Системы обработки медиаданных

Нейронные сети. Перцептрон.

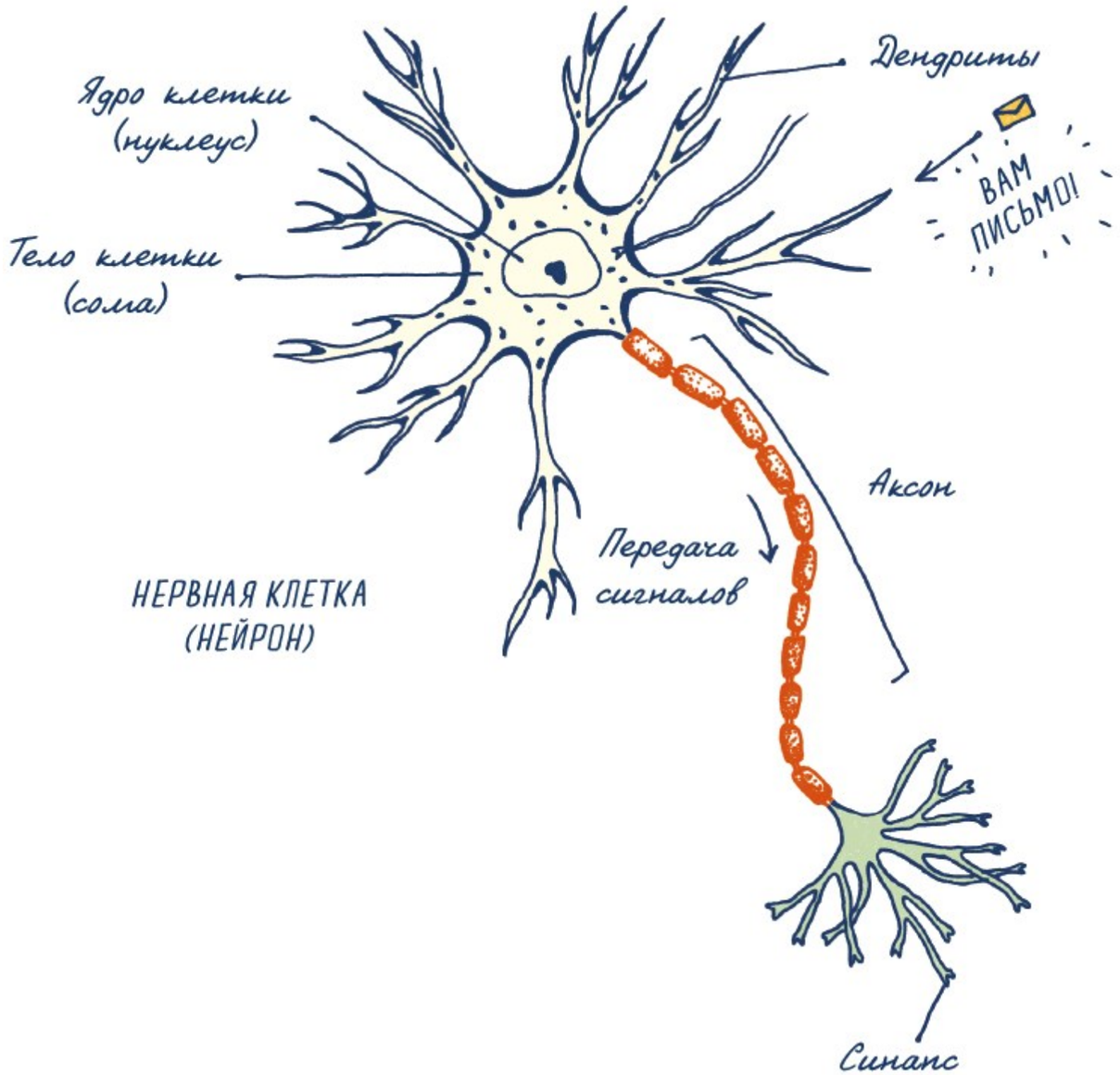
д.т.н., доцент Вашкевич М. И.

vashkevich@bsuir.by

Белорусский государственный университет информатики и радиоэлектроники

Кафедра электронных вычислительных средств

Нейрон: базовый вычислительный элемент мозга



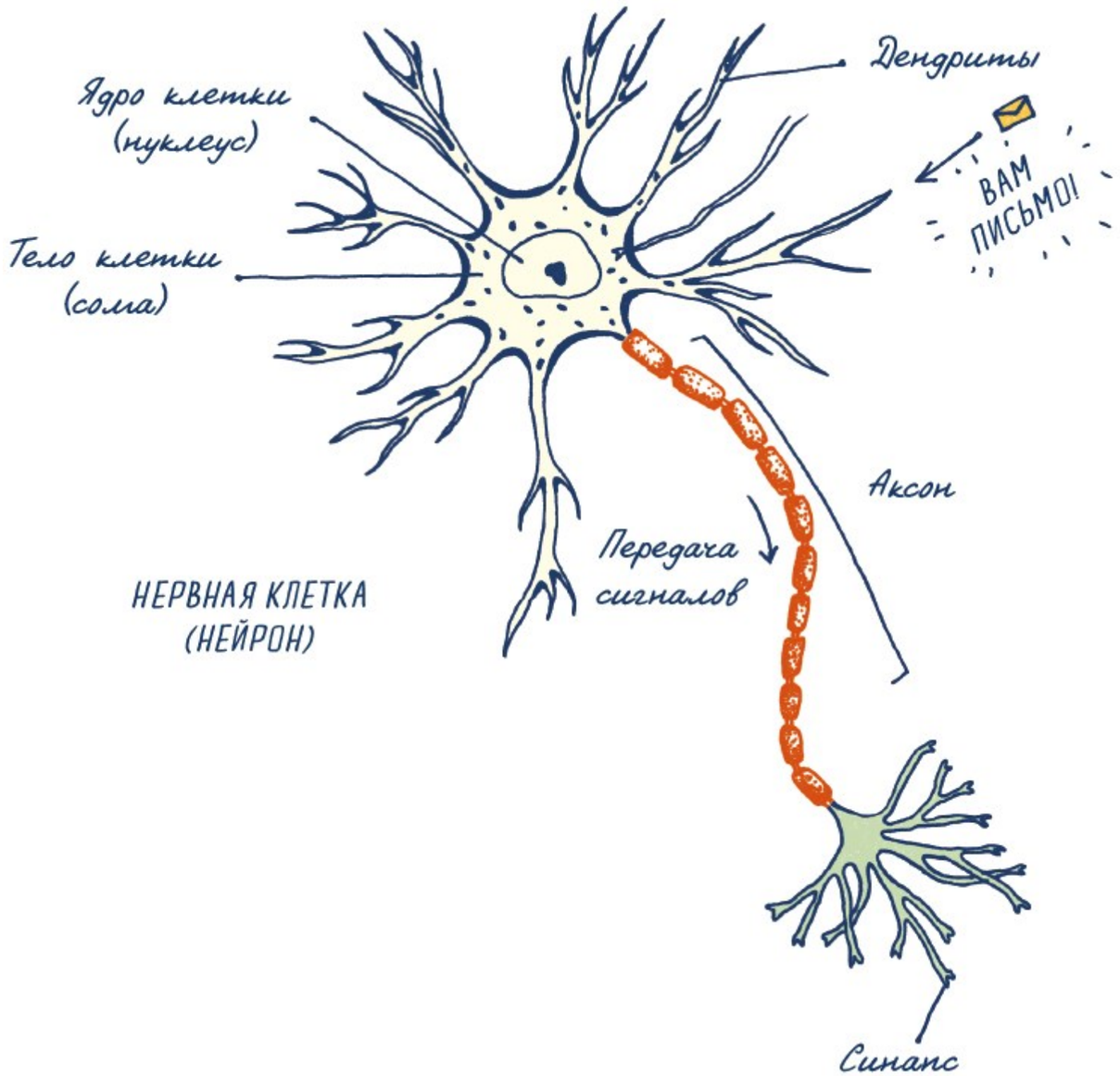
Нейроны – нервные клетки, которые передают сигналы в мозге.

Каждый нейрон взаимодействует с множеством других нейронов (≈ от 1000 до 10 000)

В нашем мозге около 10^{10} нейронов

Большинство нейронов, которые человека имеет в своей жизни, уже присутствуют при рождении

Нейрон: базовый вычислительный элемент мозга



Основные компоненты нейрона

– ***Тело клетки***, которое содержит информацию в ДНК ядра;

– ***Дендриты***, их число доходит до нескольких тысячи, обычно они короткие;

– ***Аксон*** – длинная структура, которая на конце распадается, возможно, на тысячи ветвей. Аксон может быть длиной до 1 метра.

Краткая история развития нейросетей

1943 – теоретическая модель У. Мак-Каллока и У. Питтса: Логическое исчисление идей, присущих нервной деятельности

- *Используя только математику и алгоритмы, построили модель того, как может работать нейронная сеть;*
- *Показали, что с помощью их сети можно построить любую вычислимую функцию;*
- *Можно ли было создать модель мыслей человеческого существа?*
- *Это работа считается «рождением» искусственного интеллекта.*

1949 – Дональд Хебб представил первую (чисто психологическую) теорию обучения – «правило Хебба»

- *Если два нейрона срабатывают вместе, они усиливают реакции друг друга и, вероятно, будут срабатывать вместе в будущем*

Краткая история развития нейросетей

1957 – Фрэнк Розенблатт предложил модель персептрона.
1969 – М. Минский и С. Паперт выступили с критикой персептрона.
1986 – Д. Румельхарт, Д. Хинтон и Р. Уильямс теорема об обратном распространении ошибки.
1989 – Ян Лекун разработал и обучил модель сверточной нейронной сети.
2006 – Д. Хинтон предложил алгоритм предобучения глубоких нейронных сетей.
2010 – В. Нэйр, Д. Хинтон предложили использовать функцию активации ReLU при обучении НС.
2012 – CNN AlexNet побеждает в соревновании ILSVRC (А. Крижевский, Д. Хинтоном и И. Суцкевер).
2014 – И. Гудфеллоу, Д. Бенджио предложили модель генеративно-состяза-тельной сети (GAN).

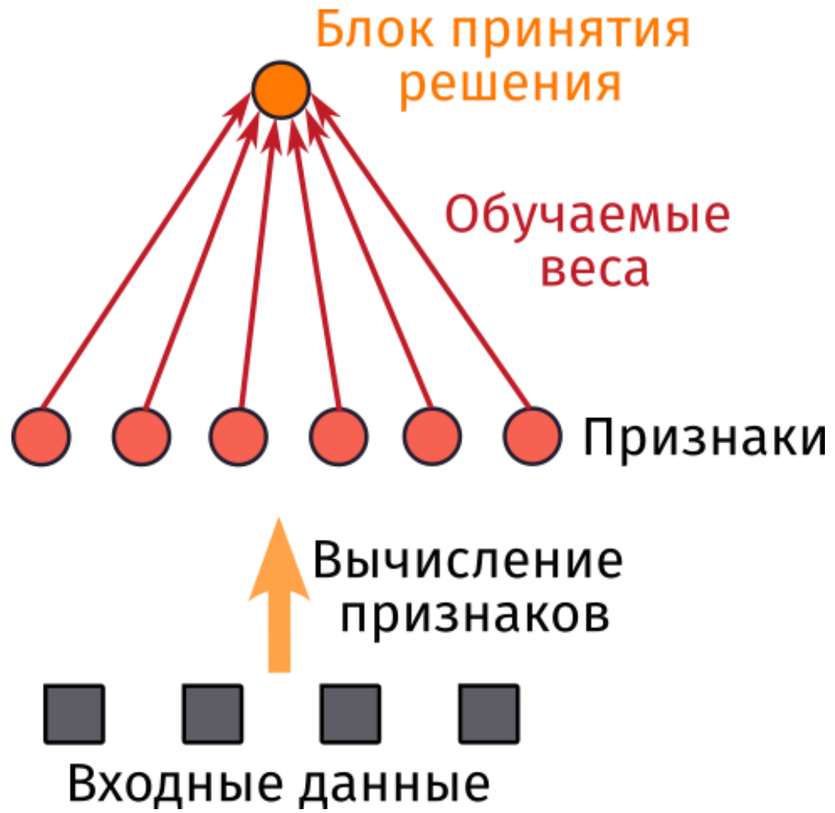
2017 – А. Васвани и др. предложили модель трнасформер (Whisper, ChatGPT).

Перцептрон: первое поколение нейронных сетей

Парадигма статистического обучения

1. Преобразовать входной вектор «сырых» данных в вектор признаков.

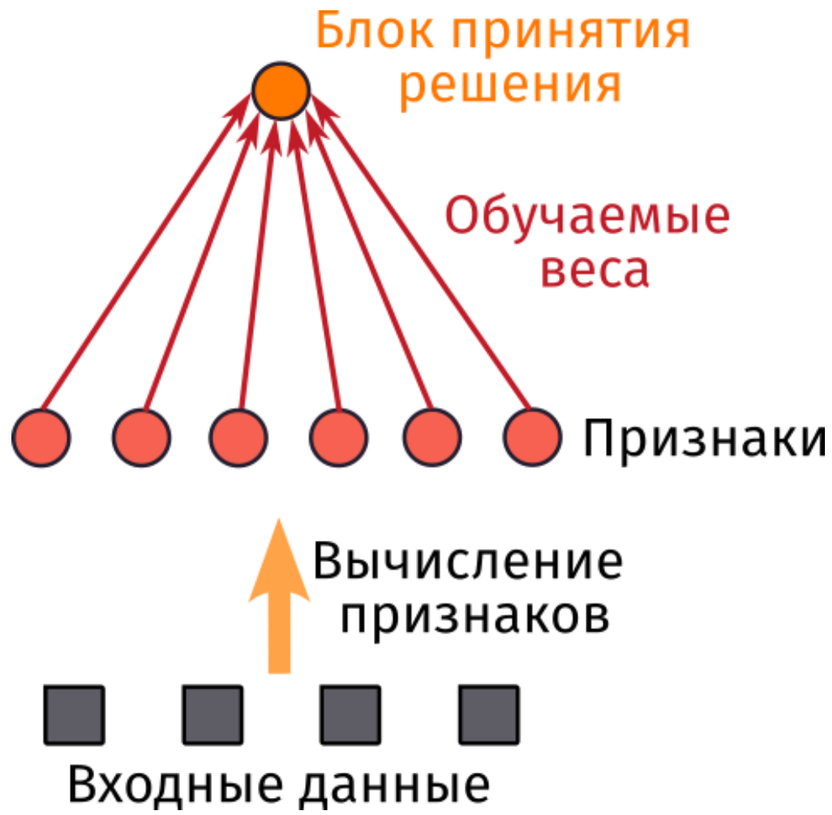
Признаки конструируются на основе знаний из предметной области.



Парадигма статистического обучения

1. Преобразовать входной вектор «сырых» данных в вектор признаков.

Признаки конструируются на основе знаний из предметной области.

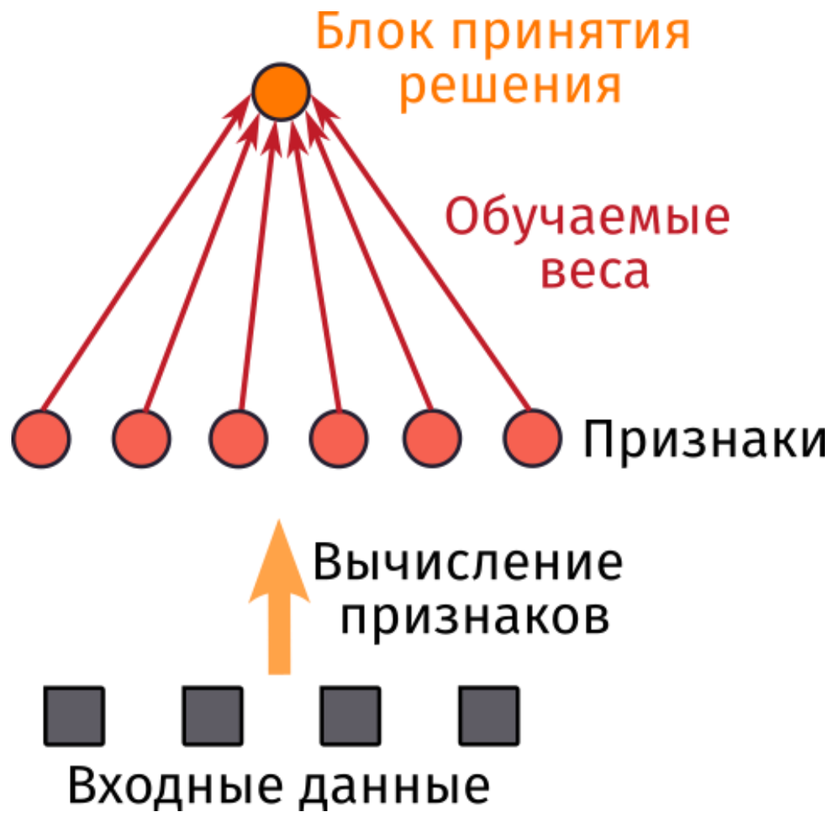


2. Обучить (оценить), какой вес имеет каждый отдельный признак, чтобы получить численное значение его вклада в решение.

