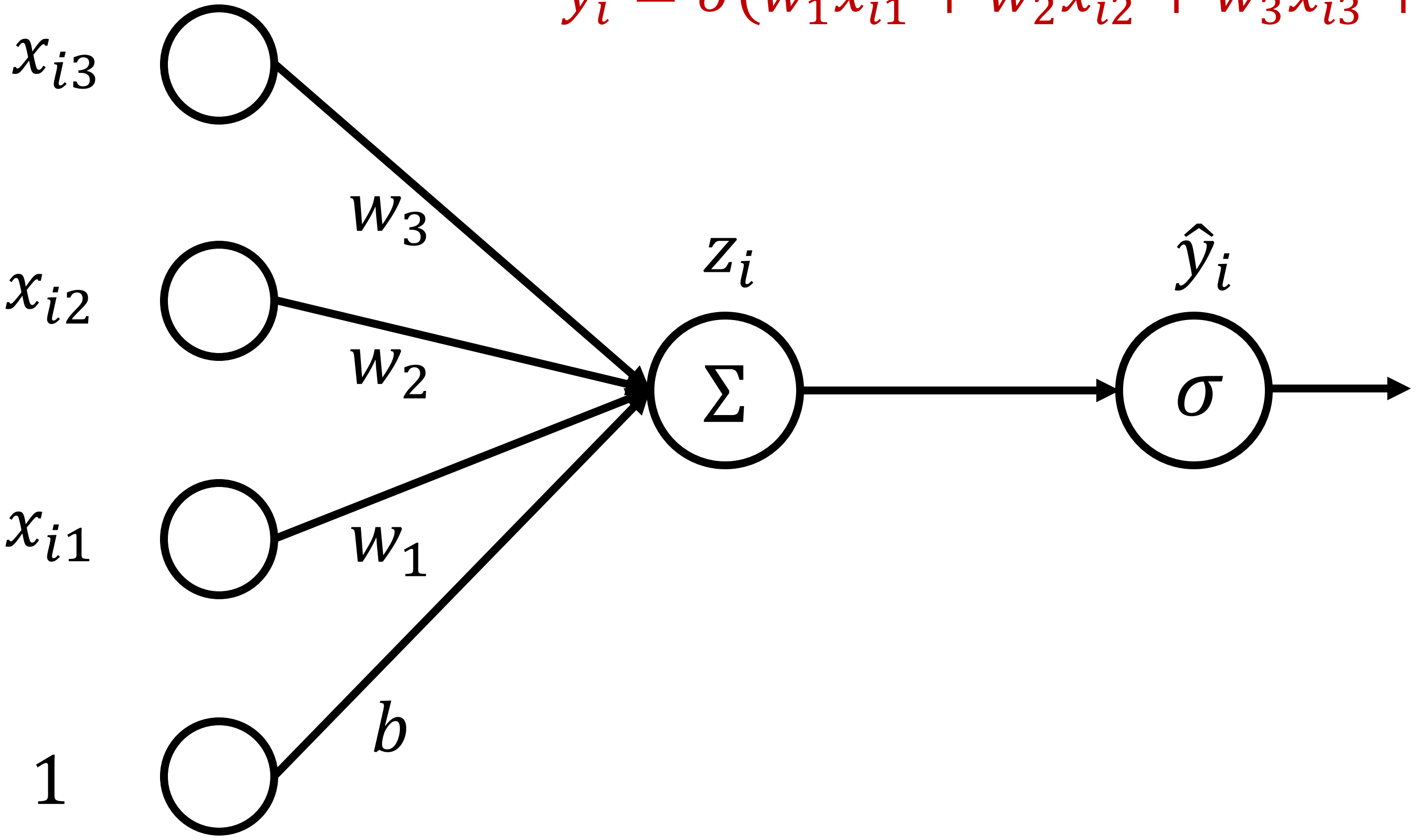
$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i) \rightarrow \min_{b, w_1, w_2, w_3}$$

Сигмоида



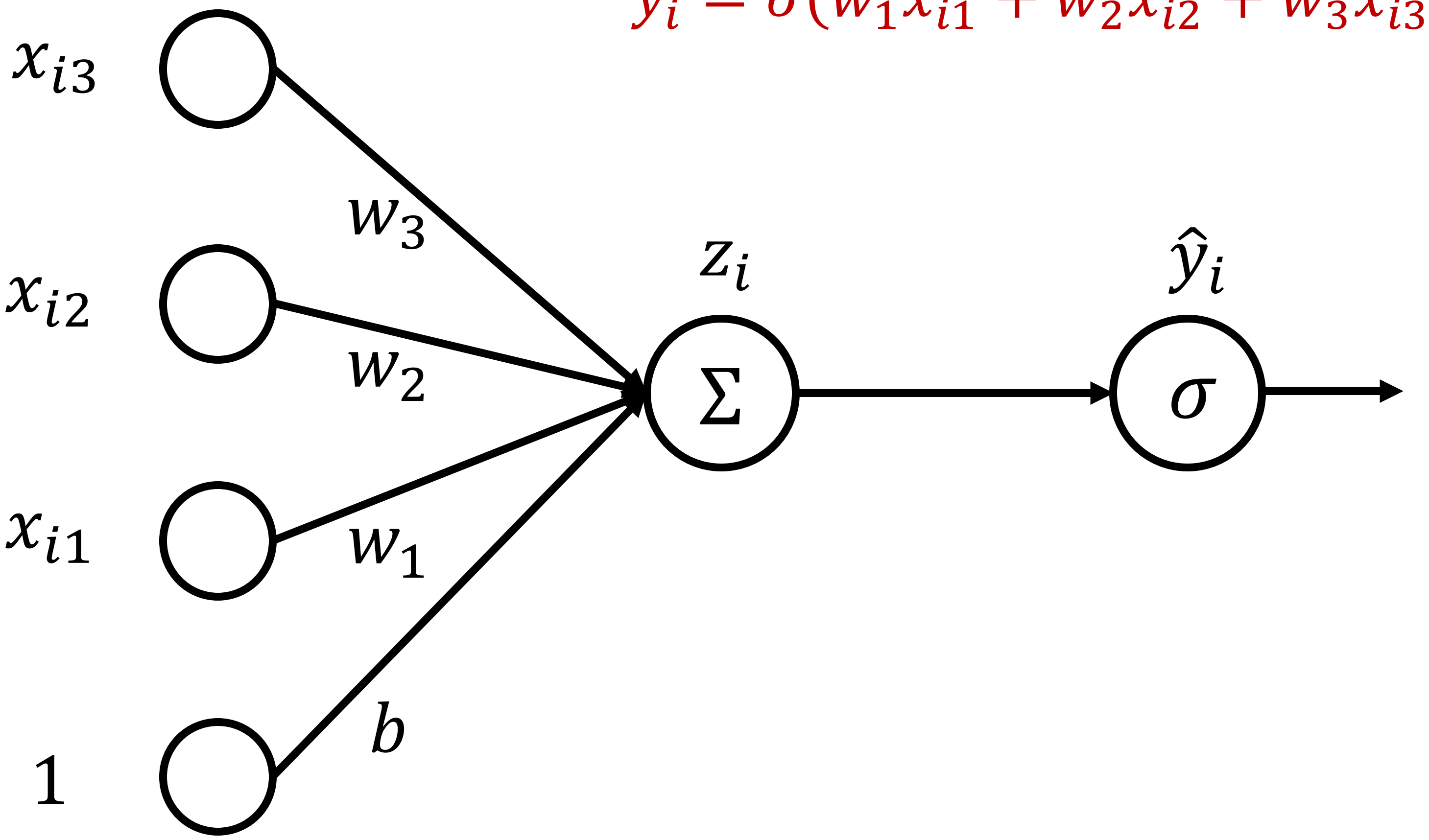
Логистическая регрессия как граф вычислений

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



Логистическая регрессия как граф вычислений

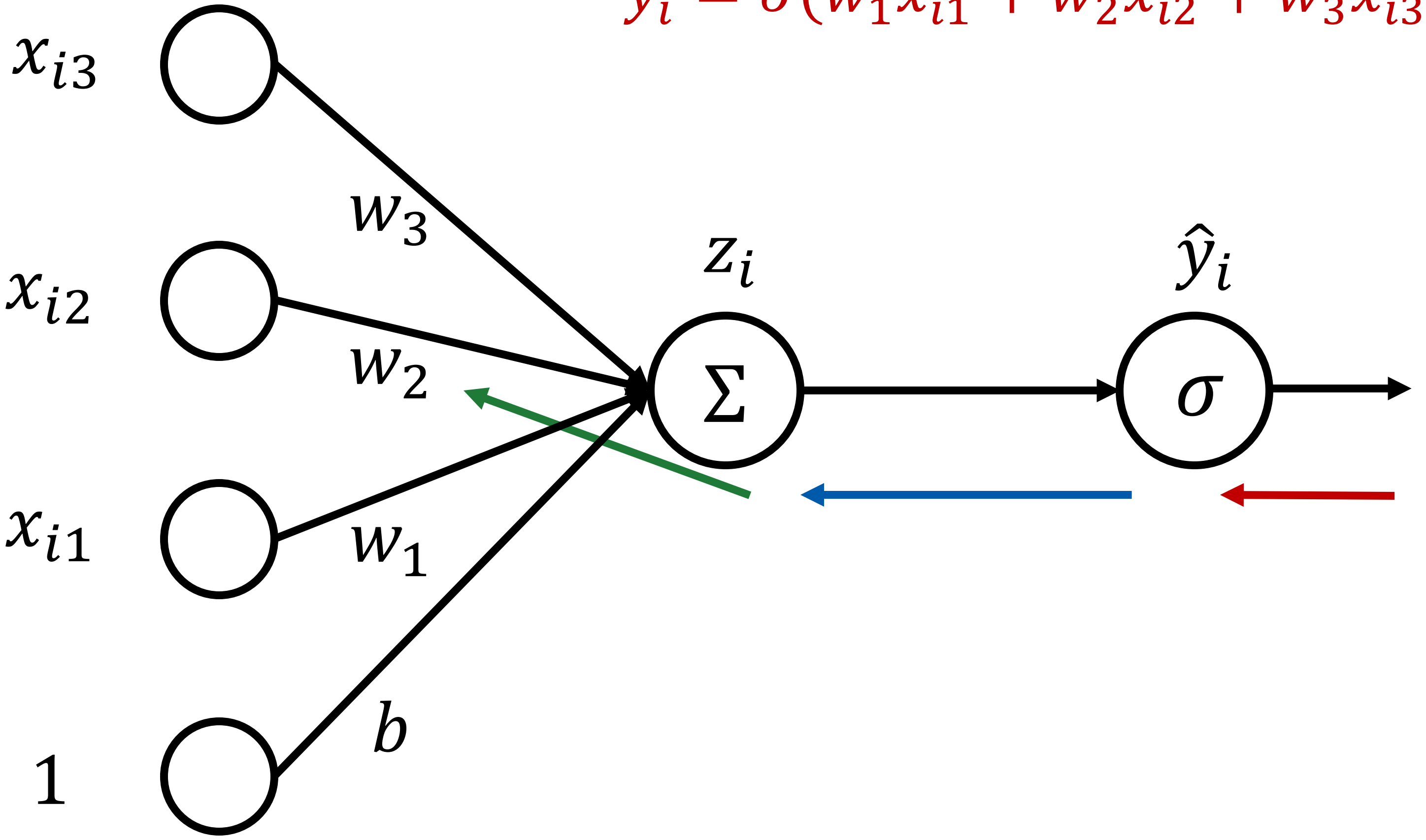
$$\hat{y}_i = \sigma(w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + b) = \sigma(x_i^T w + b)$$