Парадигма статистического обучения

1. Преобразовать входной вектор «сырых» данных в вектор признаков.

Признаки конструируются на основе знаний из предметной области.



2. Обучить (оценить), какой вес имеет каждый отдельный признак, чтобы получить численное значение его вклада в решение.

3. Сложить взвешенные значения признаков и определить превышает ли полученное значение порог. Если значение выше порога, то входной вектор является примером положительного класса, а иначе – отрицательного.

История перцептрона

•Фрэнк Розенблатт предложил модель персептрон в начале 1960-х.

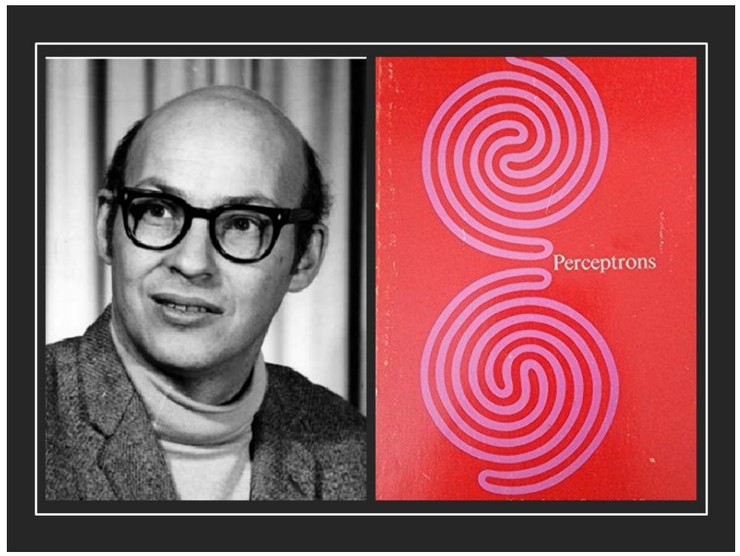
– Модель перцептрона имела достаточно эффективный алгоритм обучения.

– Было сделано много громких заявлений относительно того, чему может обучится данная модель.

•В 1969 г. М. Минский и С. Паперт выступили книгу «Перцептрон», которая показала ограничения персептрона.

– Многие исследователи сделали в то время неправильный вывод, подумав, что ограничения модели перцептрона относятся ко всем нейронным сетям.

• Процедура обучения персептрона по-прежнему широко используется сегодня для задач с огромными векторами признаков



Перцептрон Розенблатта

• ***Перцептрон*** – линейная модель классификации. Все объекты в тренировочной выборке помечены одной из меток $+1$ или -1.

• Решение на основе модели перцептрона

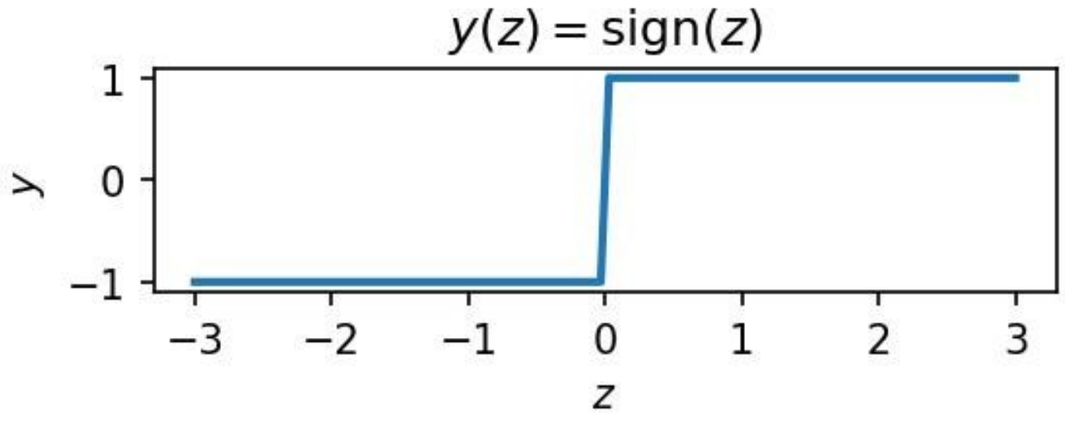
1) Вычислить взвешенную сумму от входных нейронов (признаков) и прибавить смещение

$$z = \sum_{j=1}^{d} w_j x_j + b$$

где x_j – выходные данные, w_j – веса, b – смещение.

2) Установить выход в «1», если взвешенная сумма больше нуля и в «-1» иначе

$$y = \text{sign}(z) = \begin{cases} 1, & z \geq 0 \\ -1, & z < 0 \end{cases}.$$



Геометрическая интерпретация перцептрона

• Разделение между двумя классами (в двумерном случае) определяется выражением

$$w_1 x_1 + w_2 x_2 + b = 0.$$

• Таким образом, однослойный перцептрон является ***линейной дискриминантной функцией***.

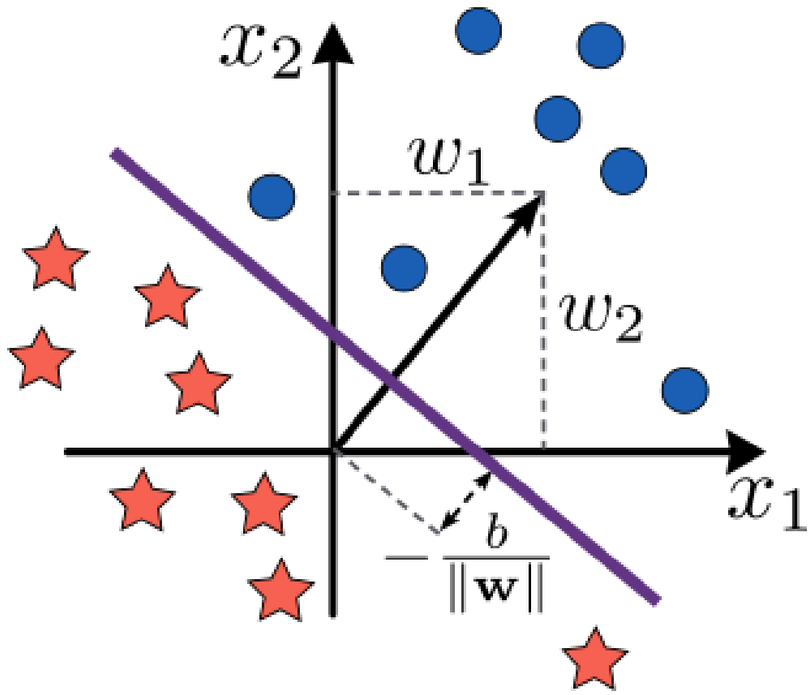
• Геометрическая интерпретация перцептрона задается уравнением

$$x_2 = \frac{w_1}{w_2} x_1 + \frac{b}{w_2}.$$

• Набор весов определяет наклон прямой.

• Смещение определяет расстояние до начала координат.

• Вектор весов всегда перпендикулярен вектору весов.



Перцептрон Розенблатта

$$y = \text{sign}\left(\sum_{j=1}^{d} w_j x_j + b\right).$$

Перцептрон Розенблатта

$$y = \text{sign}\left(\sum_{j=1}^{d} w_j x_j + b\right).$$

Введем обозначения

$$\mathbf{x} = [1 \quad x_1 \quad x_2 \quad \dots \quad x_d]^T, \qquad \mathbf{w} = [b \quad w_1 \quad w_2 \quad \dots \quad w_d]^T.$$

Перцептрон Розенблатта тогда примет вид:

$$y = \text{sign}(\mathbf{w}^T \mathbf{x}).$$

Перцептрон Розенблатта

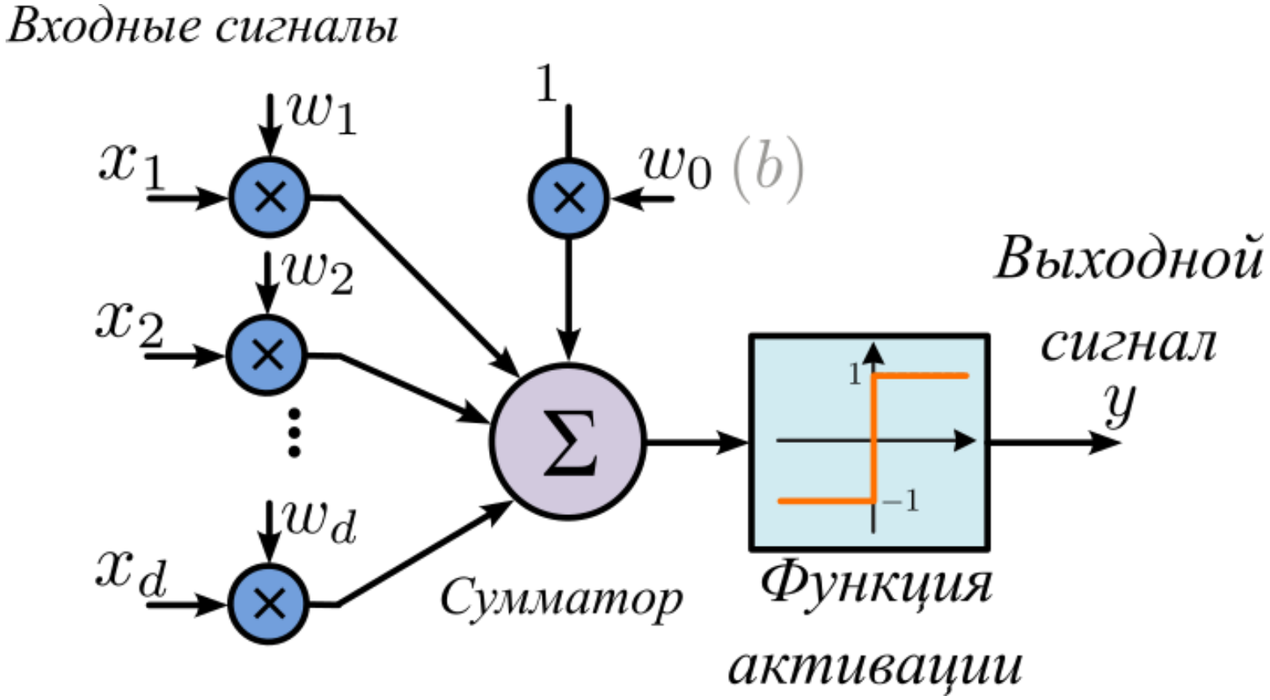
$$y = \text{sign}\left(\sum_{j=1}^{d} w_j x_j + b\right).$$

Введем обозначения

$$\mathbf{x} = [1 \quad x_1 \quad x_2 \quad \dots \quad x_d]^T, \qquad \mathbf{w} = [b \quad w_1 \quad w_2 \quad \dots \quad w_d]^T.$$

Перцептрон Розенблатта тогда примет вид:

$$y = \text{sign}(\mathbf{w}^T\mathbf{x}).$$



Перцептрон Розенблатта

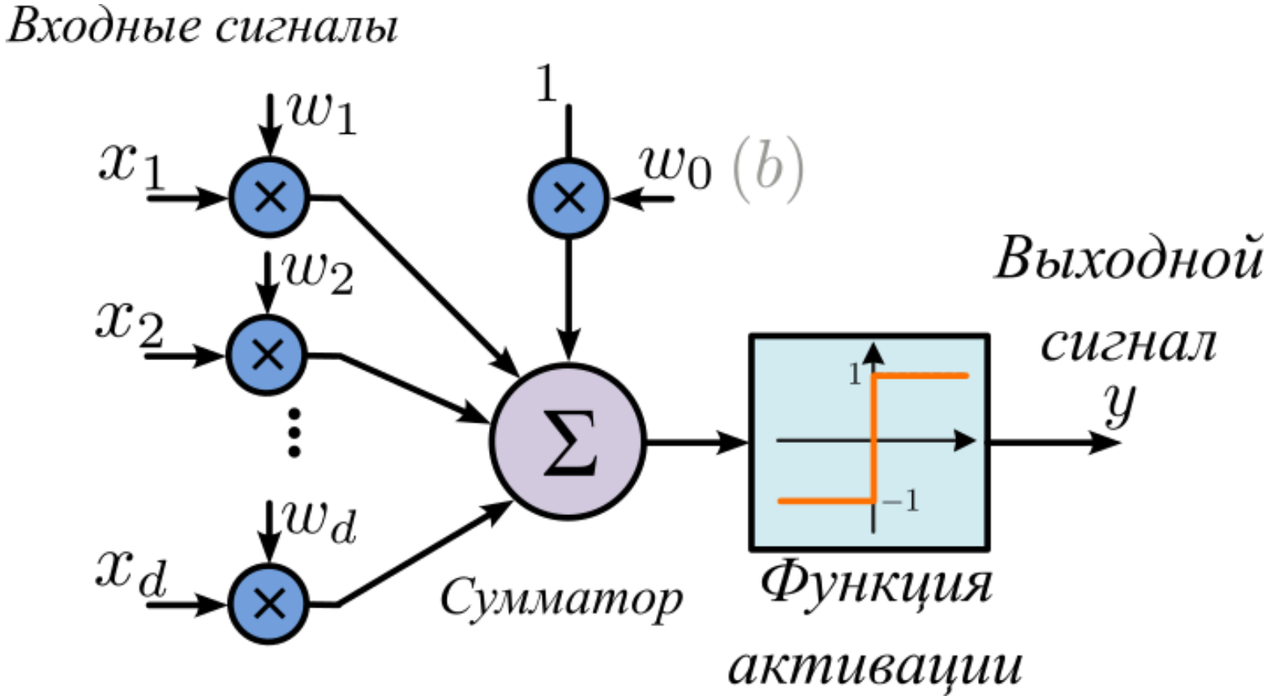
$$y = \text{sign}\left(\sum_{j=1}^{d} w_j x_j + b\right).$$

Введем обозначения

$$\mathbf{x} = [1 \quad x_1 \quad x_2 \quad \dots \quad x_d]^T, \qquad \mathbf{w} = [b \quad w_1 \quad w_2 \quad \dots \quad w_d]^T.$$

Перцептрон Розенблатта тогда примет вид:

$$y = \text{sign}(\mathbf{w}^T\mathbf{x}).$$



Как обучать перцептрон?

Обучение перцептрона

• Пусть имеется набор данных $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$, причем $y_i \in \{-1,\ 1\}$.

• Перцептрон перебирает обучающий набор, обновляя вектор весов всякий раз, как встречает неправильно классифицированный пример.

• Пусть $\mathbf{x}_i$ – неправильно классифицированный положительный пример, т.е.

$$\mathbf{w}^T \mathbf{x}_i < 0$$

• Значит нам нужен такой вектор $\mathbf{w}_{new}$, чтобы $\mathbf{w}_{new}^T \mathbf{x}_i > \mathbf{w}^T \mathbf{x}_i$. При этом решающая граница сдвинется вперед и (хочется надеяться) оставит $\mathbf{x}_i$ позади.

• Вопрос: как найти $\mathbf{w}_{new}$?

Обучение перцептрона

Задача

Пусть $\mathbf{x}_i$ – неправильно классифицированный положительный пример, т.е.

$$\mathbf{w}^T\mathbf{x}_i < 0$$

Как найти $\mathbf{w}_{new}$ такой, чтобы

$$\mathbf{w}_{new}^T\mathbf{x}_i > \mathbf{w}^T\mathbf{x}_i \, ?$$

Решение

$$\mathbf{w}_{new} = \mathbf{w} + \eta\mathbf{x}_i,$$

где величина $0 < \eta < 1$ называется ***скоростью обучения***.

Доказательство:

$$\mathbf{w}_{new}^T\mathbf{x}_i = (\mathbf{w} + \eta\mathbf{x}_i)^T\mathbf{x}_i = \mathbf{w}^T\mathbf{x}_i + \eta\mathbf{x}_i^T\mathbf{x}_i > \mathbf{w}^T\mathbf{x}_i$$

Общее правило обучения перцептрона (при неправильной классификации):

$$\mathbf{w}_{new} = \mathbf{w} + \eta y_i\mathbf{x}_i.$$

Алгоритм обучения перцептрона

Алгоритм `Perceptron (D,`η`)` – обучение перцептрона для бинарной классификации.

Вход: Помеченные данные $D \subseteq \mathbb{R}^d$, η – скорость обучения;

Выход: Вектор весов $\mathbf{w}$, определяющий классификатор $\hat{y} = \text{sign}(\mathbf{w}^T\mathbf{x})$;

Начало:

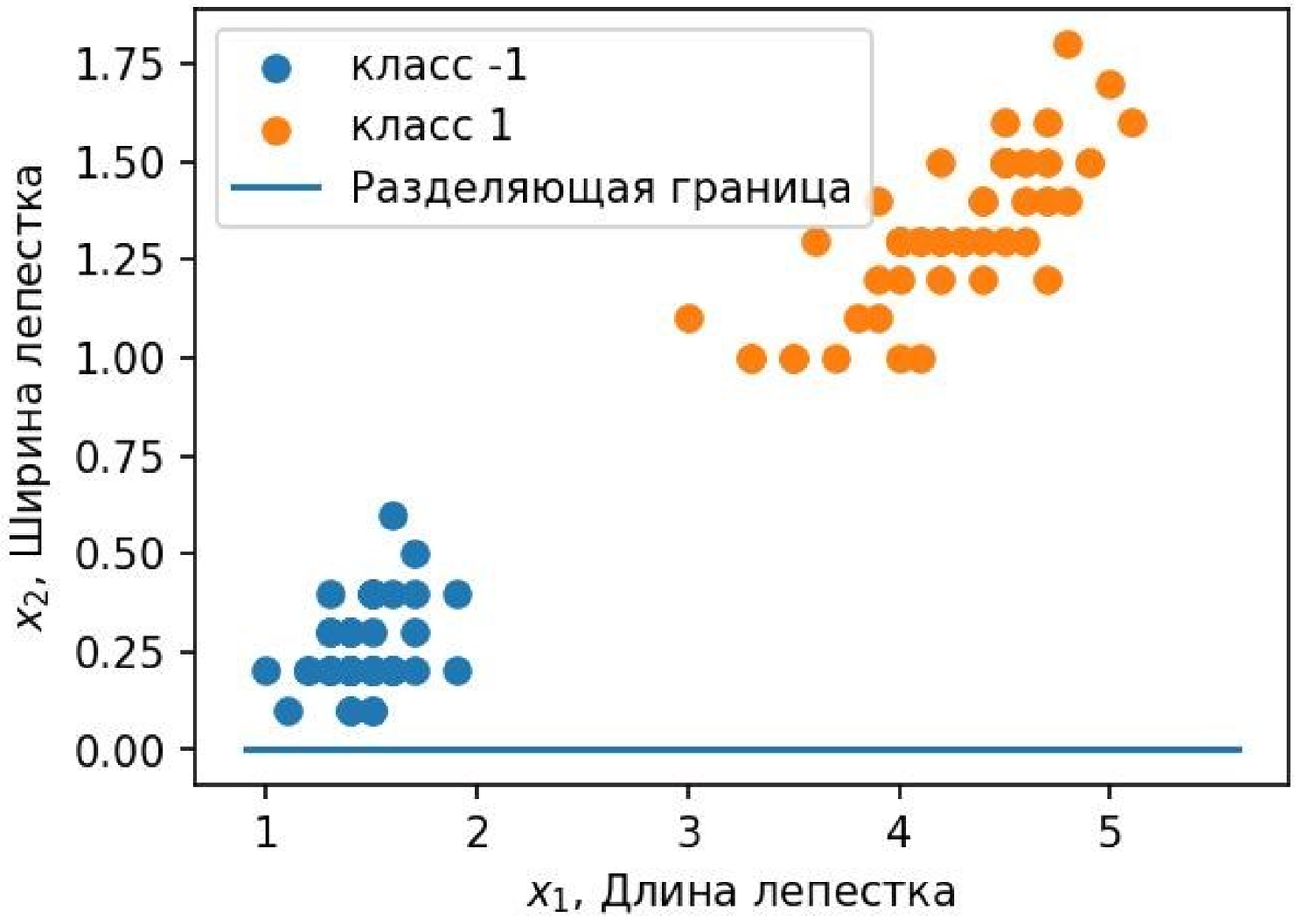
```
1 w ← 0;                                  // можно инициализировать иначе
2  converged ← False
3 while converged = False do
4     converged ← True
5     for x_i, y_i in  D do
6         if (y_i w^T x_i<0) then
7             w ← w + η y_i x_i;
8             converged ← False // w изменился, значит алгоритм не сошелся
9         end if
10    end for
11 end while
```

Конец

Пример обучения перцептрона

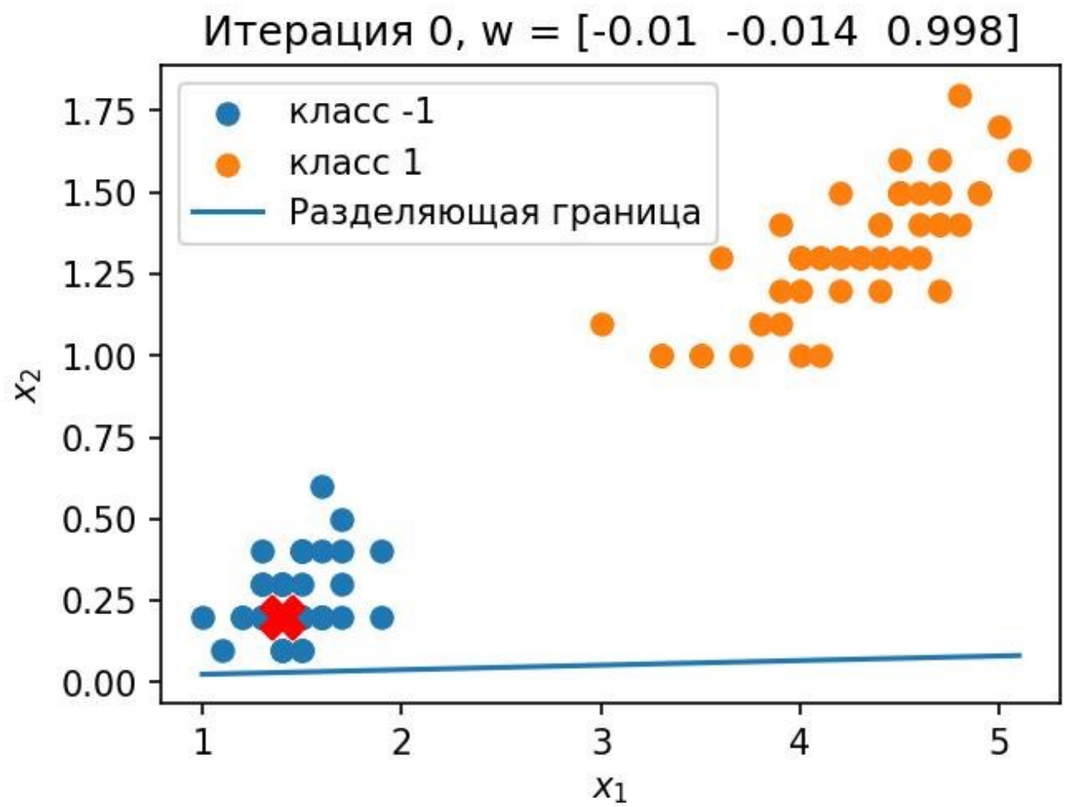
$\mathbf{w} = [0 \quad 0 \quad 1]$ – начальные значения

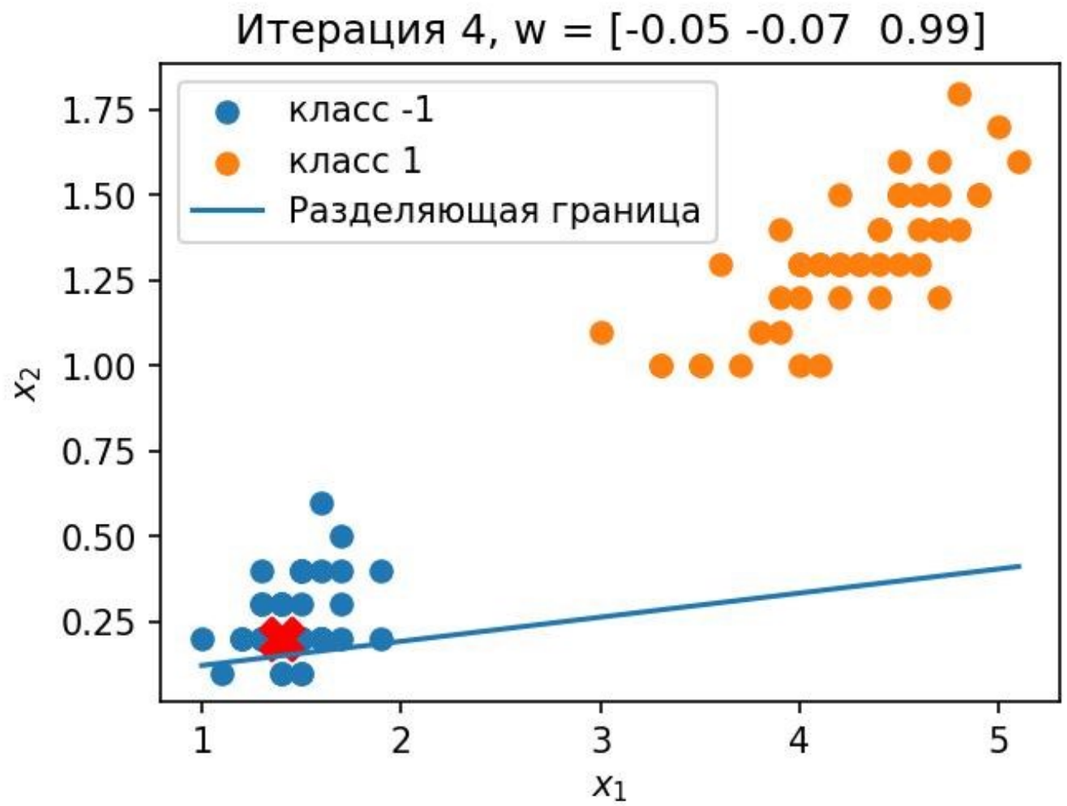
Разделяющая граница определяется уравнением: $\mathbf{w}^T\mathbf{x} = 0$