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

Градиент функции потерь

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

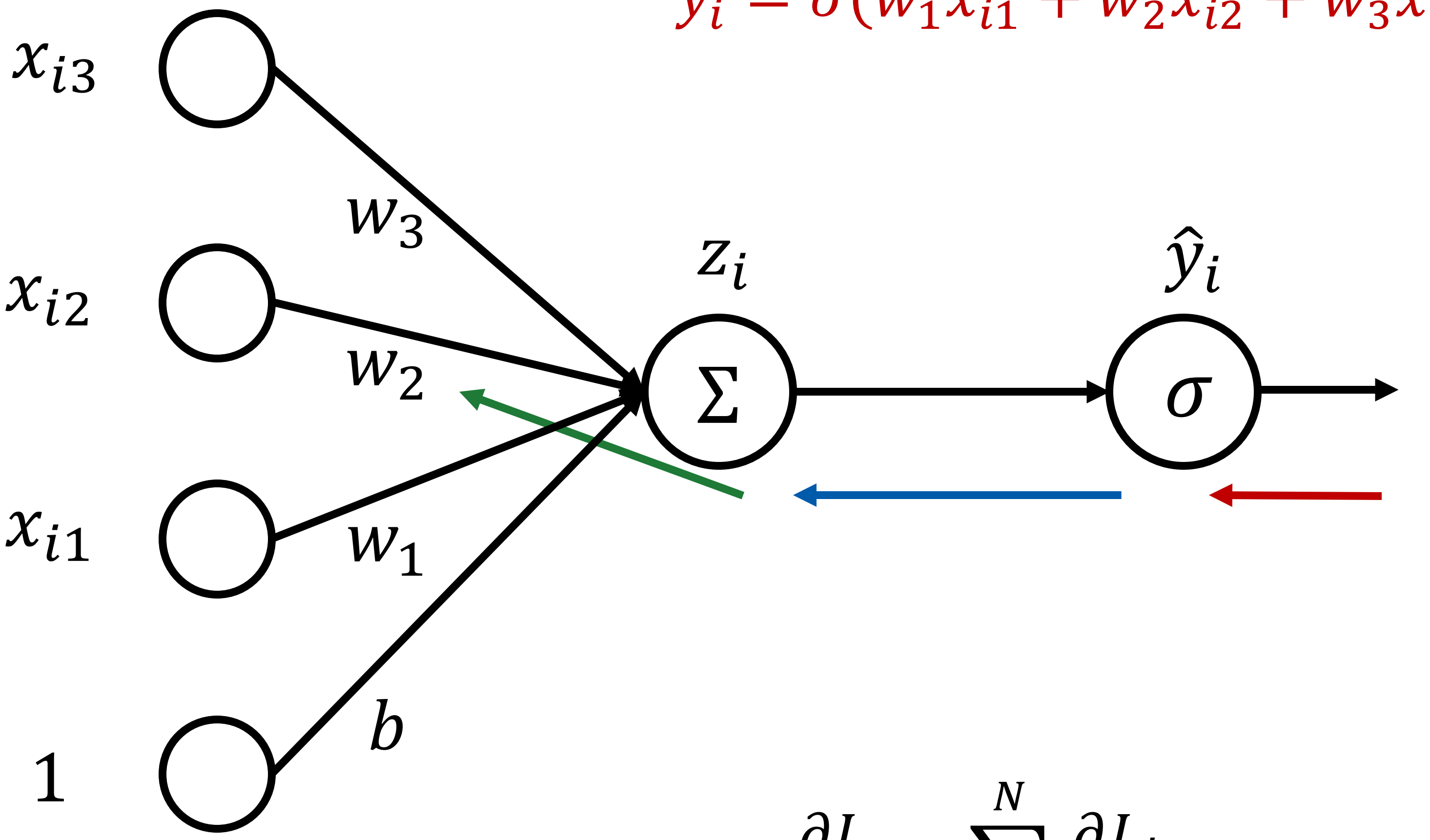


$$L_i = -y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i)$$

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

Градиентный спуск

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



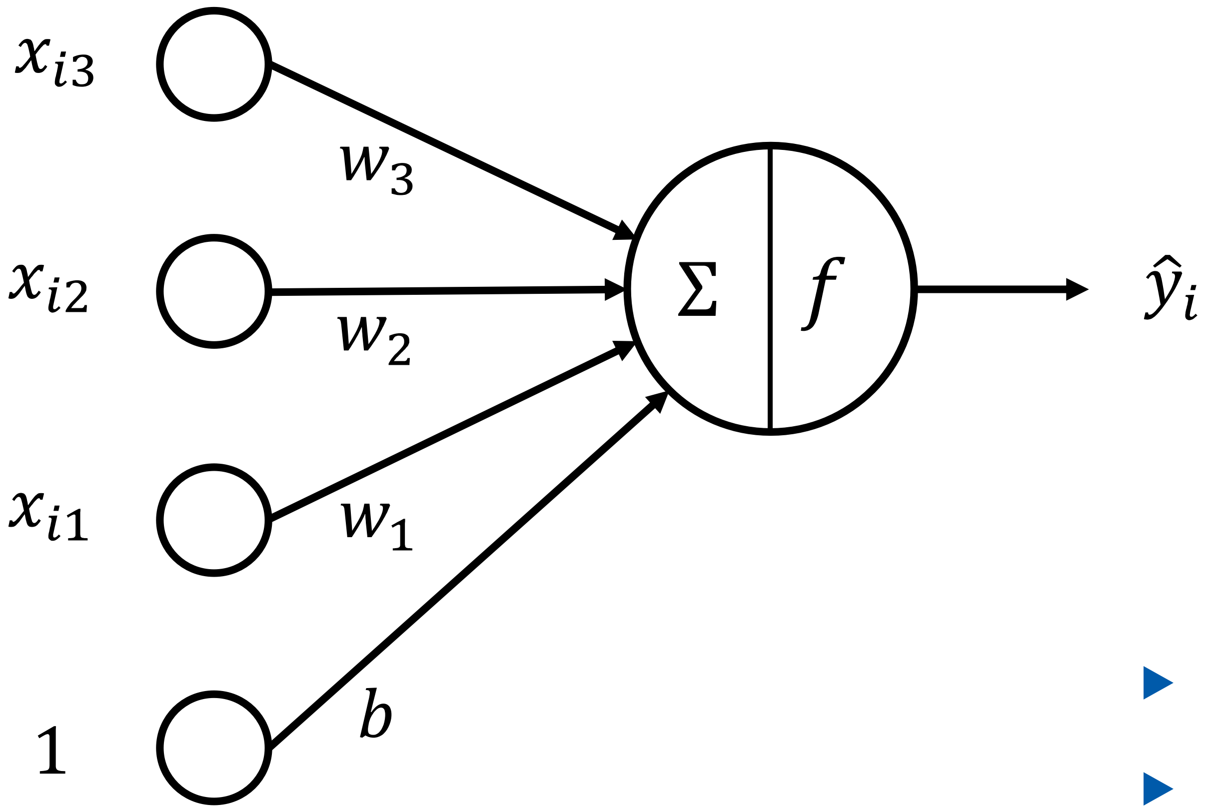
$$L_i = -y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i)$$

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

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

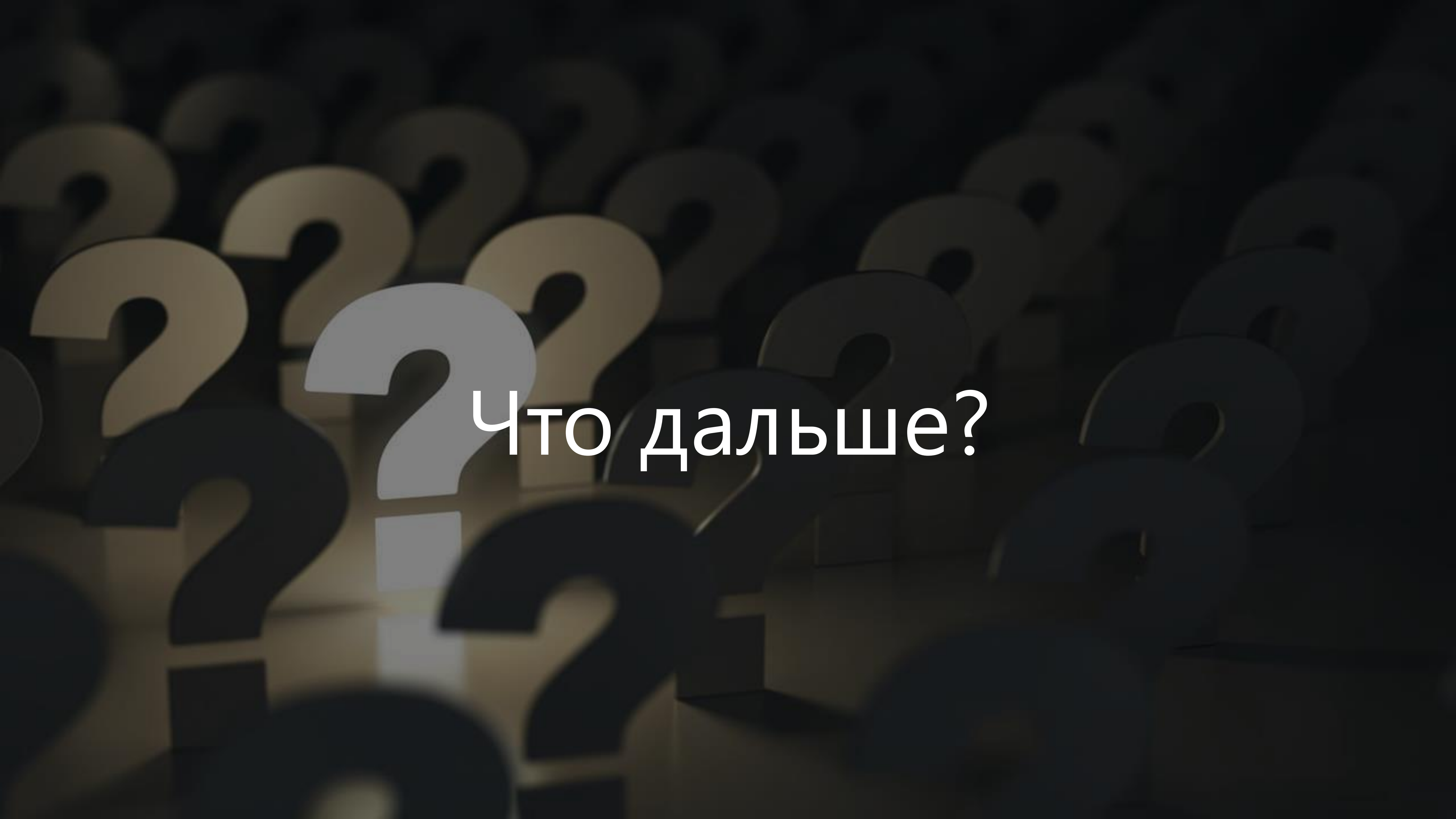
$$w_2^{(t+1)} = w_2^{(t)} - \alpha \frac{\partial L}{\partial w_2}$$