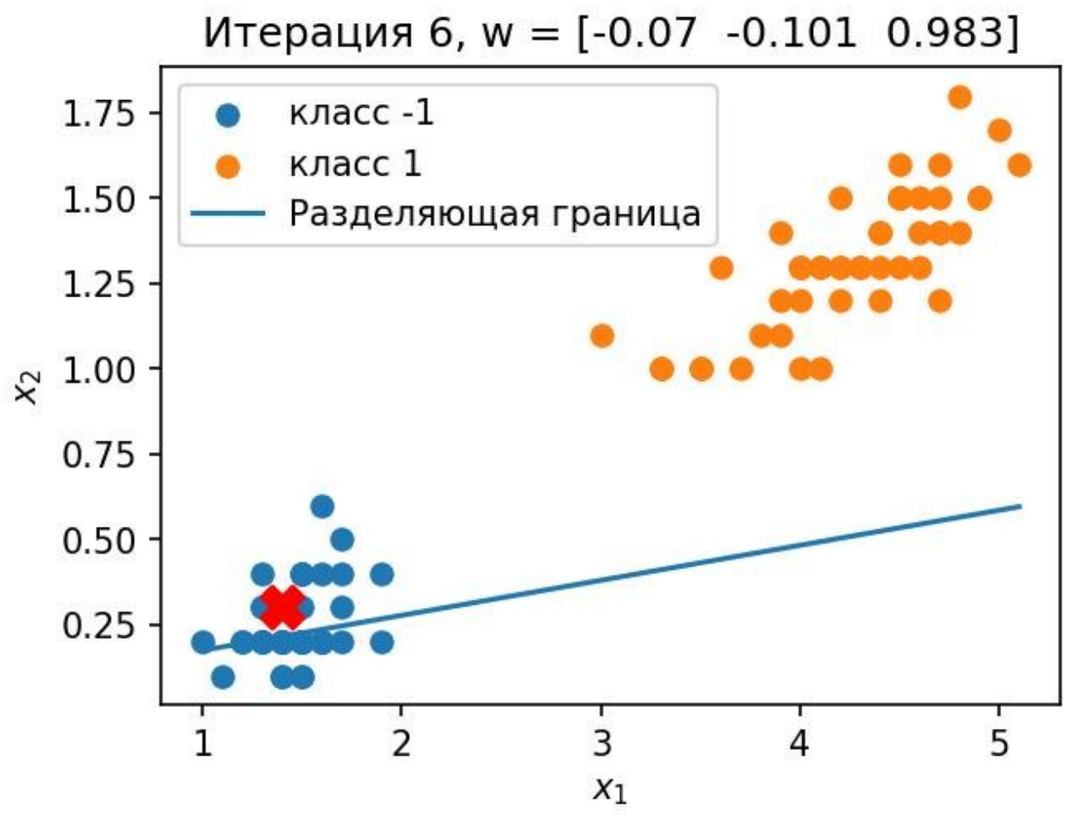
Пример обучения перцептрона

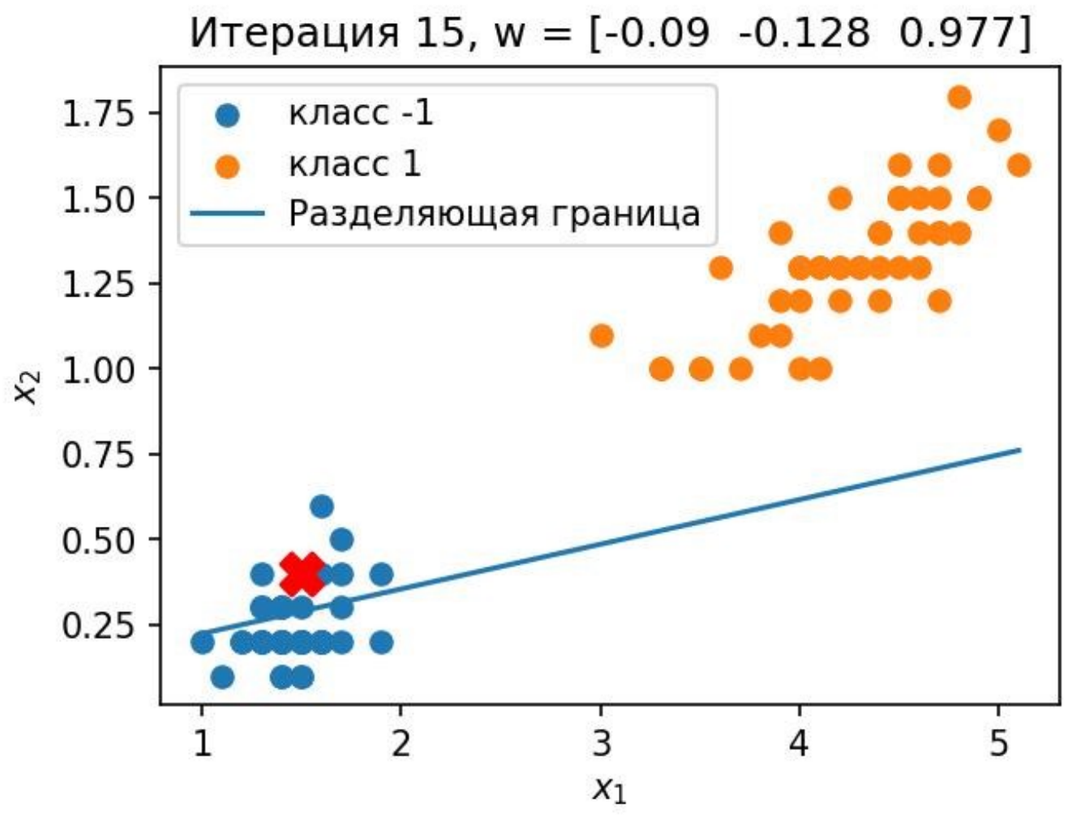
$\eta = 0{,}008$



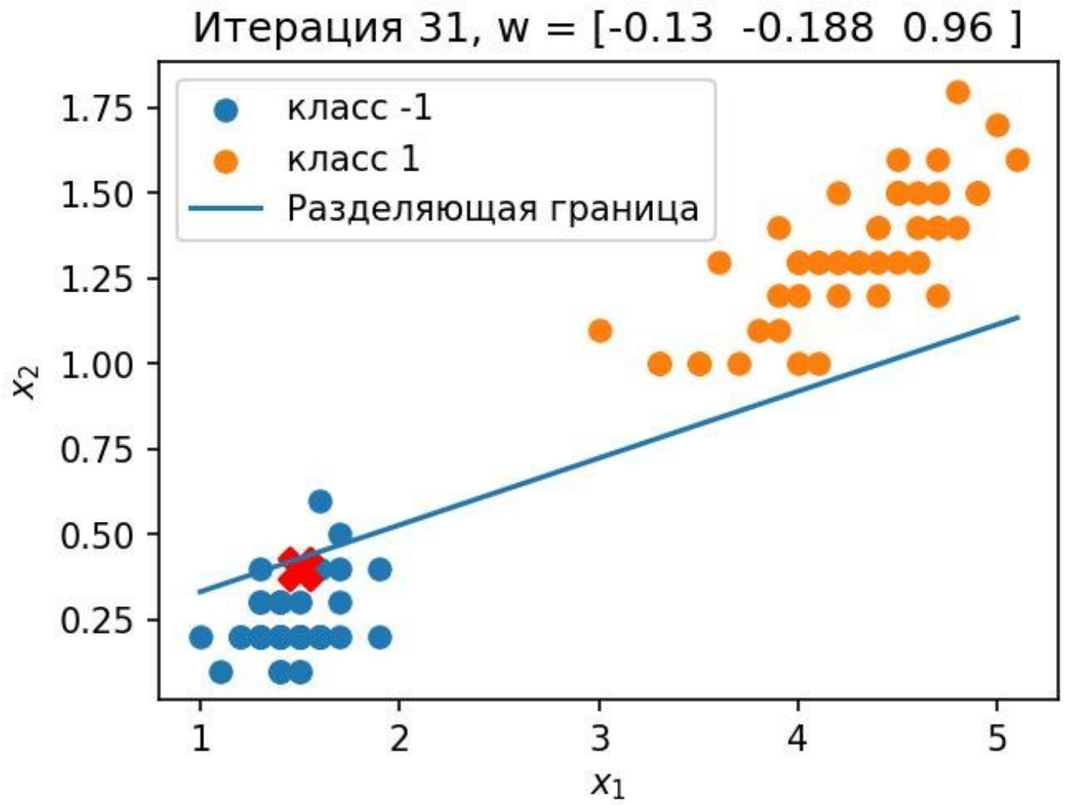


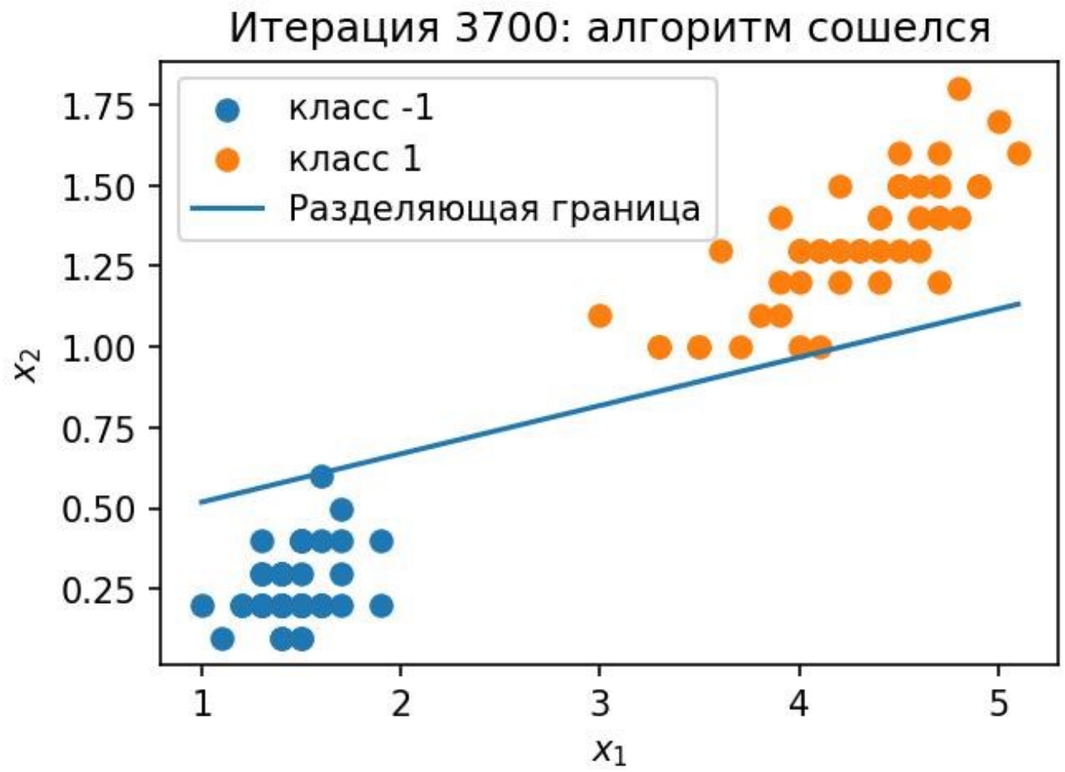
Пример обучения перцептрона



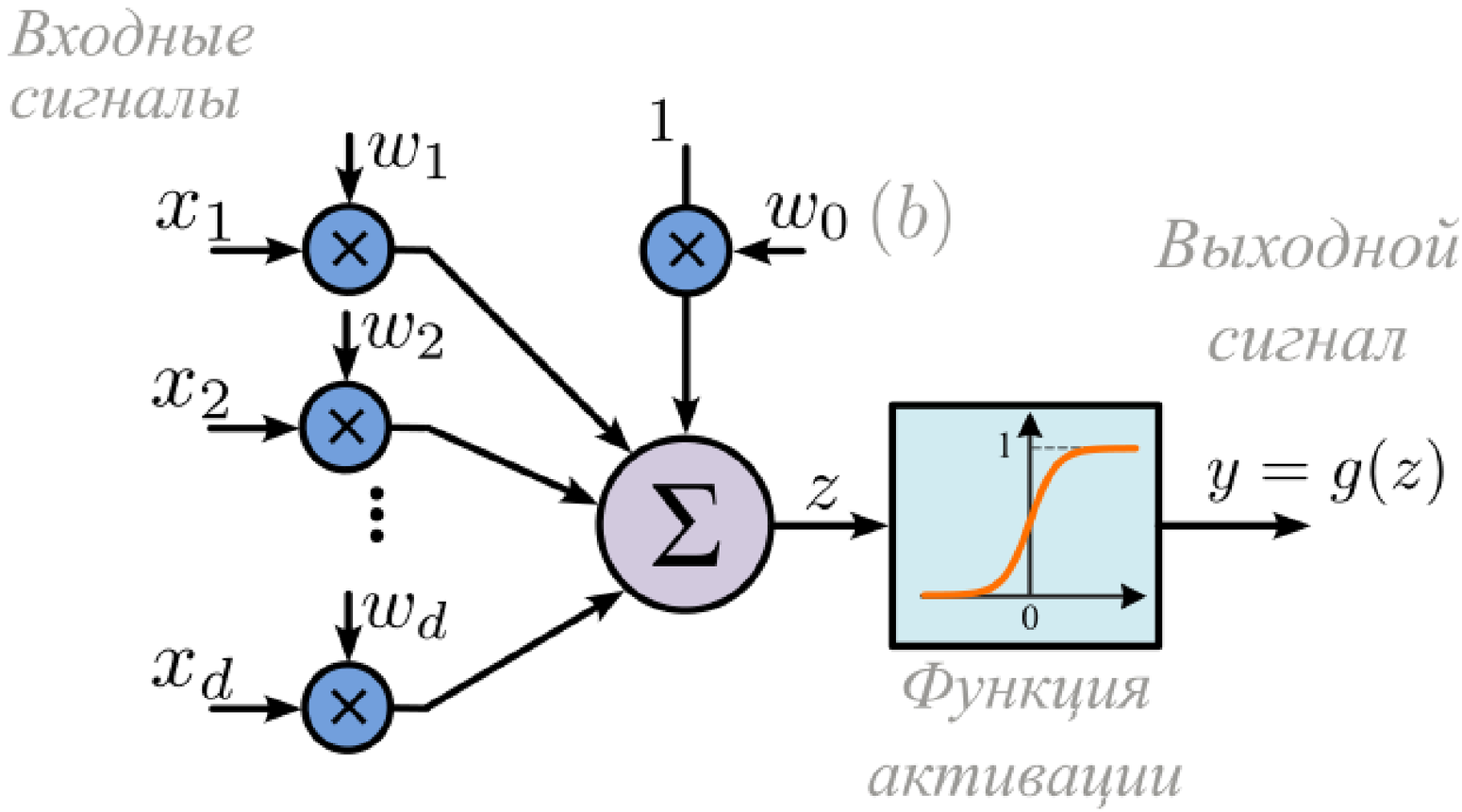


Пример обучения перцептрона



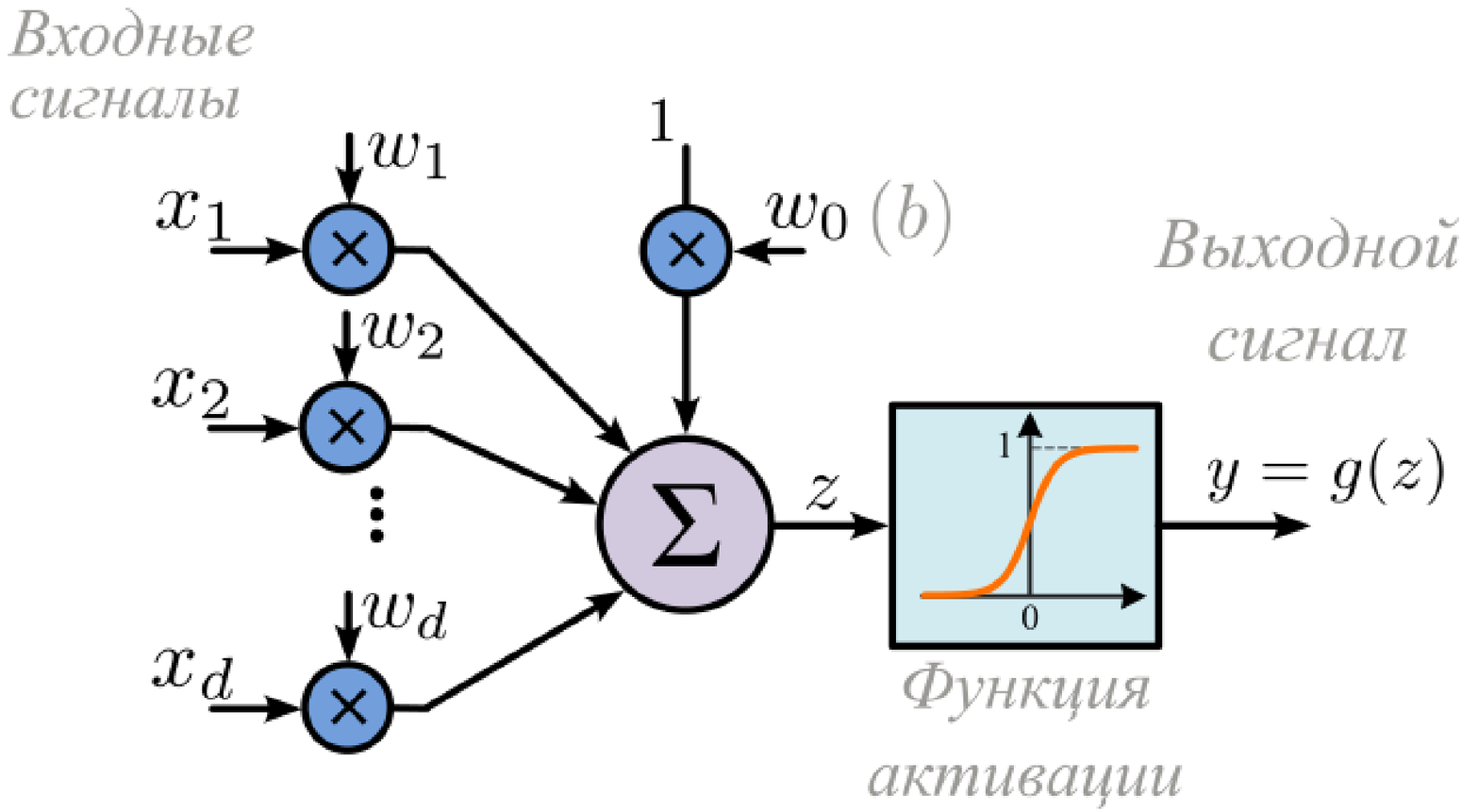


Базовый блок нейронной сети



• Смещение b представляет из себя пороговое значение для срабатывания перцептрона.

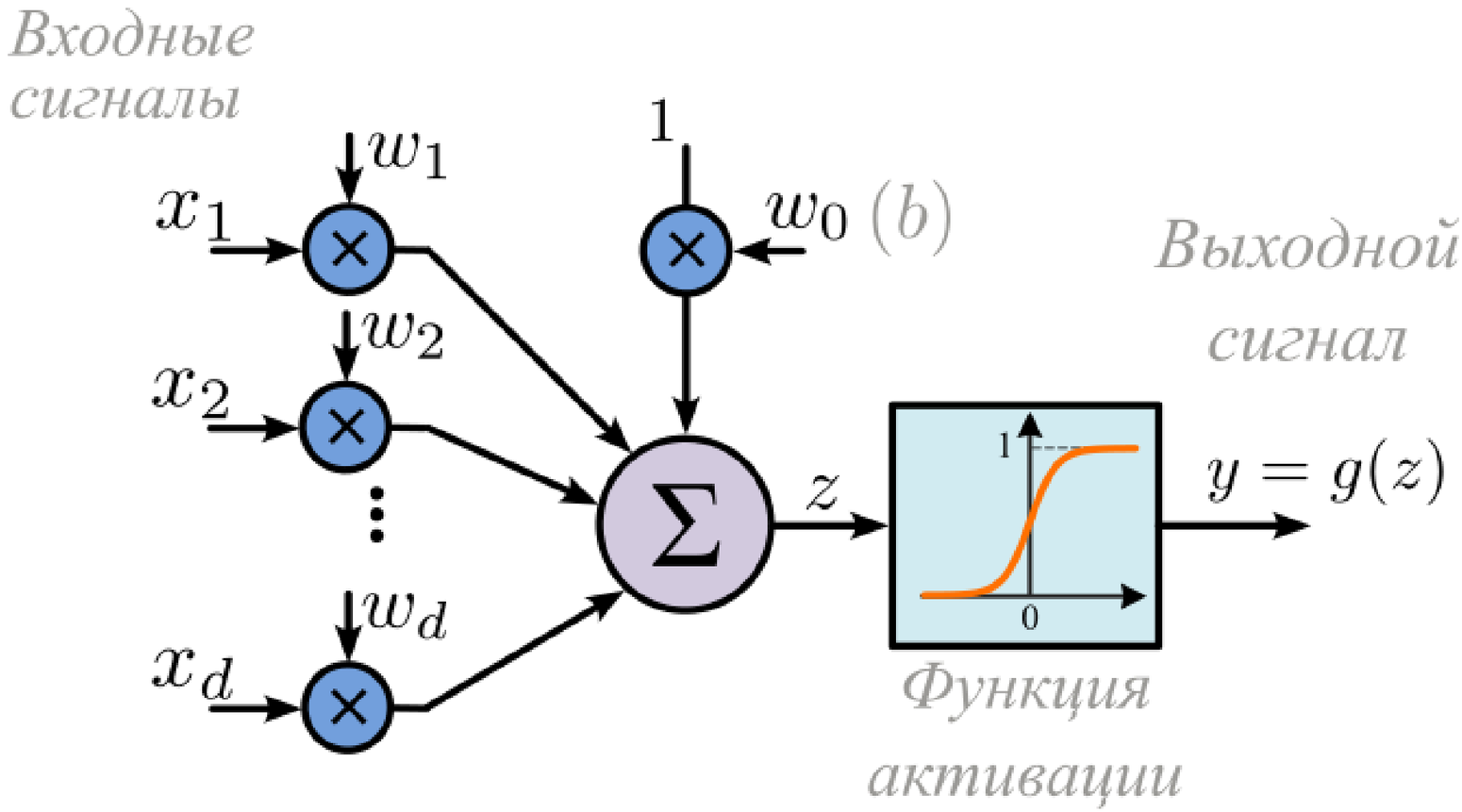
Базовый блок нейронной сети



• Смещение b представляет из себя пороговое значение для срабатывания перцептрона.