Линейная модель в общем виде



$$\hat{y}_i = f(x_i^T w + b)$$

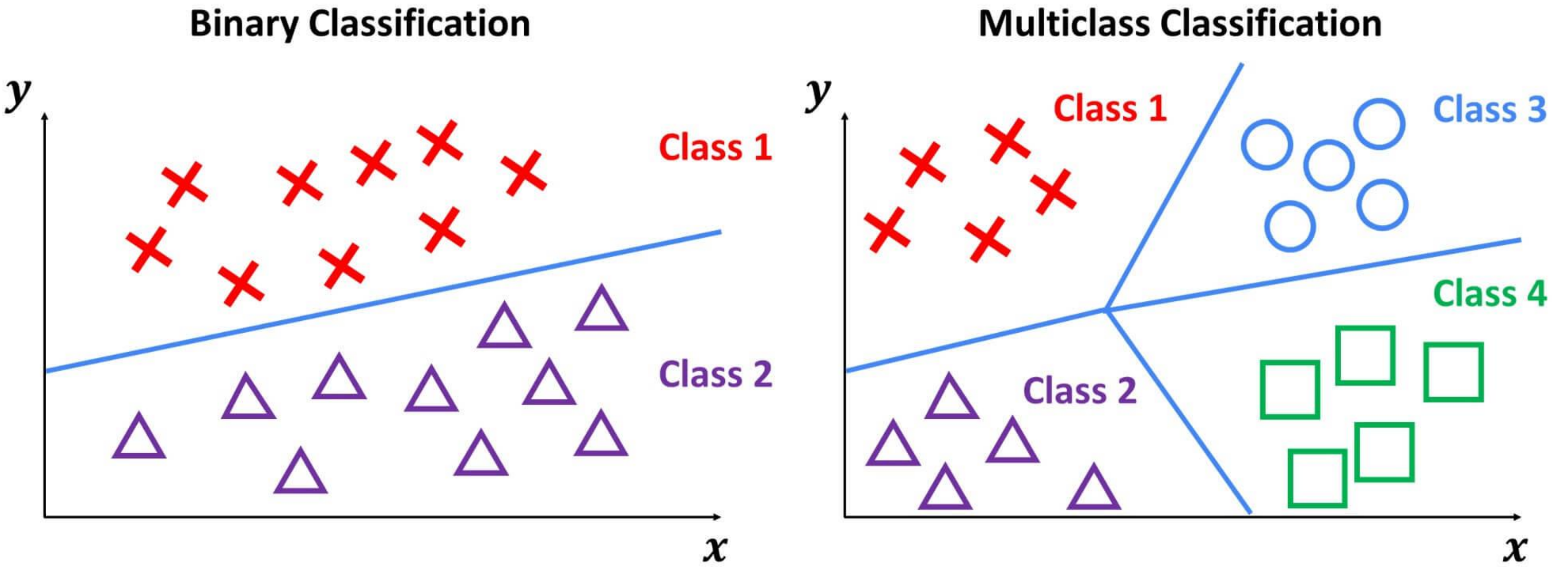
- $f(z) = z$ – линейная регрессия
- $f(z) = \sigma(z)$ – логистическая регрессия



Что дальше?

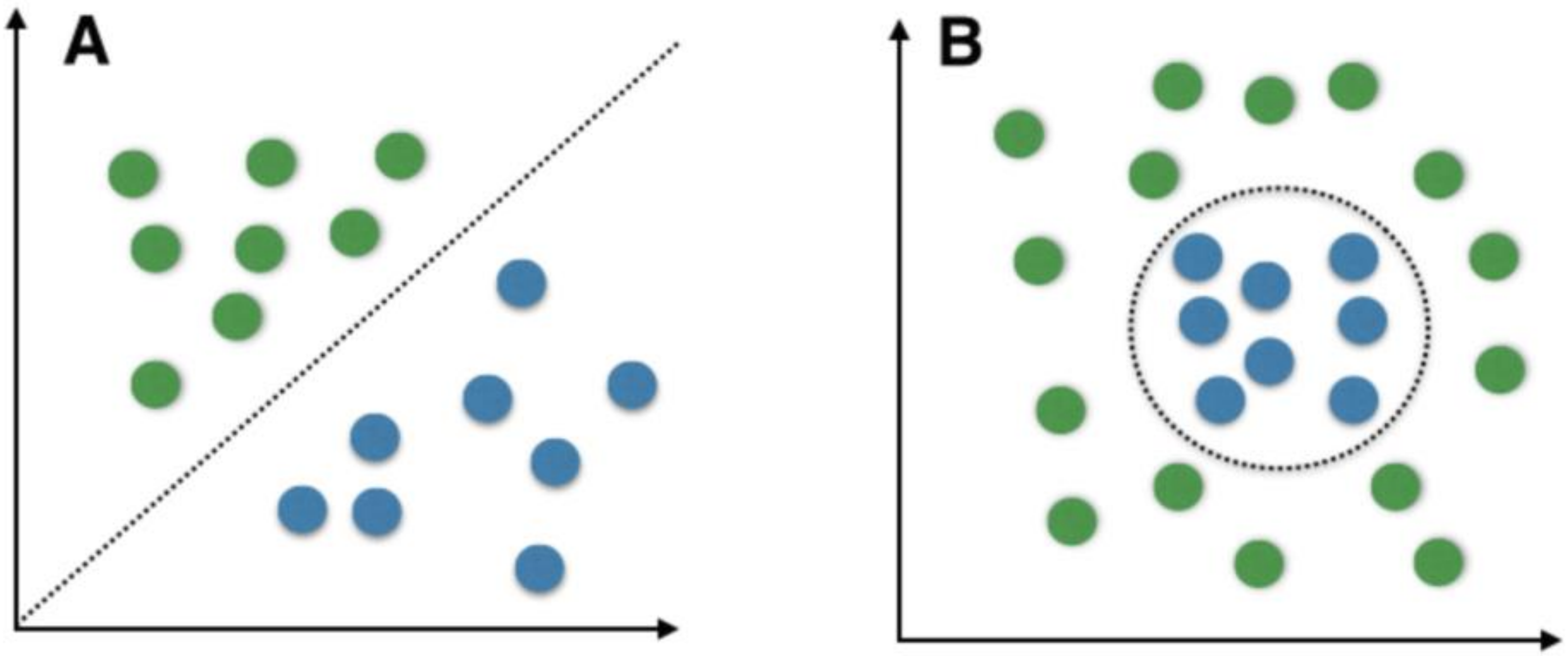
А что дальше?

- Что делать, если у меня больше, чем два класса в данных?



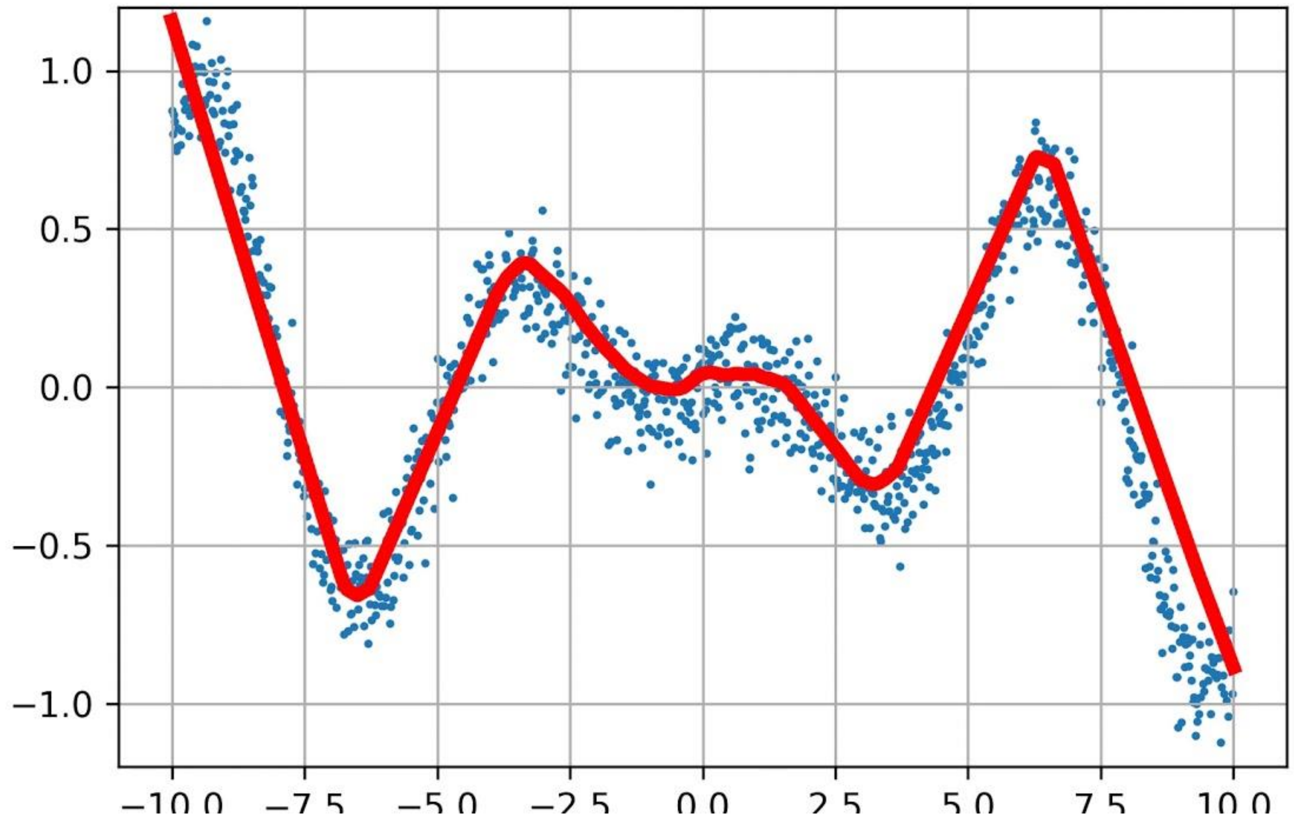
А что дальше?