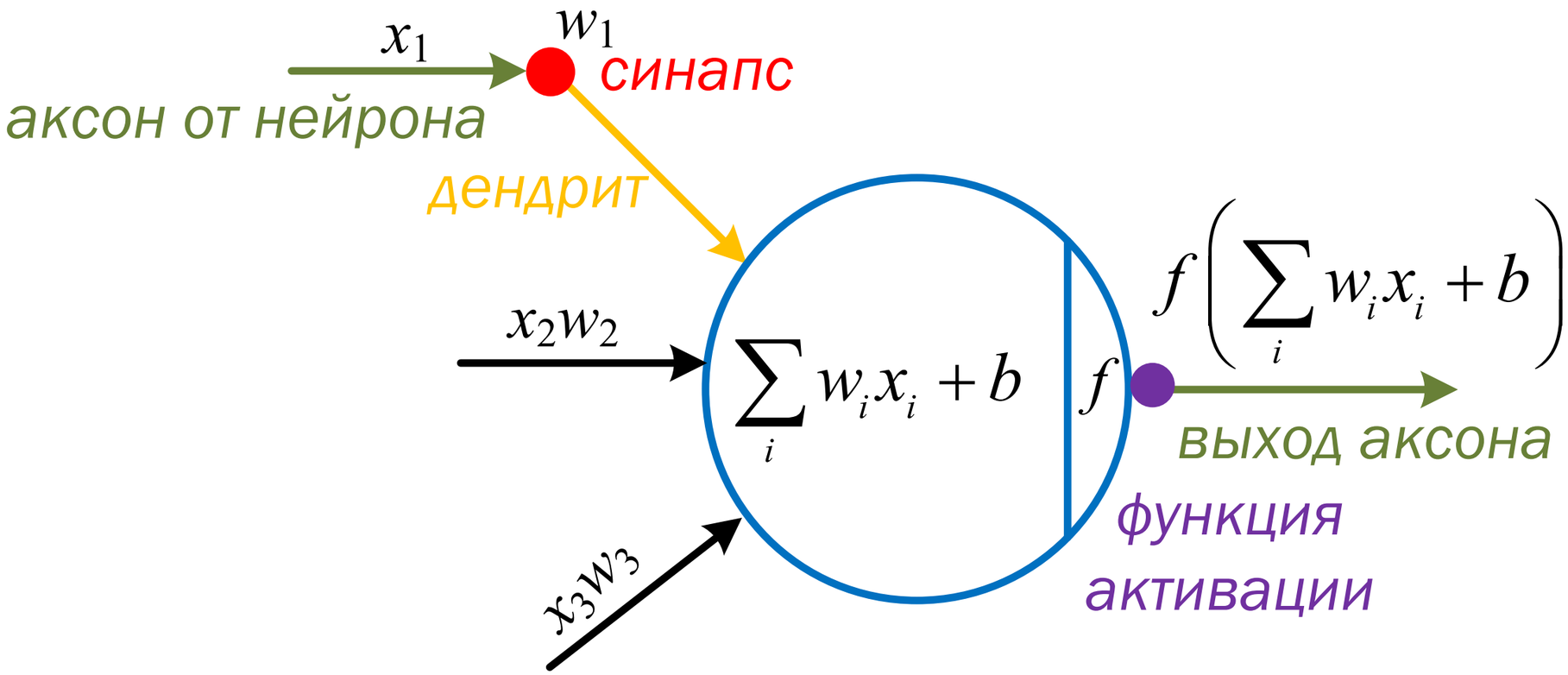
• Активационная функция не обязательно должна быть пороговой (sign).

Базовый блок нейронной сети



• Смещение b представляет из себя пороговое значение для срабатывания перцептрона.
• Активационная функция не обязательно должна быть пороговой (sign).
• Параметрами персептрона (которые определяют его поведение) являются его веса и смещение.

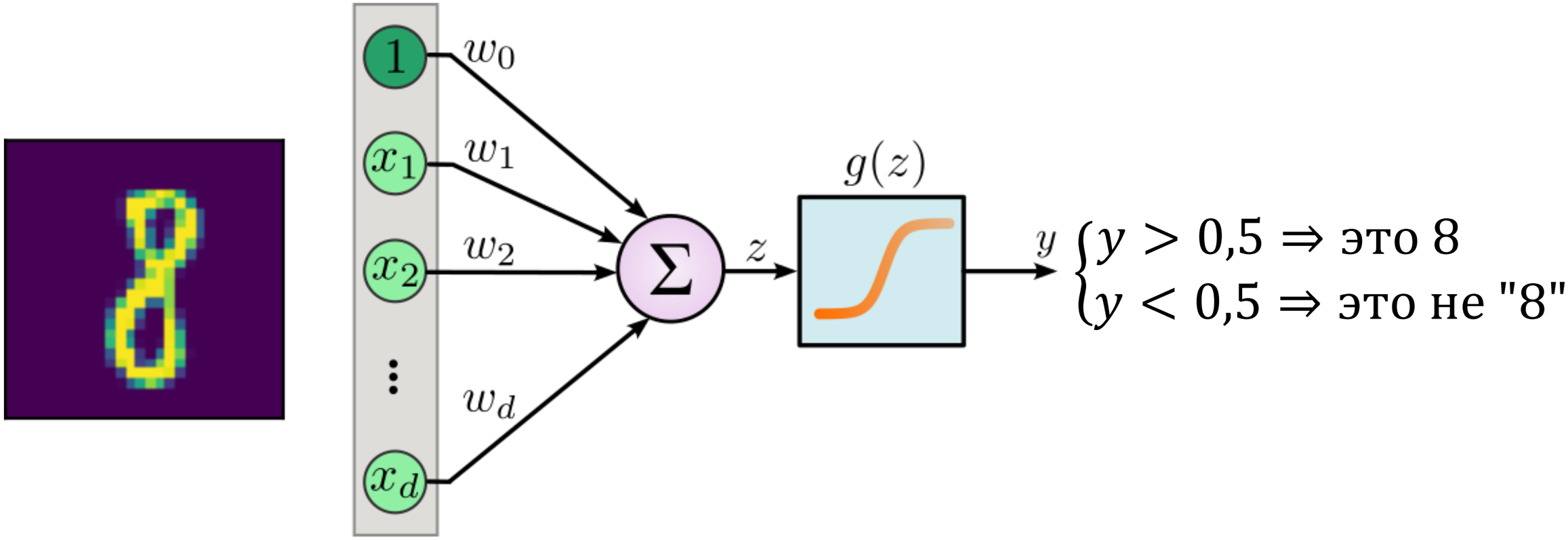
Структурный граф нейрона



Математическая модель нейрона искусственной нейронной сети

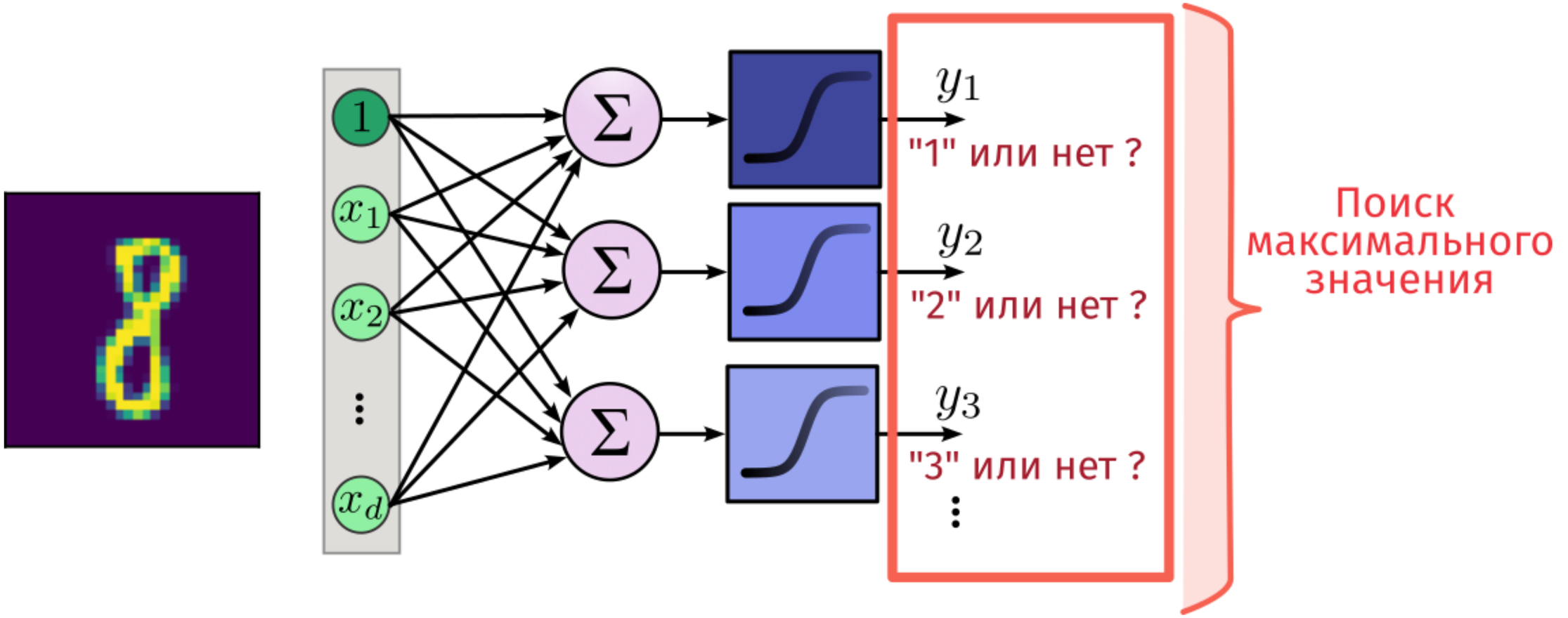
Нейронная сеть – это ориентированный граф, состоящий из узлов, соединенных синаптическими и активационными связями.

Единичный нейрон



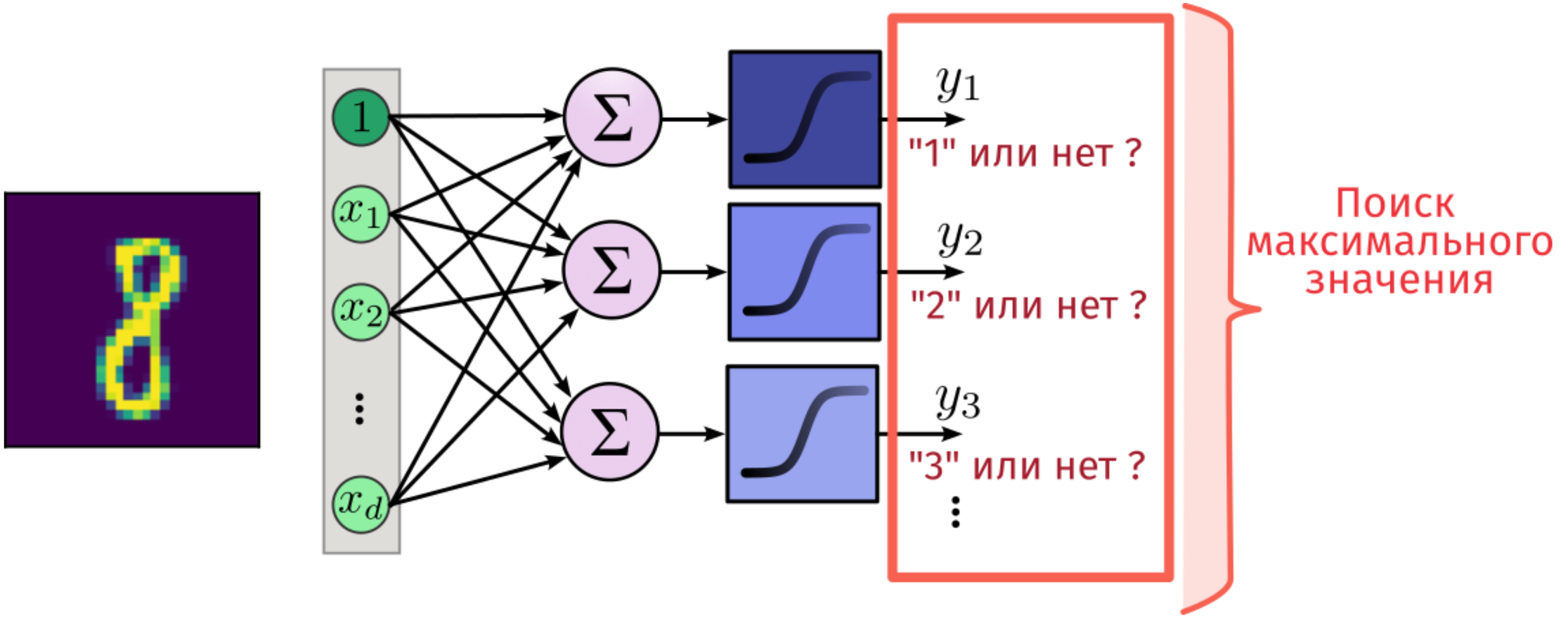
Единичный нейрон может решить только задачу бинарной классификации.

Слой нейронов



- 10 нейронов ⟶ 10 классов

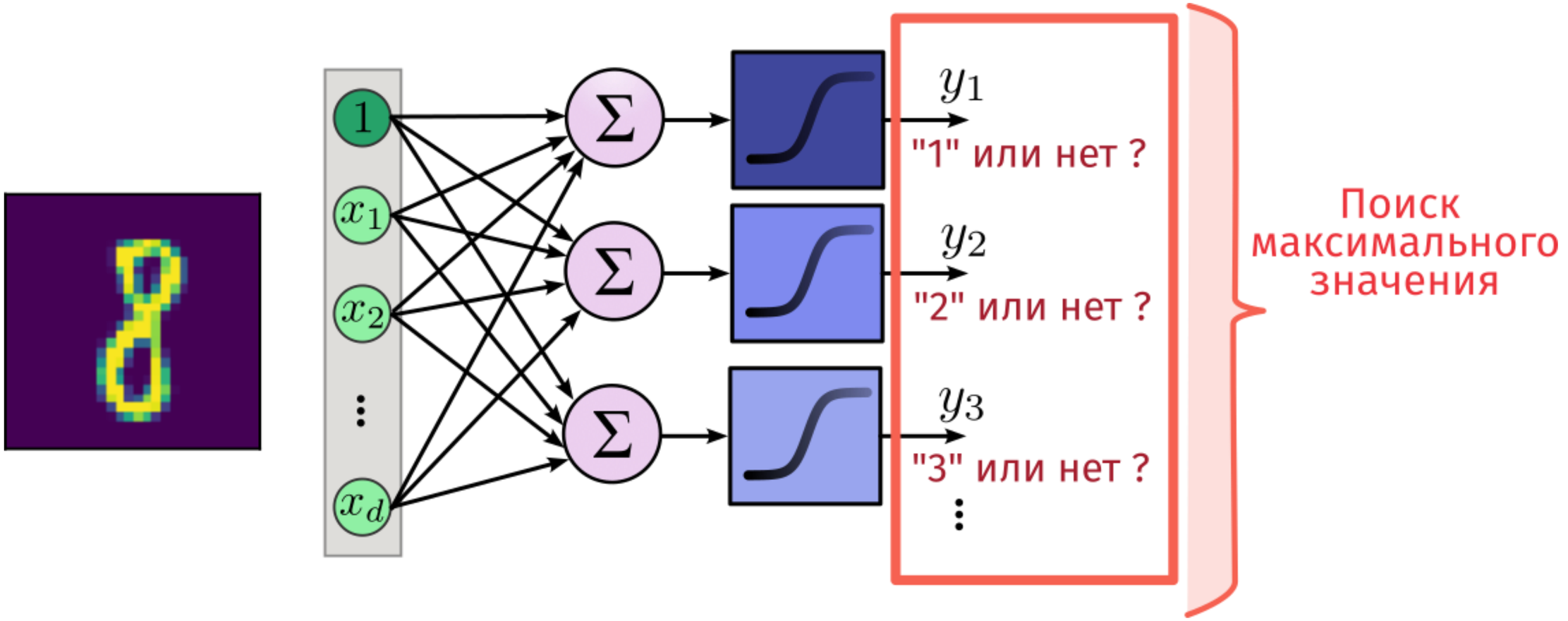
Слой нейронов



- 10 нейронов ⟶ 10 классов

- Все нейронны работают независимо (нет общих весов)

Слой нейронов



• 10 нейронов ⟶ 10 классов

• Все нейронны работают независимо (нет общих весов)

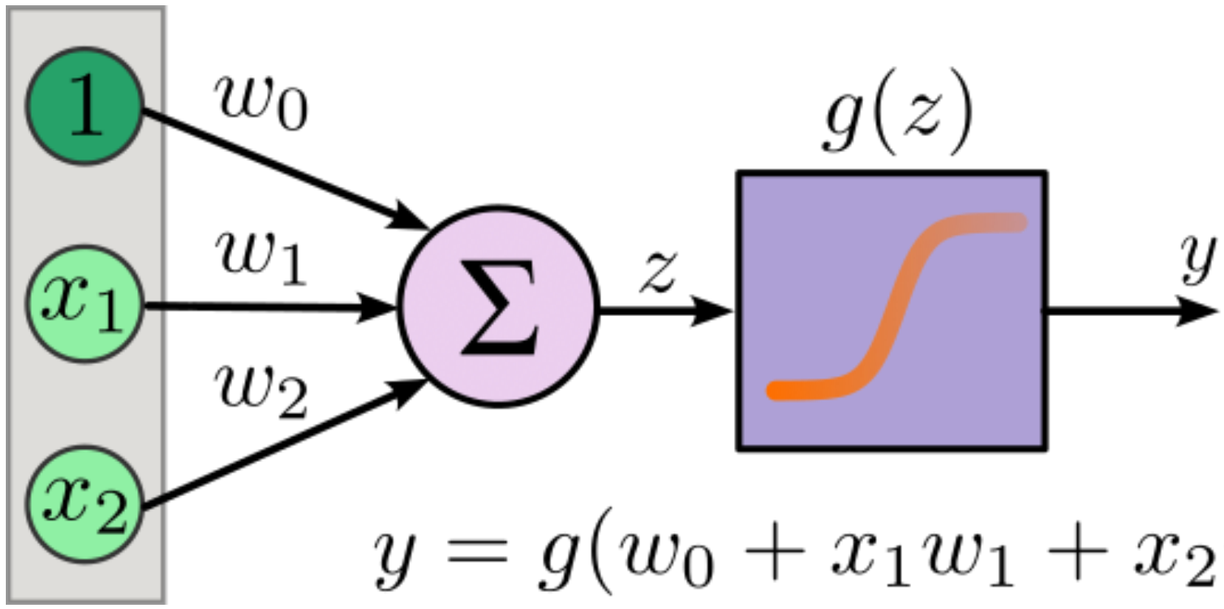
• Слой нейронов дают на выходе несколько значений, результат определяется максимальным значением на выходе.

Ограничения линейных классификаторов

• Линейные классификаторы классифицируют входы на основе линейной комбинации признаков x_i

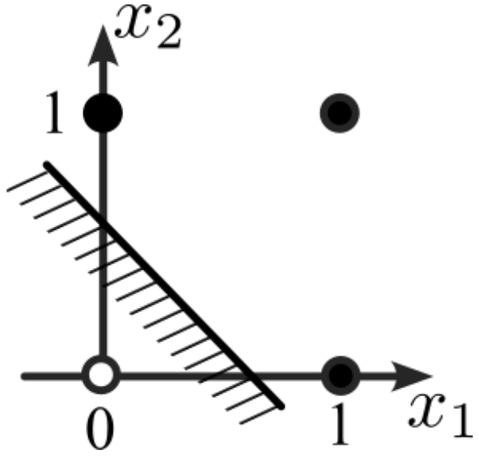
• В многих ситуациях решения имеют нелинейную зависимость от входных признаков

Ограничения линейных классификаторов



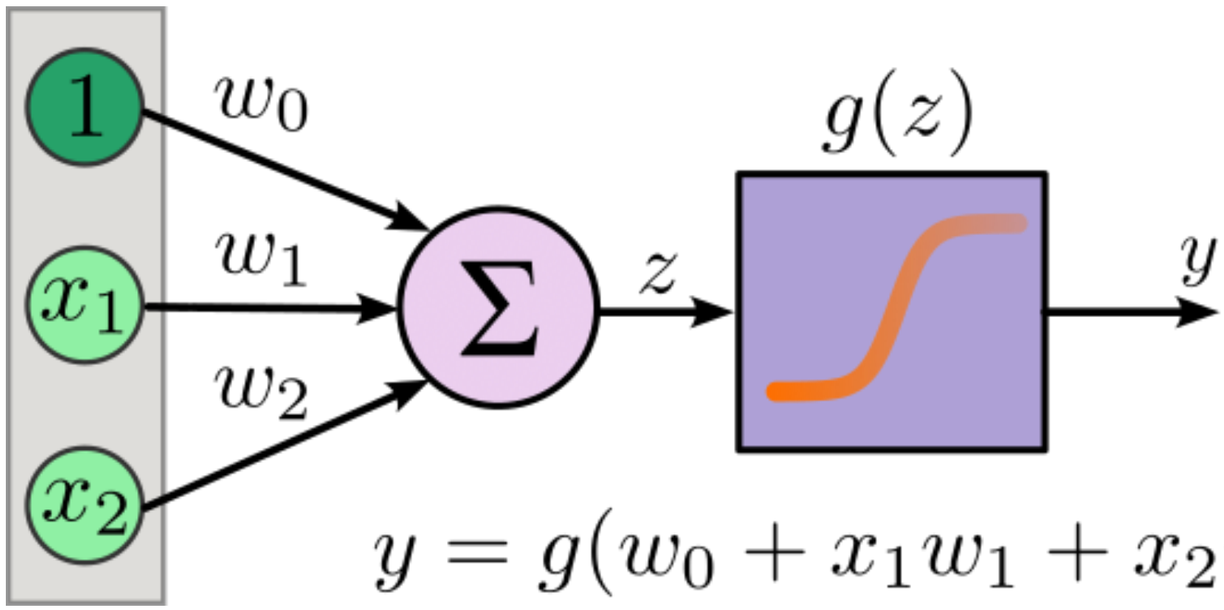
$$y = g(w_0 + x_1 w_1 + x_2 w_2)$$

$$\begin{cases} y > \text{thr} \Rightarrow True \\ y < \text{thr} \Rightarrow False \end{cases}$$



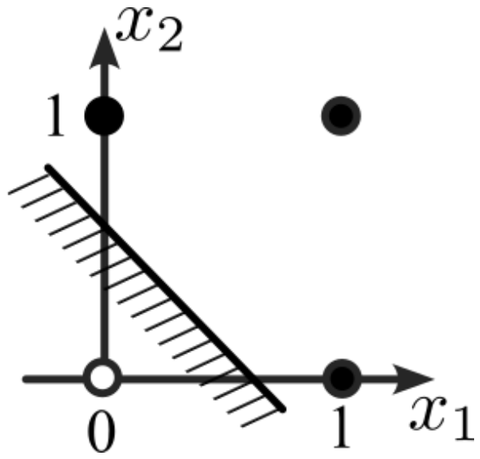
$x_1\ OR\ x_2$

Ограничения линейных классификаторов

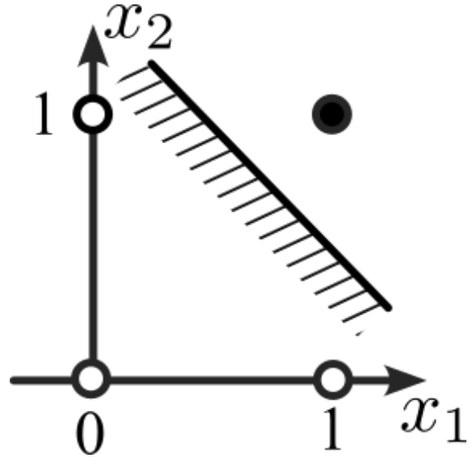


$$y = g(w_0 + x_1 w_1 + x_2 w_2)$$

$$\begin{cases} y > \text{thr} \Rightarrow True \\ y < \text{thr} \Rightarrow False \end{cases}$$



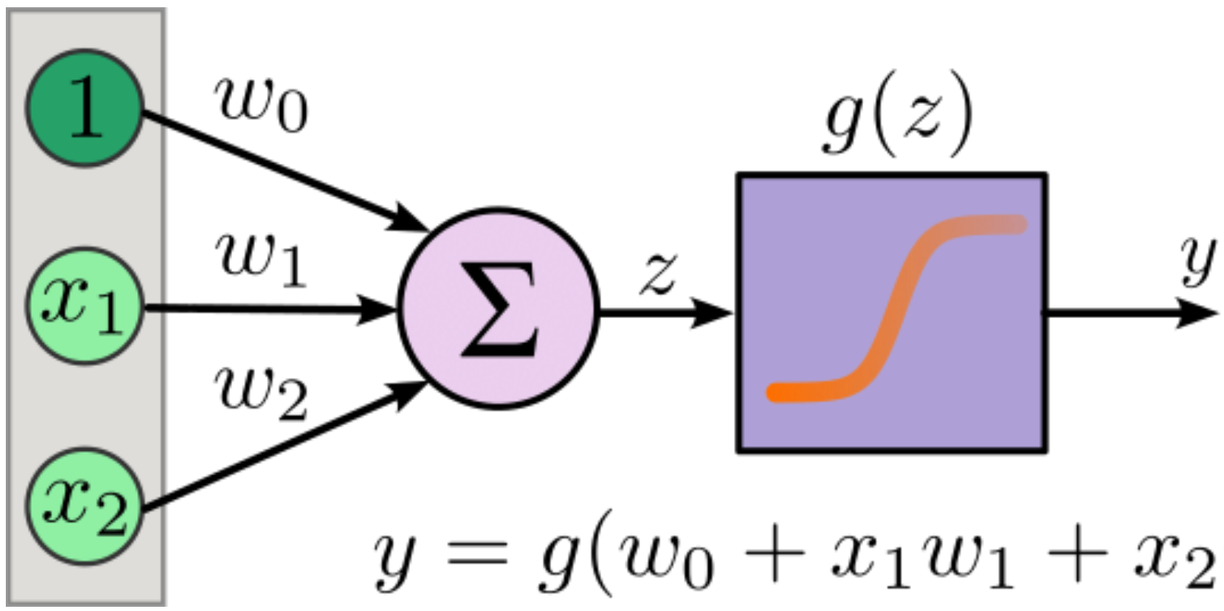
$x_1\ OR\ x_2$



$x_1\ AND\ x_2$

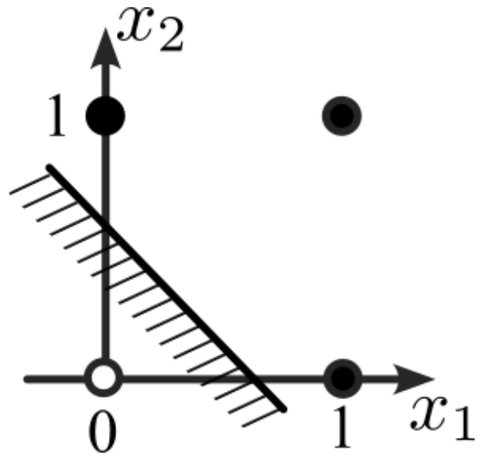
.

Ограничения линейных классификаторов

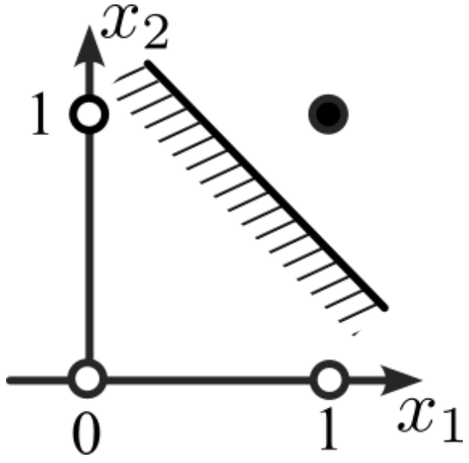


$$y = g(w_0 + x_1 w_1 + x_2 w_2)$$

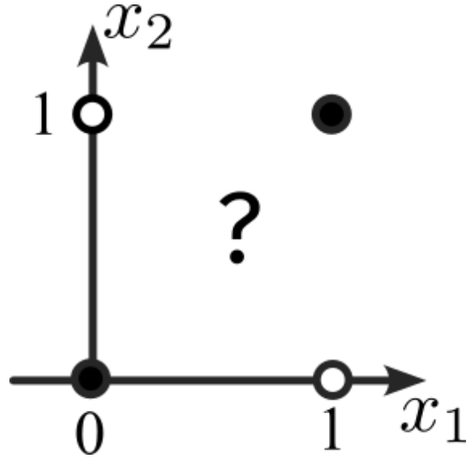
$$\begin{cases} y > \text{thr} \Rightarrow True \\ y < \text{thr} \Rightarrow False \end{cases}$$



$x_1\ OR\ x_2$



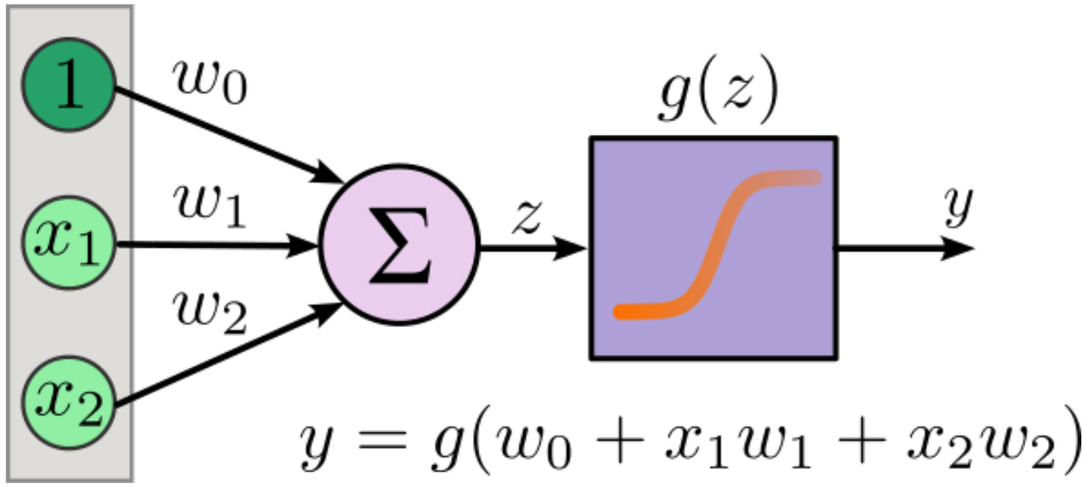
$x_1\ AND\ x_2$



$x_1\ XOR\ x_2$

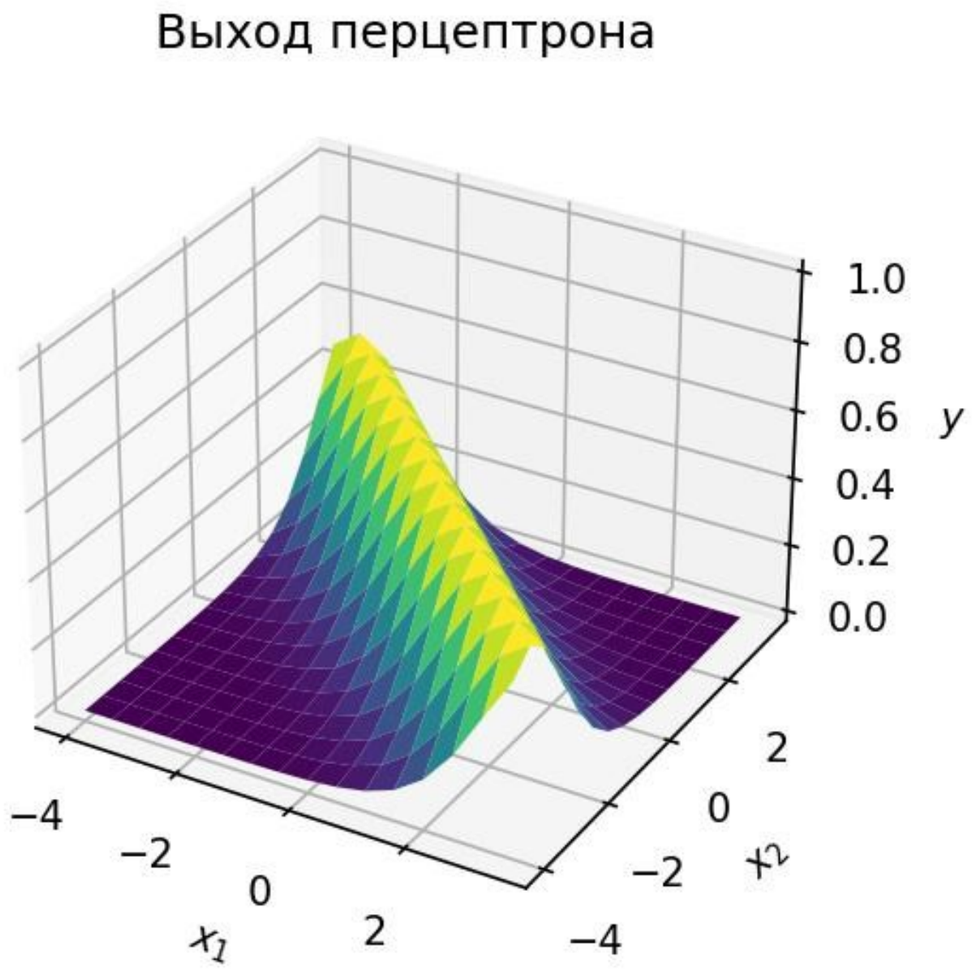
- Перцептрон может представить функции OR, AND, NOT, но не XOR.