- Что делать, если классы не разделяются линейной функцией?



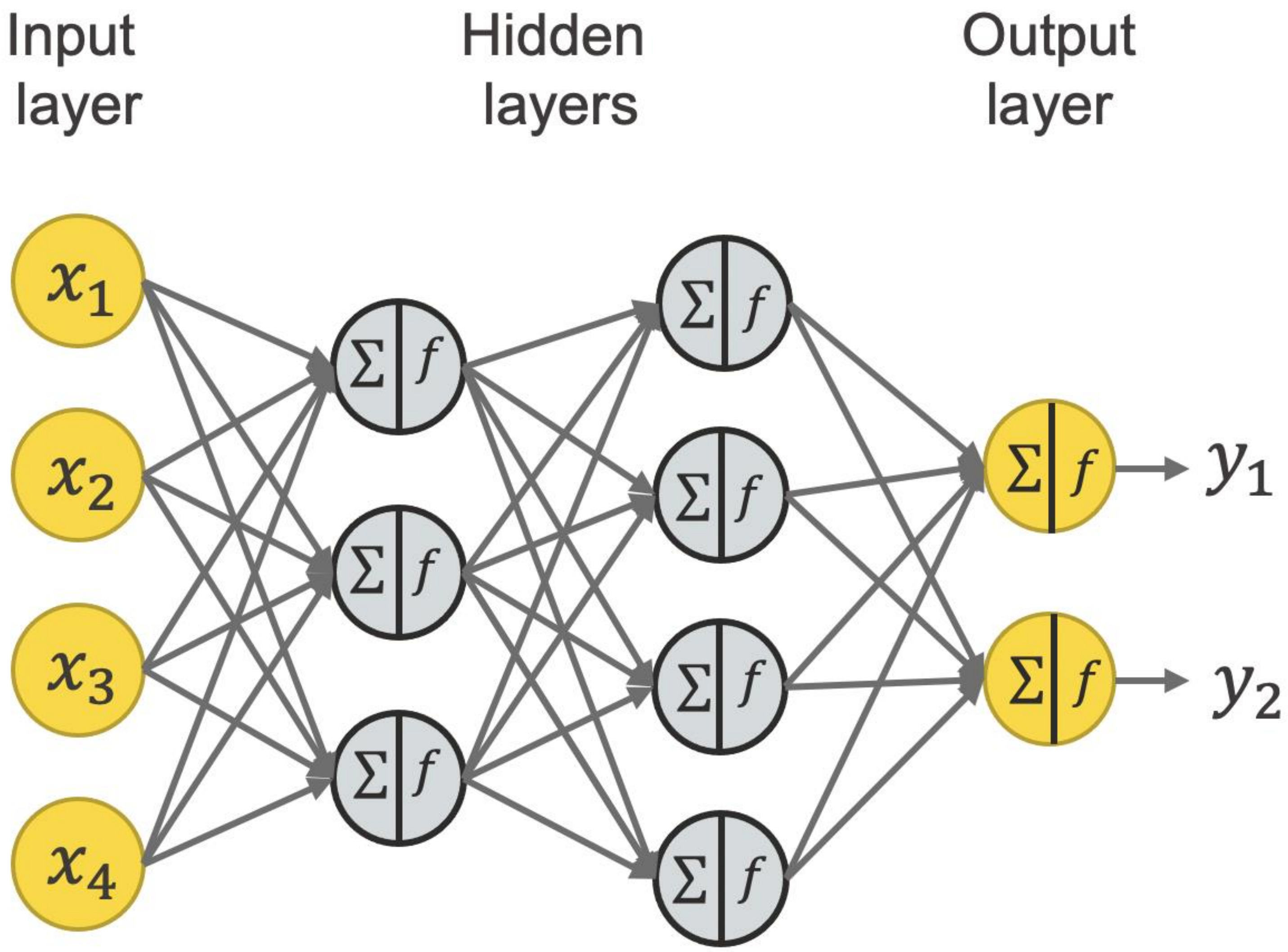
А что дальше?

- Что делать, если у меня нелинейная регрессия?