Ограничения линейных классификаторов



Ограничения линейных классификаторов

Непрерывная функция для двухслойной сети



Построение нелинейных классификаторов

Нейронная сеть, которая имеет хотя бы один скрытый слой является универсальным аппроксиматором (может представить любую функцию).

Math. Control Signals Systems (1989) 2: 303–314

Mathematics of Control, Signals, and Systems

Approximation by Superpositions of a Sigmoidal Function*

G. Cybenko†

Abstract. In this paper we demonstrate that finite linear combinations of compositions of a fixed, univariate function and a set of affine functionals can uniformly approximate any continuous function of n real variables with support in the unit hypercube; only mild conditions are imposed on the univariate function. Our results settle an open question about representability in the class of single hidden layer neural networks. In particular, we show that arbitrary decision regions can be arbitrarily well approximated by continuous feedforward neural networks with only a single internal, hidden layer and any continuous sigmoidal nonlinearity. The paper discusses approximation properties of other possible types of nonlinearities that might be implemented by artificial neural networks.

Key words. Neural networks, Approximation, Completeness.

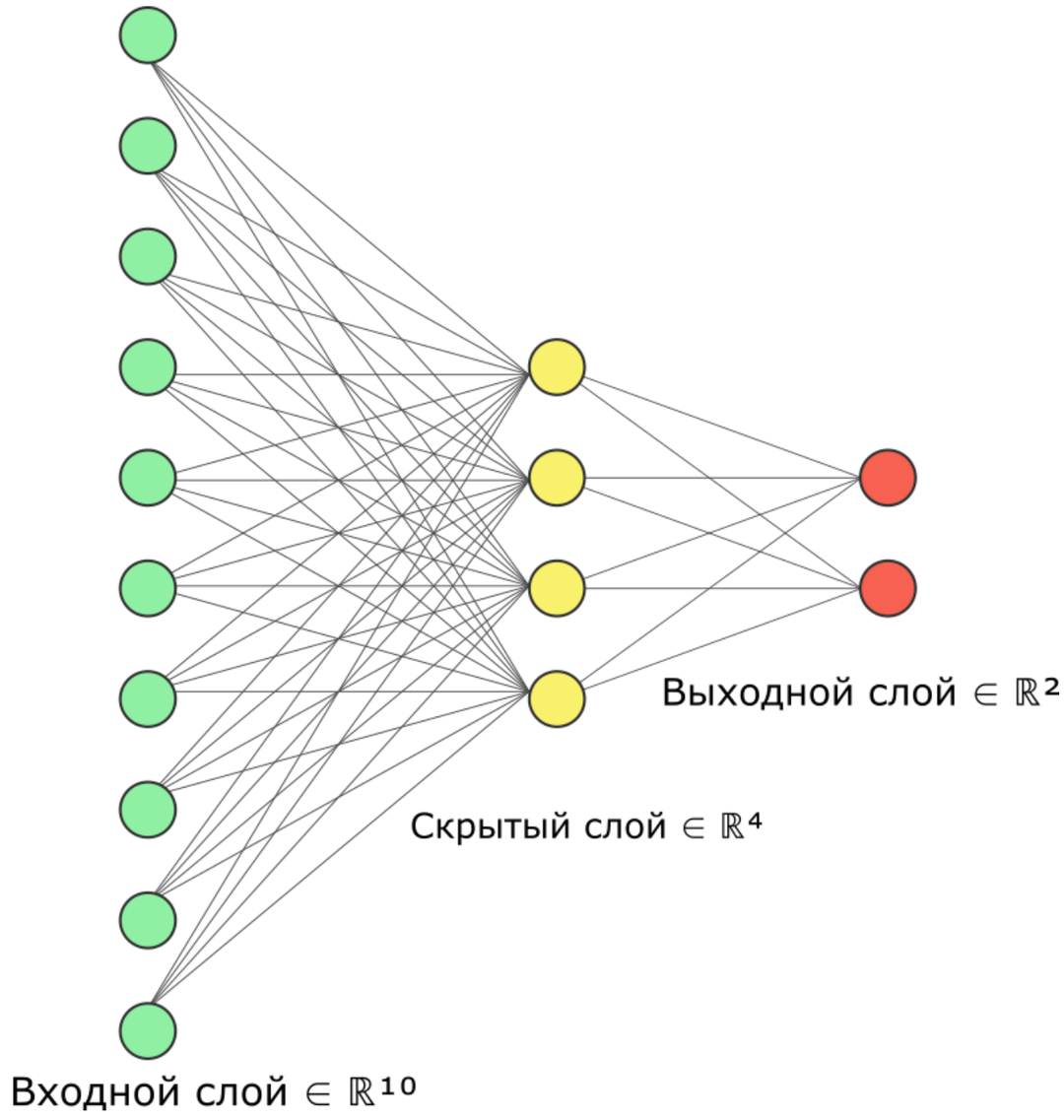
1. Introduction

A number of diverse application areas are concerned with the representation of general functions of an n-dimensional real variable, $x \in \mathbb{R}^n$, by finite linear combinations of the form

$$\sum_{j=1}^{N} \alpha_j \sigma(y_j^T x + \theta_j), \tag{1}$$

Нейронные сети прямого распространения (НСПР)

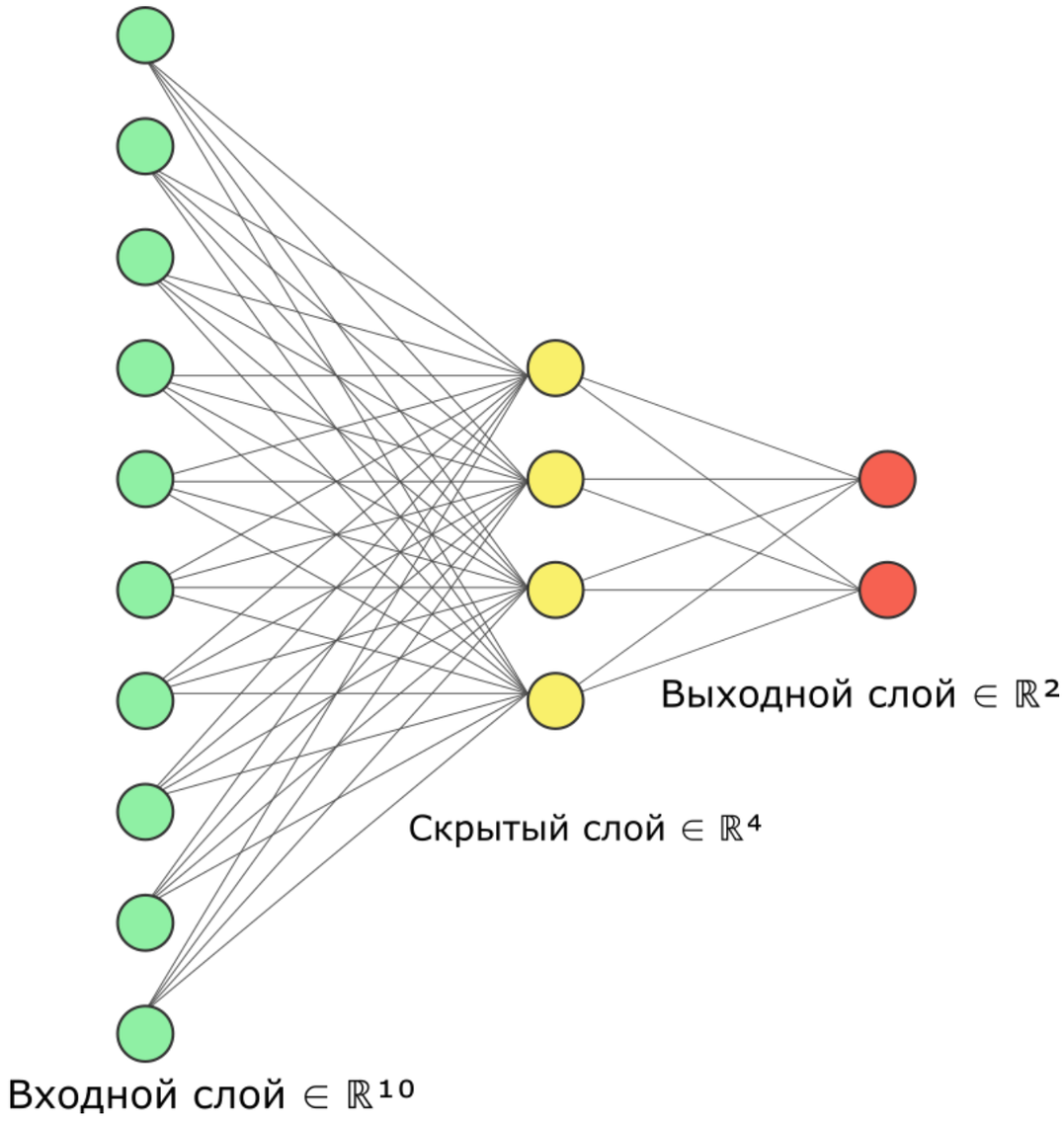
- Это самый распространенный тип нейронной сети в практических приложениях.
  - Первый слой – это входные данные, а последний слой - выходные.



DNN: 10-4-2

Нейронные сети прямого распространения (НСПР)

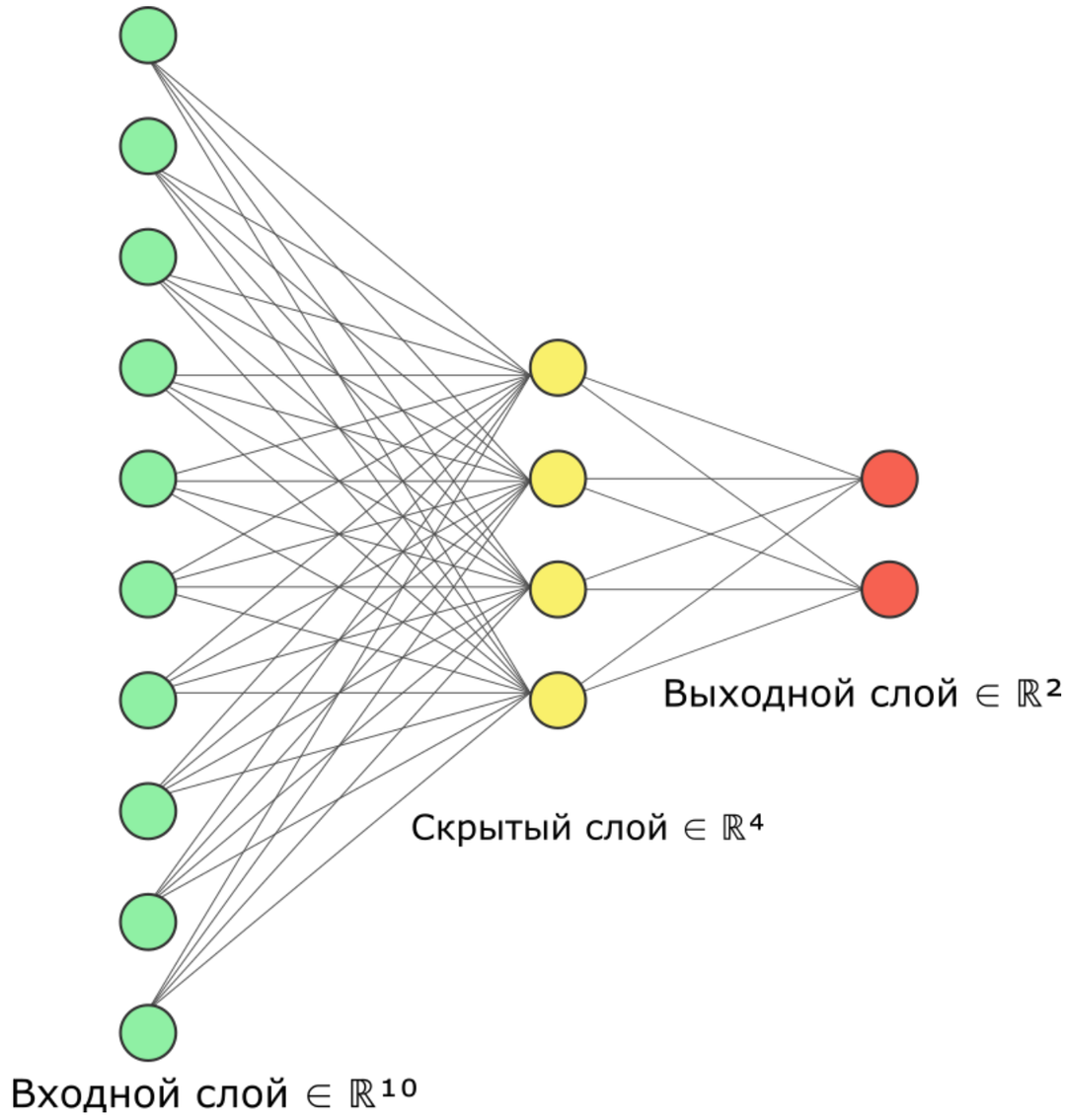
- Это самый распространенный тип нейронной сети в практических приложениях.
  - Первый слой – это входные данные, а последний слой - выходные.
  - Если существует более одного скрытого слоя, мы называем нейросеть “глубокой”.



DNN: 10-4-2

Нейронные сети прямого распространения (НСПР)

- Это самый распространенный тип нейронной сети в практических приложениях.
  - Первый слой – это входные данные, а последний слой - выходные.
  - Если существует более одного скрытого слоя, мы называем нейросеть “глубокой”.
- На рисунке показана НС 10-4-2, так как она имеет 10 входных, 4 скрытых и 2 выходных нейрона.

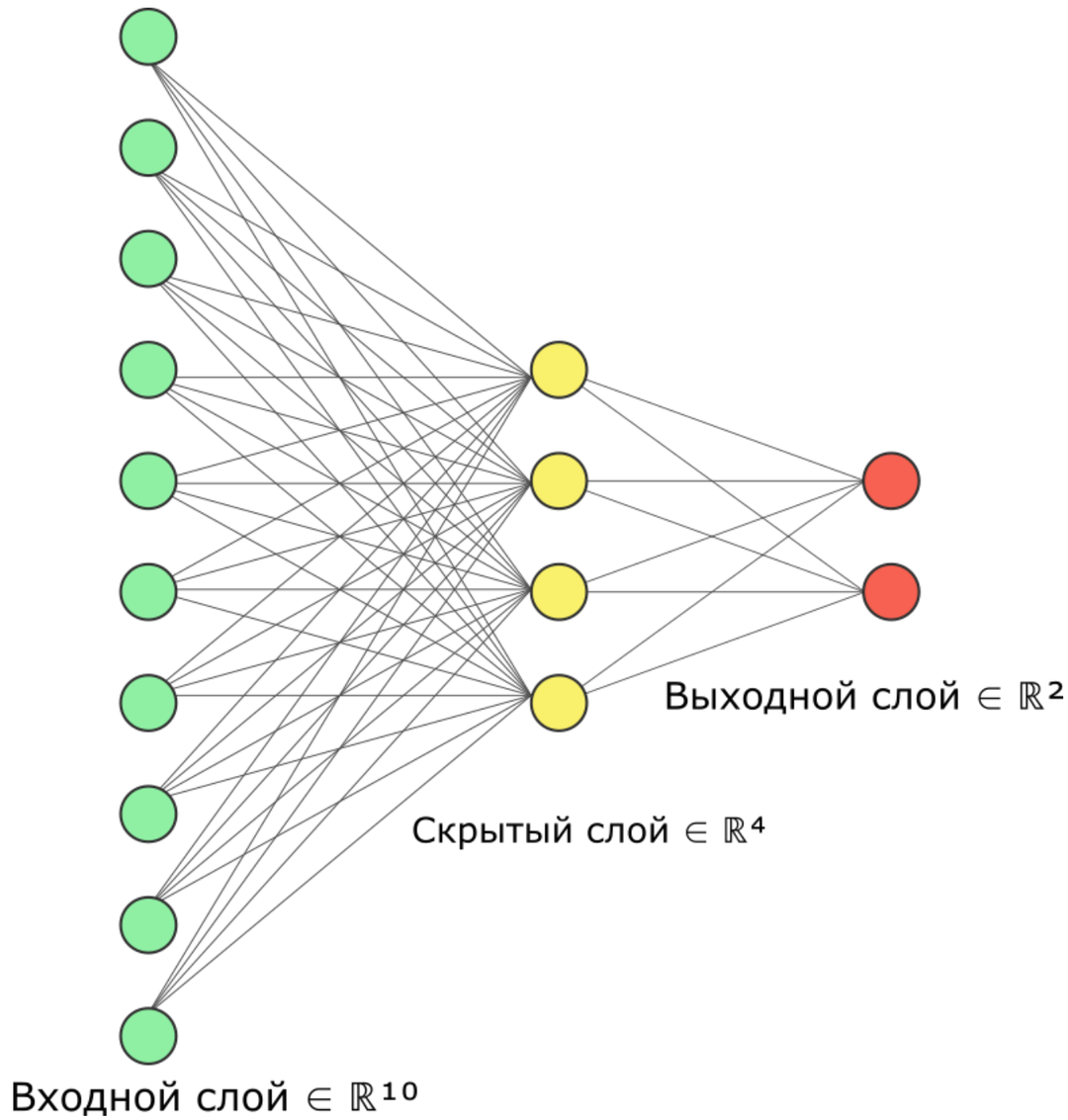


DNN: 10-4-2

Нейронные сети прямого распространения (НСПР)

- Это самый распространенный тип нейронной сети в практических приложениях.
  - Первый слой – это входные данные, а последний слой - выходные.
  - Если существует более одного скрытого слоя, мы называем нейросеть “глубокой”.
- На рисунке показана НС 10-4-2, так как она имеет 10 входных, 4 скрытых и 2 выходных нейрона.
- Данная НС называется ***полносвязной***.