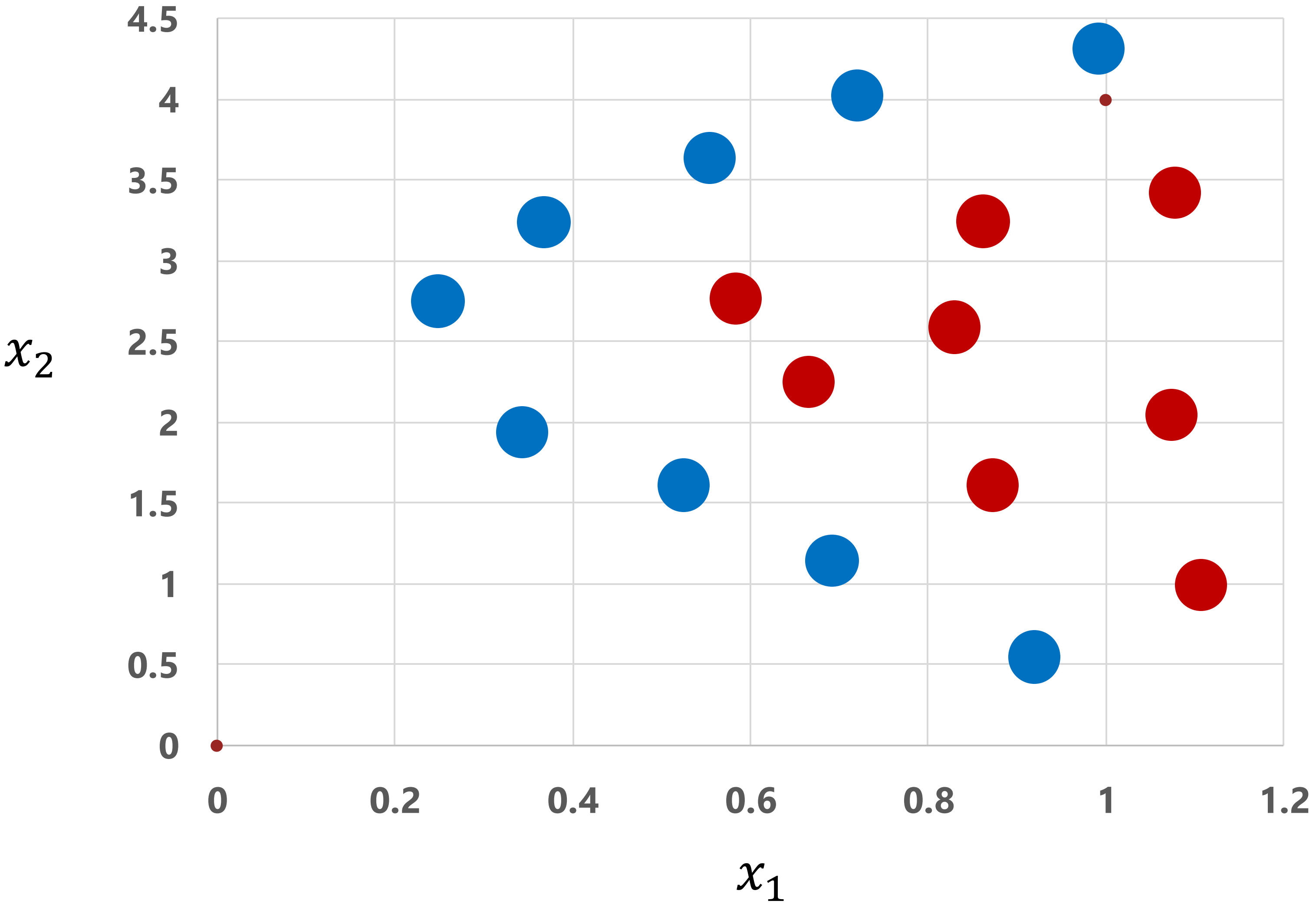


Нейронные сети

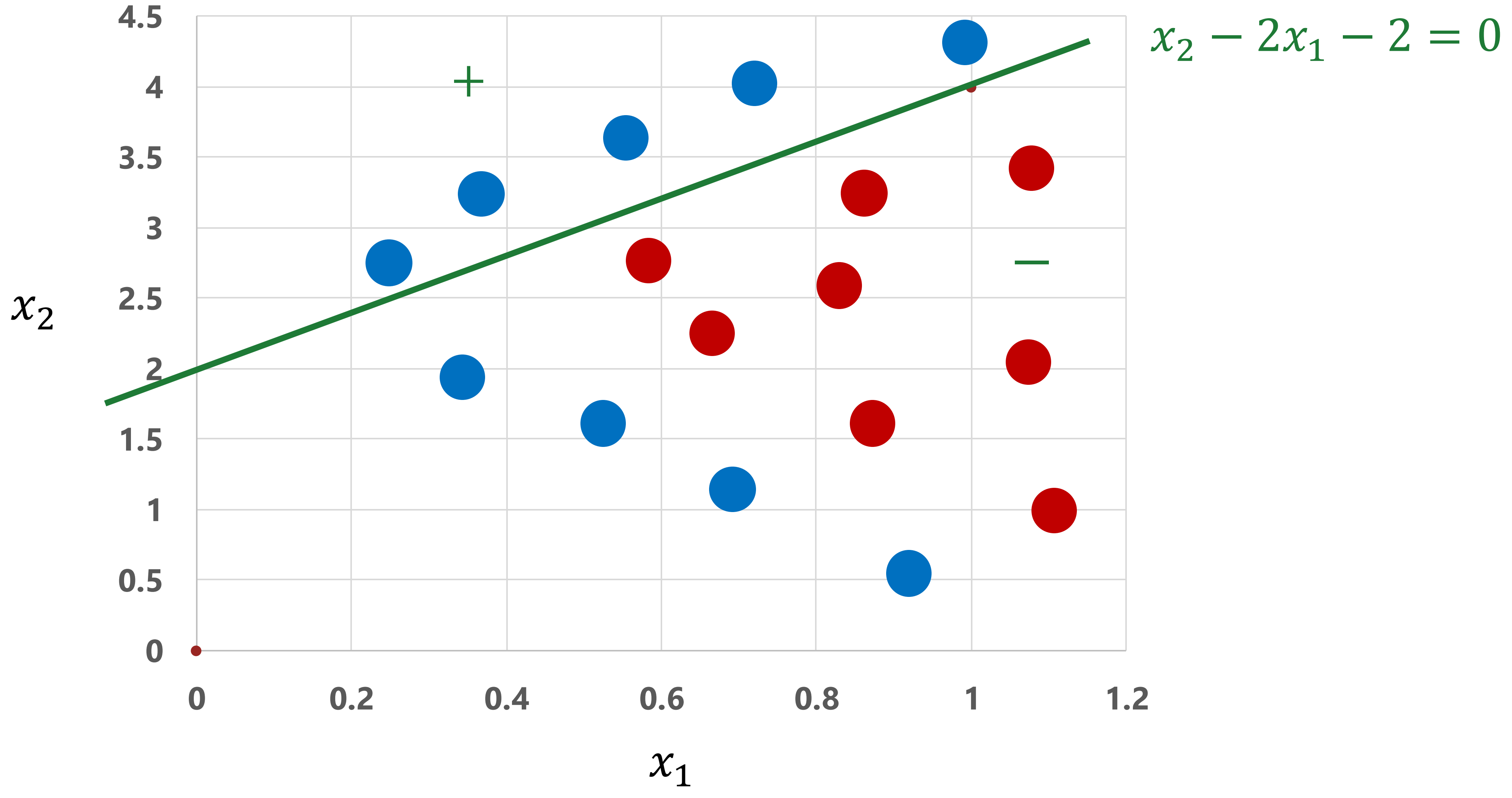
Нейронная сеть



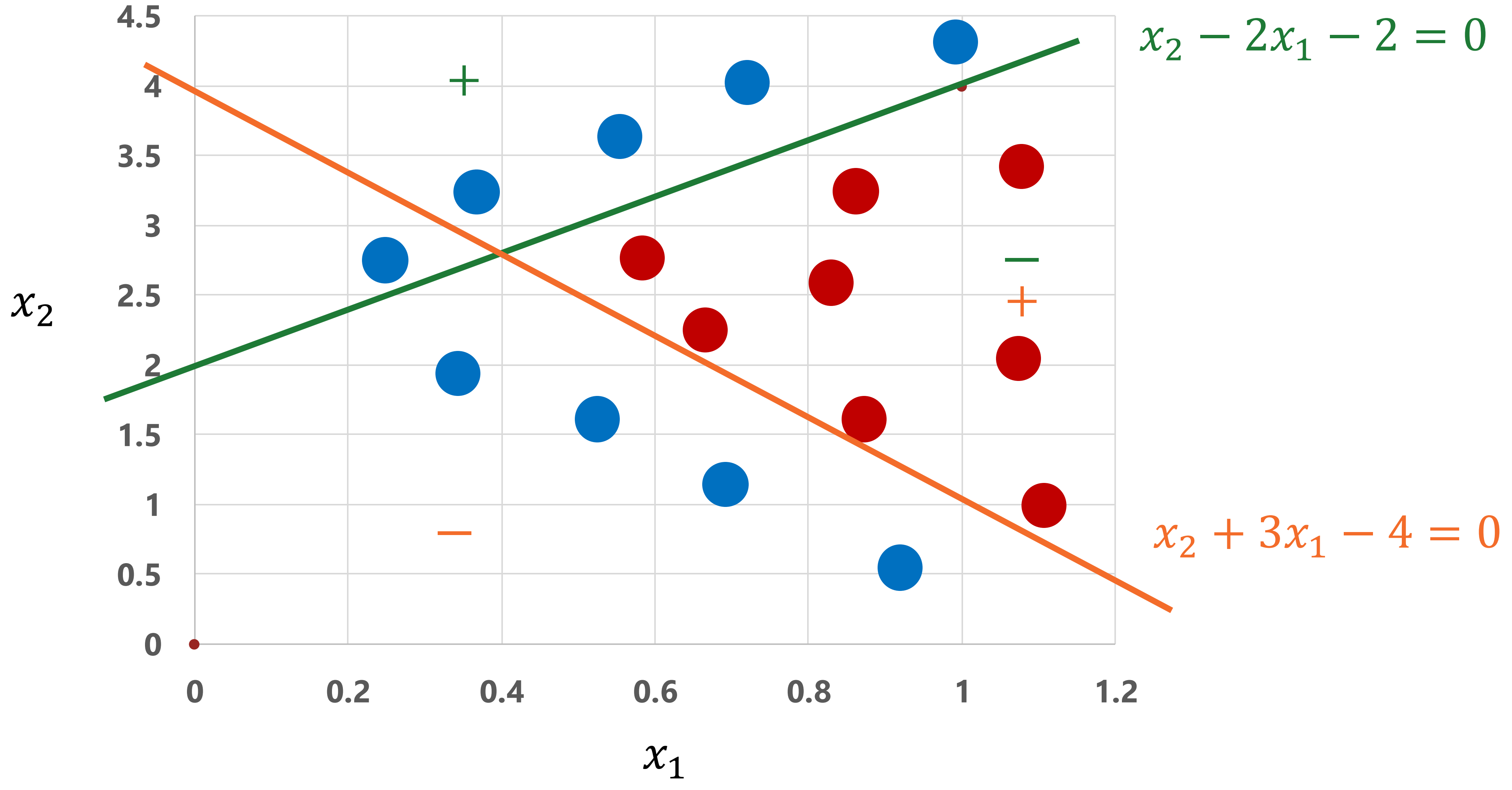
Пример



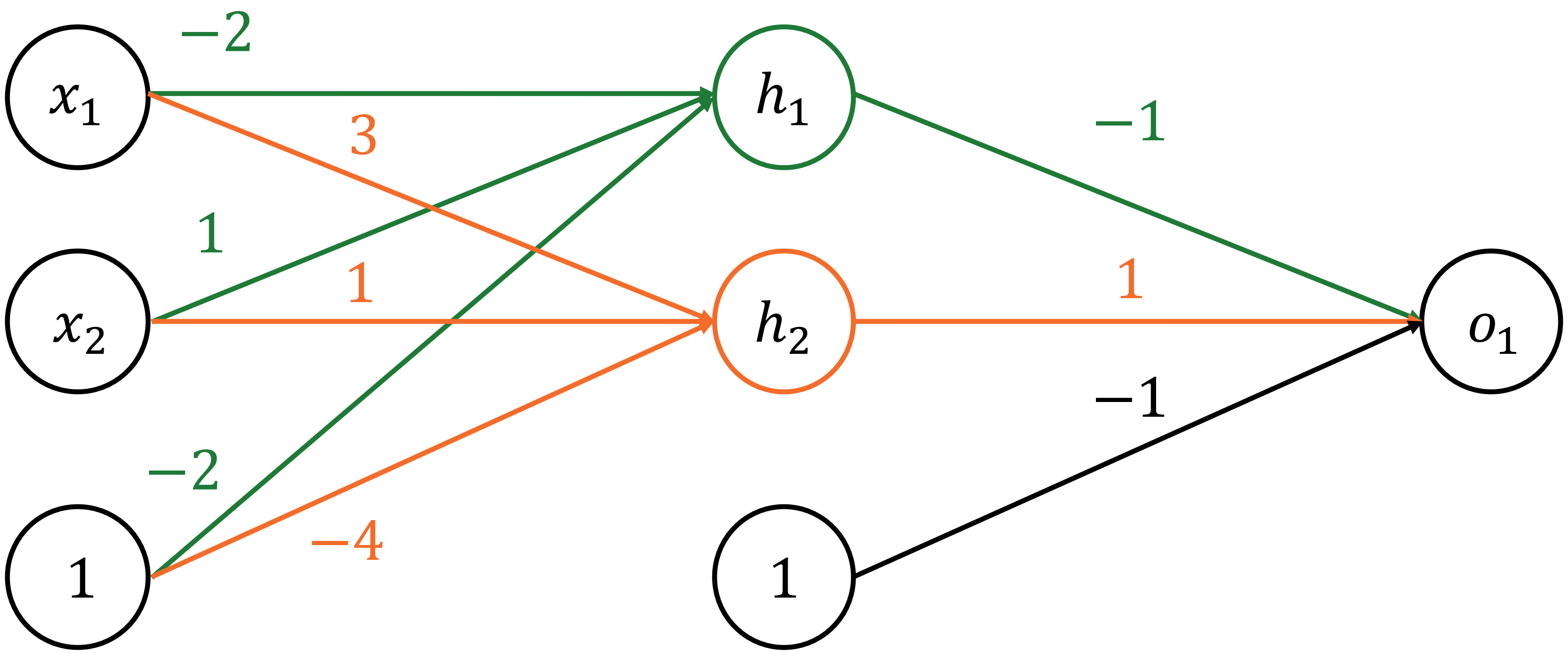
Пример



Пример



Нейронная сеть для примера



$$h_1 = sign(x_2 - 2x_1 - 2)$$

$$o_1 = sign(h_2 - h_1 - 1)$$

$$h_2 = sign(x_2 + 3x_1 - 4)$$

Векторная запись нейронной сети

$$h = f_1(x^T W^{(1)} + b^{(1)})$$
$$o = f_2(h^T W^{(2)} + b^{(2)})$$

$$W^{(1)} = \begin{pmatrix} -2 & 3 \\ 1 & 1 \end{pmatrix},\ b^{(1)} = \begin{pmatrix} -2 \\ -4 \end{pmatrix},\ f_1(z) = sign(z)$$

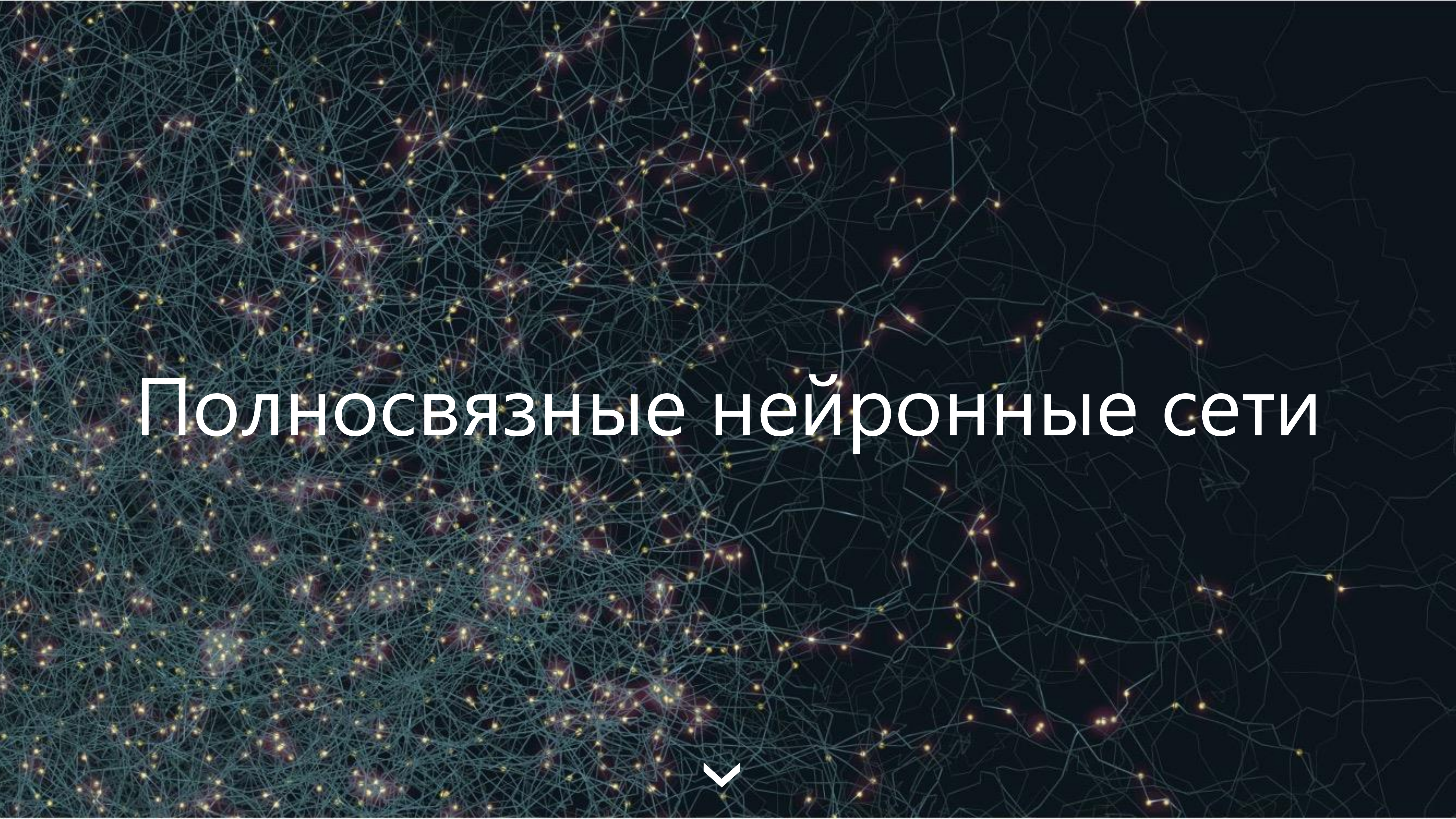
$$W^{(2)} = \begin{pmatrix} -1 \\ 1 \end{pmatrix},\qquad b^{(2)} = (-1),\ f_1(z) = sign(z)$$

Матричная запись нейронной сети

$$H = f_1(XW^{(1)} + b^{(1)})$$
$$O = f_2(HW^{(2)} + b^{(2)})$$

$$X = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{n1} & x_{n2} \end{pmatrix}, W^{(1)} = \begin{pmatrix} -2 & 3 \\ 1 & 1 \end{pmatrix}, b^{(1)} = (-2 \quad -4), f_1(z) = sign(z)$$

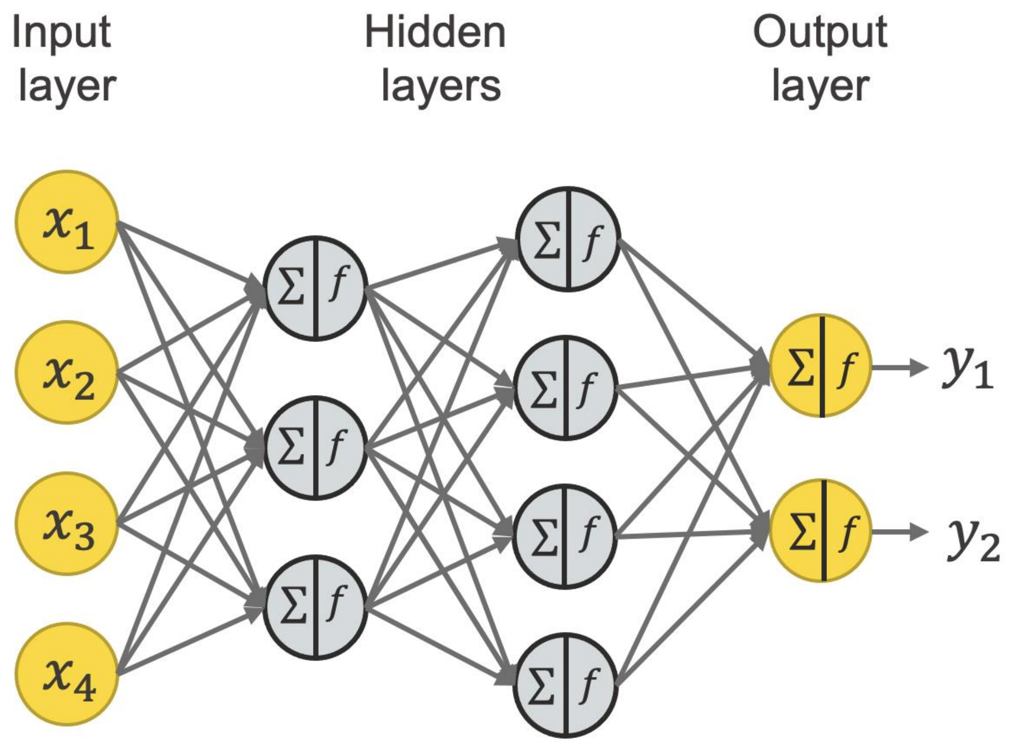
$$H = \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{n1} & h_{n2} \end{pmatrix}, W^{(2)} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}, b^{(2)} = (-1), \quad f_1(z) = sign(z)$$



Полносвязные нейронные сети

Нейронная сеть

- Пусть дан набор наблюдений $\{x_i, y_i\}_{i=1}^{n}$, где $x_i \in R^d$, $y_i \in R^q$
- Построим нейронную сеть



Нейронная сеть в матричной форме

- Матрица наблюдений $X \in R^{n \times d}$ из n объектов, каждый из которых имеет d входных признаков:

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}$$

- Число строк – число наблюдений (объектов)
- Число столбцов – число входных признаков

Нейронная сеть в матричной форме

$$H^{(1)} = f_1\left(XW^{(1)} + b^{(1)}\right)$$
$$H^{(2)} = f_2(H^{(1)}W^{(2)} + b^{(2)})$$
$$O = f_3(H^{(2)}W^{(3)} + b^{(3)})$$

Размеры матриц:

- $X \in R^{n\times d}$, $W^{(1)} \in R^{d\times h}$, $b^{(1)} \in R^{1\times h}$, h - число нейронов в первом слое
- $H^{(1)} \in R^{n\times h}$, $W^{(2)} \in R^{h\times m}$, $b^{(2)} \in R^{1\times m}$, m - число нейронов во втором слое
- $H^{(2)} \in R^{n\times m}$, $W^{(3)} \in R^{m\times q}$, $b^{(3)} \in R^{1\times q}$, q - число выходов сети
- $O \in R^{n\times q}$ - матрица прогнозов сети

Полносвязный слой

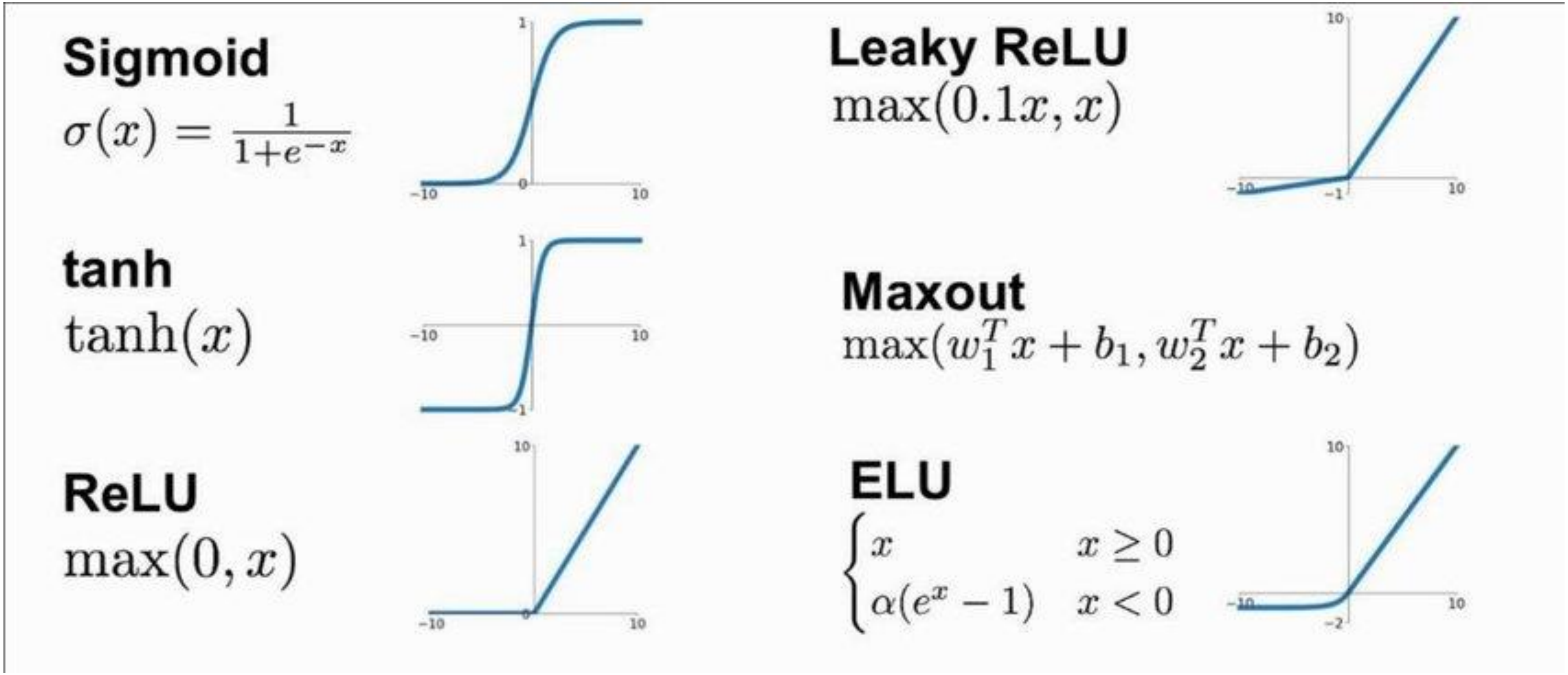
$$H^{(1)} = f_1\left(XW^{(1)} + b^{(1)}\right)$$

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}, \quad W^{(1)} = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1h} \\ w_{21} & w_{22} & \cdots & w_{2h} \\ \vdots & \vdots & \vdots & \vdots \\ w_{d1} & w_{d2} & \cdots & w_{dh} \end{pmatrix}, \quad \mathrm{H}^{(1)} = \begin{pmatrix} h_{11} & \boxed{h_{12}} & \cdots & h_{1h} \\ h_{21} & h_{22} & \cdots & h_{2h} \\ \vdots & \vdots & \vdots & \vdots \\ h_{n1} & h_{n2} & \cdots & h_{nh} \end{pmatrix}$$

Размеры матриц:

- $X \in R^{n\times d}$, $W^{(1)} \in R^{d\times h}$, $b^{(1)} \in R^{1\times h}$, h - число нейронов в первом слое
- $H^{(1)} \in R^{n\times h}$ - выход слоя
- $f_1(z)$ - **функция активации**

Функции активации f



Функции активации f

- **ReLU** (Rectified Linear Unit) – наиболее популярная функция активации в современных сетях. Походит для глубоких сетей – сетей с большим числом слоев.
- **Sigmoid** – использует, когда в сети 1-2 слоя. Также используется в выходном слое в задаче бинарной классификации
- **Tanh** – хорошая альтернатива sigmoid. Используется в скрытых слоях.

Вопрос

- Зачем использовать функции активации?
- Что будет, если их не использовать?

Ответ

- Пусть дана нейронная сеть без функций активации:

$$H^{(1)} = XW^{(1)} + b^{(1)}$$
$$O = H^{(1)}W^{(2)} + b^{(2)}$$

- Перепишем:

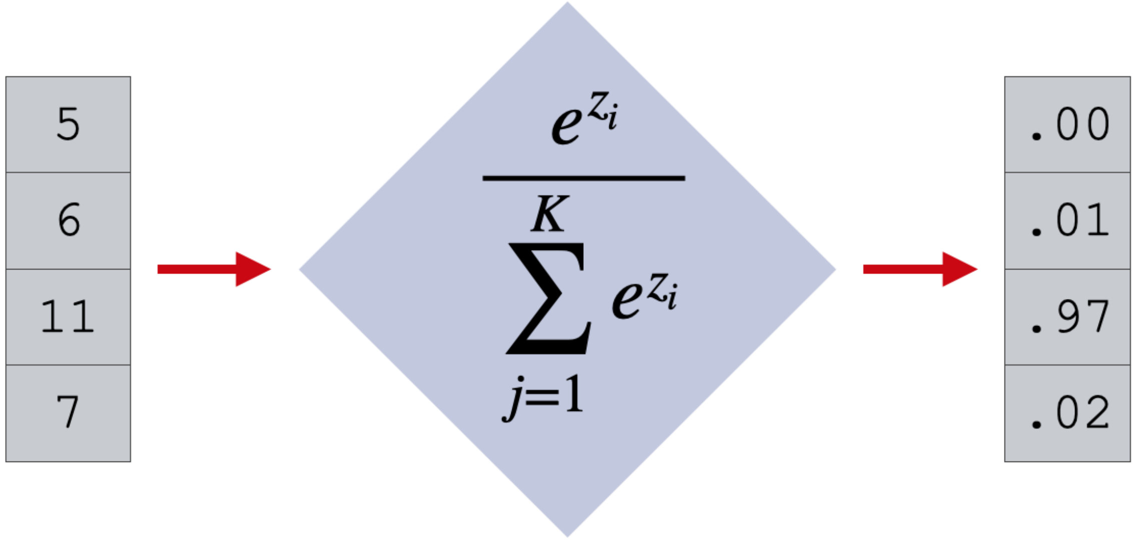
$$O = \left(XW^{(1)} + b^{(1)}\right)W^{(2)} + b^{(2)} = XW^{(1)}W^{(2)} + \left(b^{(1)}W^{(2)} + b^{(2)}\right)$$
$$O = XW + b$$

- В итоге получаем линейную зависимость

Функции активации softmax

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

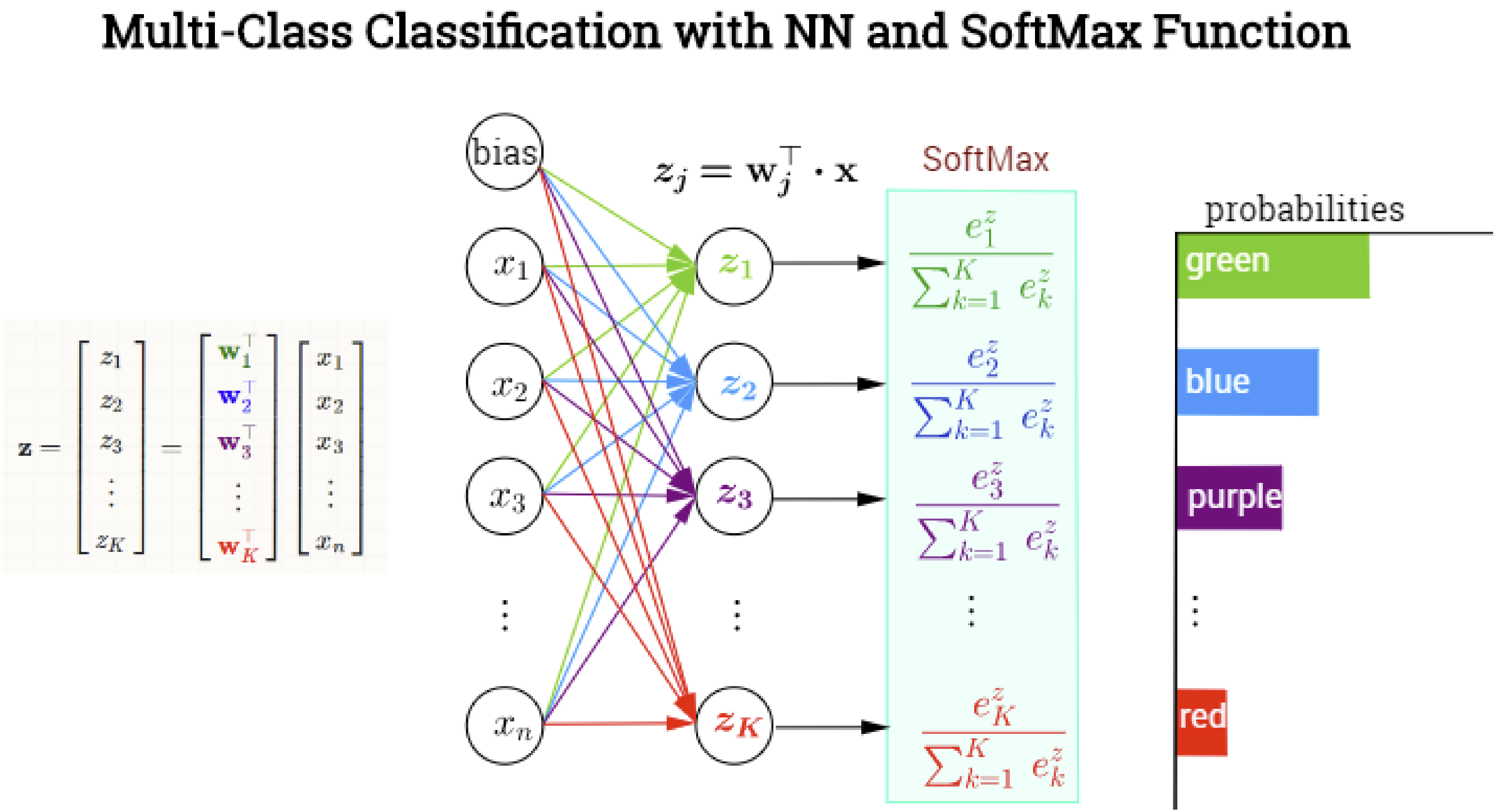
$$\mathrm{p_i} = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$



Функции активации softmax

- **Softmax** используется в выходном слое нейронной сети для решения задач многоклассовой классификации
- Выход функции можно интерпретировать как «вероятность» класса

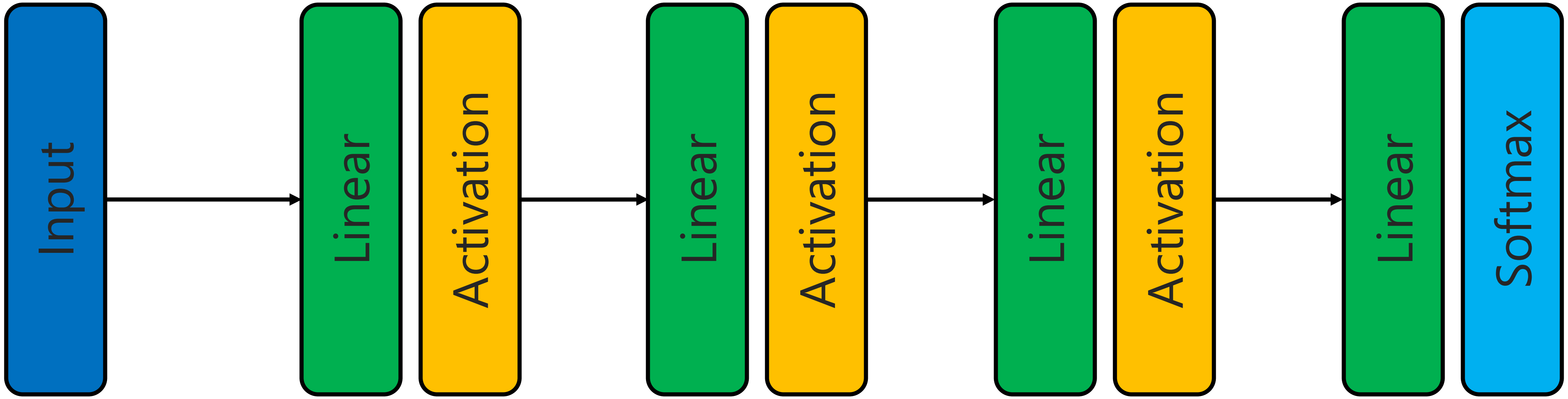
Функции активации softmax



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

Архитектура нейронной сети

- Последовательность линейных слоев и функций активации
- Обычно, во всех скрытых слоях используется одна функция активации
- На выходе softmax или sigmoid для классификации, линейная функция для регрессии



Теорема Цыбенко

- Пусть дана нейронная сеть с одним скрытым слоем
- В слое достаточно много нейронов
- Веса нейронов подобраны оптимально

Теорема Цыбенко утверждает, что такой сети достаточно для моделирования **любой функции**

Правда, обучить такую сеть может быть тяжело ☺

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

1	кот
2	жук
3	кот
4	пес
5	кот

Кросс-энтропийный функционал

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

-log 0.1 - log 0.05 - log 0.9 - log 0.6 - log 0.2

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:

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

Обучение нейронных сетей

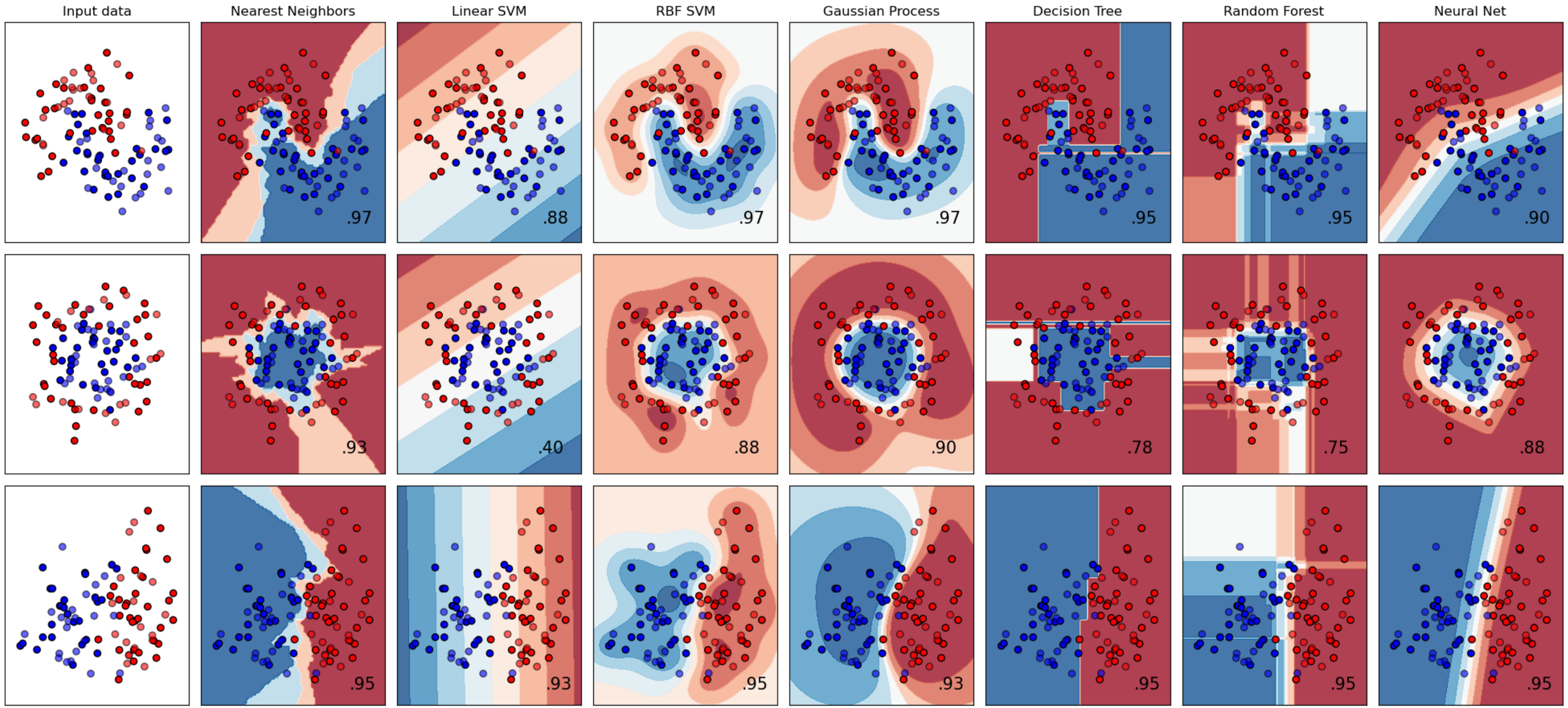
Градиентный спуск

- Нейронные сети обучаем с помощью градиентного спуска

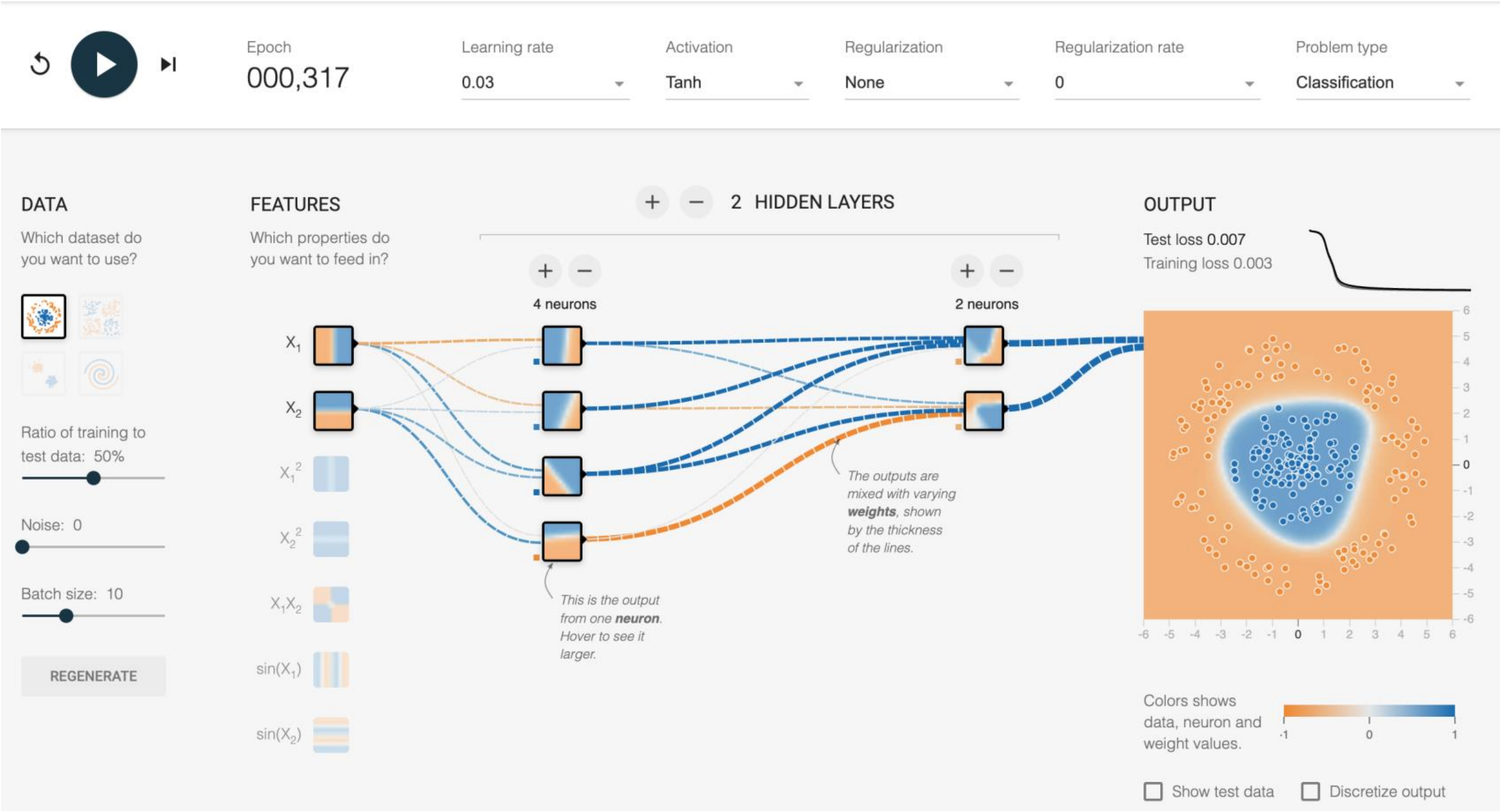
$$w = w - \alpha \frac{\partial L}{\partial w}$$

- Как посчитать градиент функции потерь по нужному весу сети?
- Про это поговорим на следующей лекции

Примеры


Input data
Nearest Neighbors
Linear SVM
RBF SVM
Gaussian Process
Decision Tree
Random Forest
Neural Net
.97
.88
.97
.97
.95
.95
.90
.93
.40
.88
.90
.78
.75
.88
.95
.93
.95
.93
.95
.95
.95

Демонстрация



https://playground.tensorflow.org