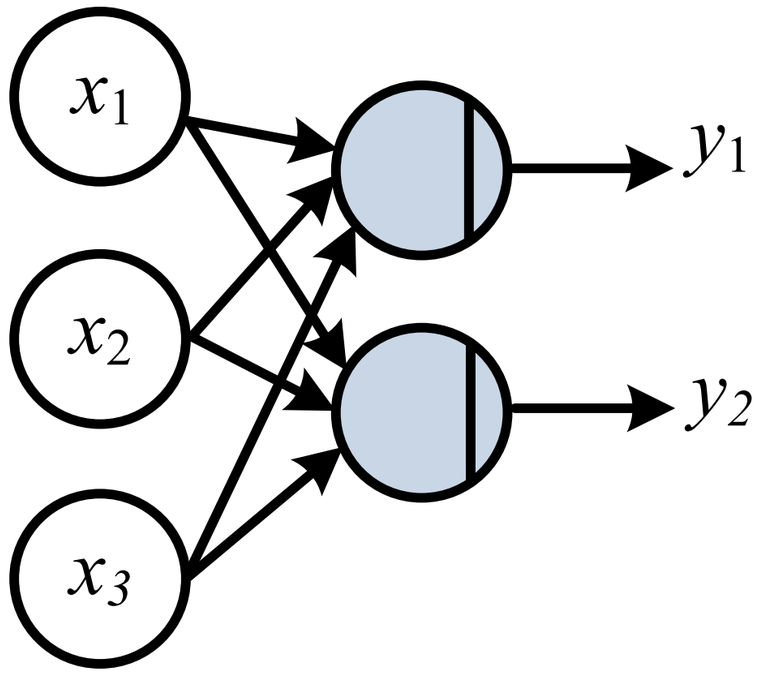
$$f_{NN}: \mathbb{R}^N \rightarrow \mathbb{R}^M$$



DNN: 10-4-2

Однослойная нейронная сеть

Рассмотрим пример: $d = 3$ (входной слой), $M = 2$ (выходной слой)

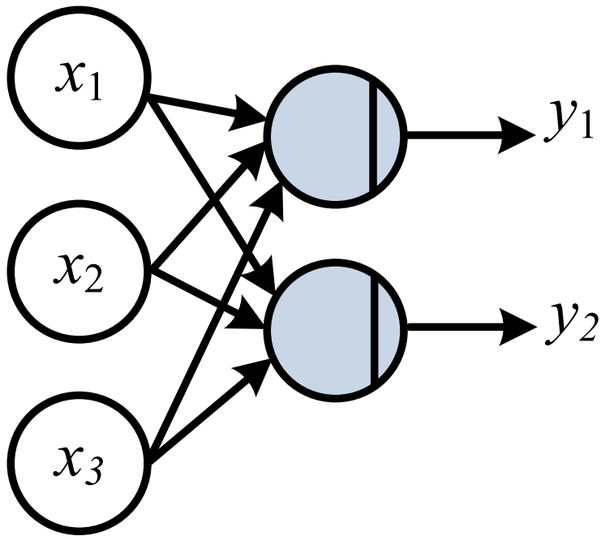


Нейросеть описывается выражением

$$y_i(\mathbf{x}) = g(\mathbf{w}_i\mathbf{x}), \qquad \text{где } x = [1 \quad x_1 \quad x_2 \quad x_3]^T,$$
$$w_i = [b_i \quad w_{i,1} \quad w_{i,2} \quad w_{i,3}]$$

Однослойная нейронная сеть

Рассмотрим пример: $d = 3$ (входной слой), $M = 2$ (выходной слой)

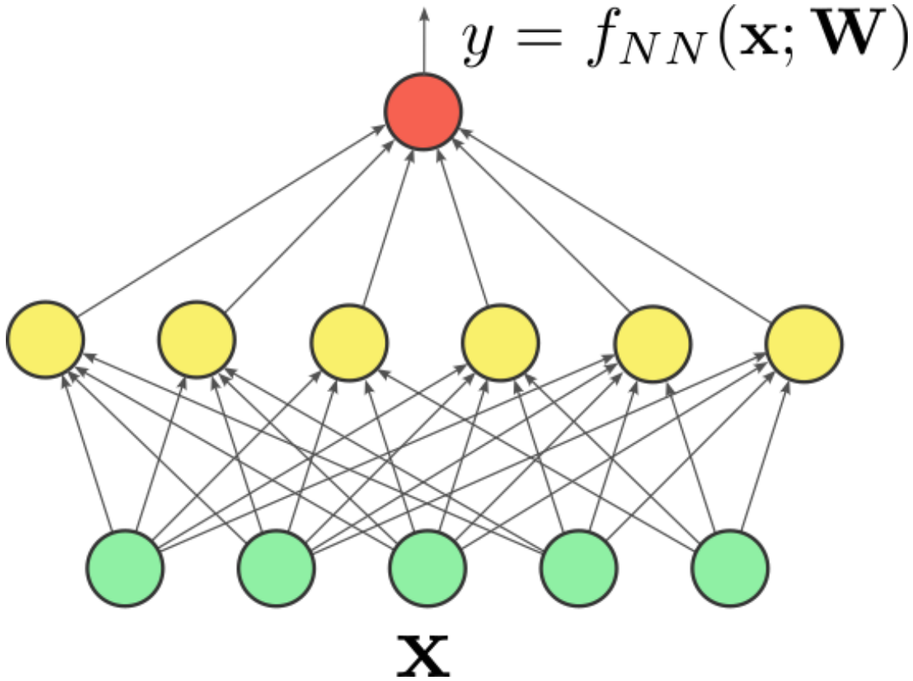


$$y_i(\mathbf{x}) = g(\mathbf{w}_i\mathbf{x}), \qquad \text{где } x = [1 \quad x_1 \quad x_2 \quad x_3]^T.$$

Матричная запись

$$\mathbf{y}(\mathbf{x}) = g(\mathbf{W}\mathbf{x}) \Leftrightarrow \begin{bmatrix} y_1 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} g(\mathbf{w}_1\mathbf{x}) \\ \vdots \\ g(\mathbf{w}_M\mathbf{x}) \end{bmatrix}, \qquad \mathbf{W} = \begin{bmatrix} b_1 & w_{1,1} & \cdots & w_{1,d} \\ \vdots & \vdots & \ddots & \vdots \\ b_M & w_{M,1} & \cdots & w_{M,d} \end{bmatrix}$$

Что мы обучаем? Параметры сети



Сеть – это функция $f_{NN}()$ с параметрами $\mathbf{W}$, для которых необходимо найти соответствующие значения, которые обеспечат нужное поведение сети.

- Архитектура сети обычно является фиксированной

- Параметры сети – это веса и смещения (веса ассоциируются со стрелками на графе сети).

- ***Обучение сети*** – нахождение таких значений параметров, чтобы нейронная сеть вычисляла заданную функцию.

Многослойный перцептрон (MLP)

- Пусть полносвязный слой НС описывается выражением:

$$\mathbf{y}(\mathbf{x}) = g(\mathbf{W}\mathbf{x}) = \mathrm{FC}(\mathbf{x}).$$

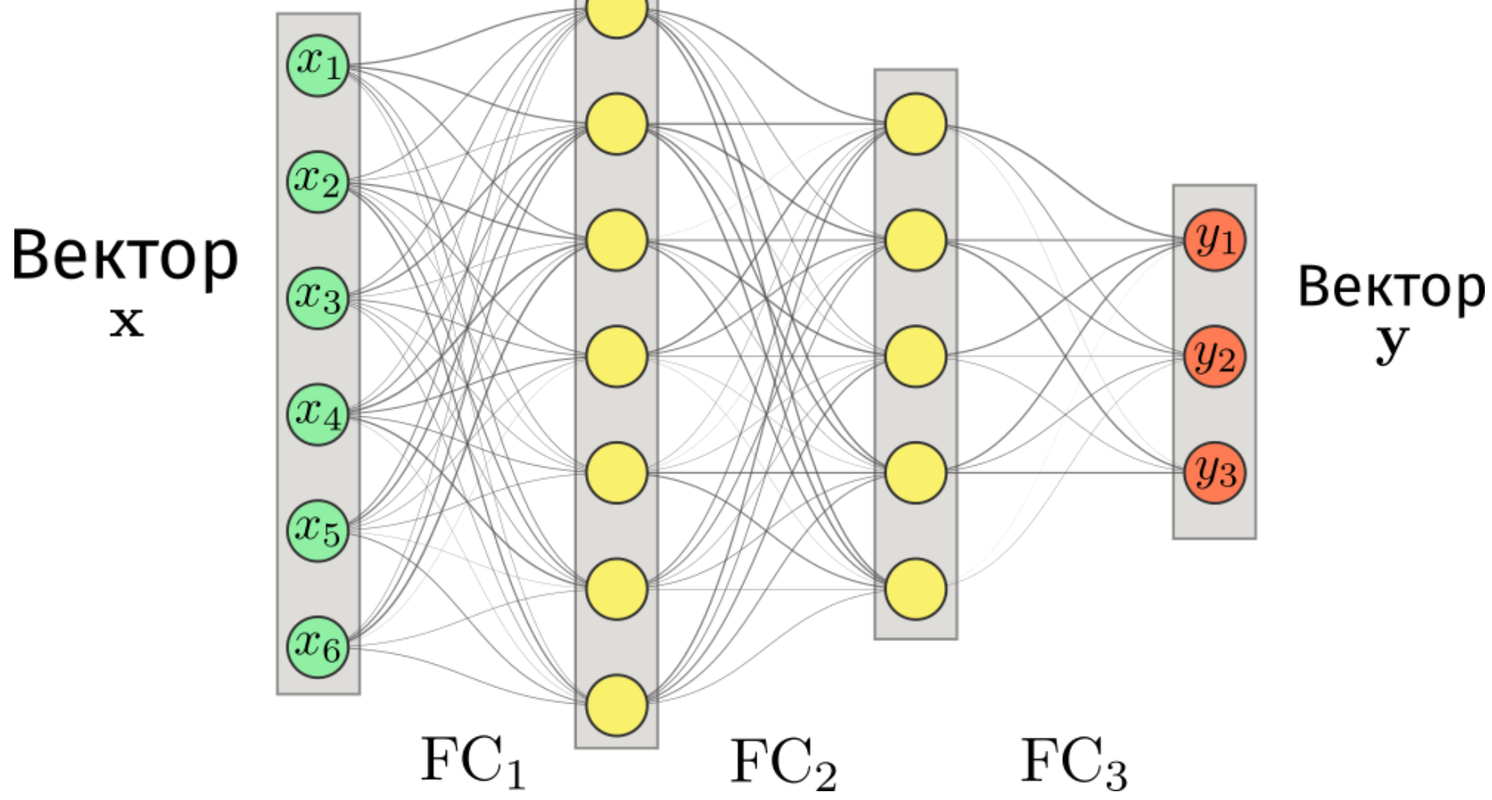
Многослойный перцептрон (MLP)

- Пусть полносвязный слой НС описывается выражением:

$$\mathbf{y}(\mathbf{x}) = g(\mathbf{W}\mathbf{x}) = \mathrm{FC}(\mathbf{x}).$$

- Многослойный перцептрон описывается выражением:

$$\mathbf{y} = f_{NN}(\mathbf{x}) = \mathrm{FC}_3\left(\mathrm{FC}_2\left(\mathrm{FC}_1(\mathbf{x})\right)\right).$$

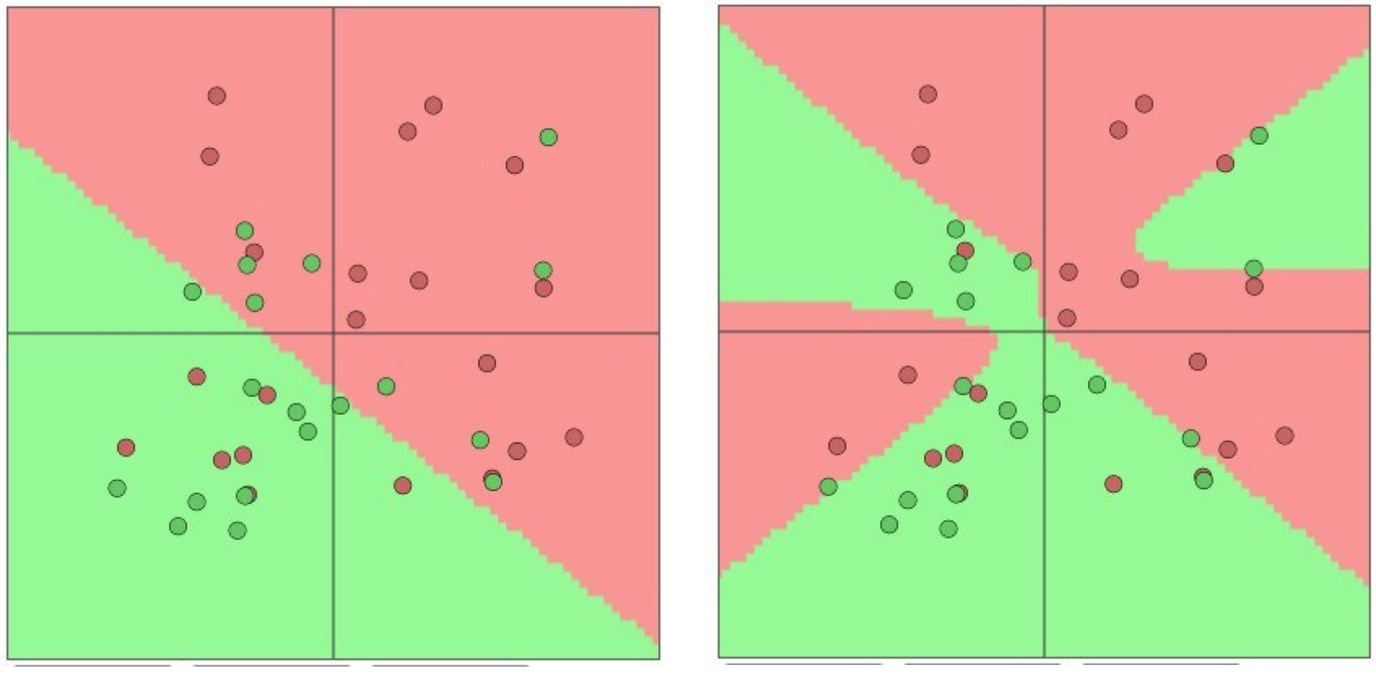


Почему нужна нелинейность между слоями?

• Без нелинейности, глубокая нейронная сеть работает, как обычное линейное преобразование:

$$\mathbf{W}_2(\mathbf{W}_1\mathbf{x}) = \mathbf{W}_2\mathbf{W}_1\mathbf{x} = \mathbf{W}\mathbf{x}$$

• При добавлении нелинейности нейронная сеть может аппроксимировать более сложные функции



Обучение нейронной сети

- Для обучения нейронной сети необходимо иметь тренировочный набор $\mathcal{D} = \{(\mathbf{x}^n, c^n) \mid n = 1, \ldots, N\}$, где c^n определяет метку класса образца $\mathbf{x}^n$)

Обучение нейронной сети

- Для обучения нейронной сети необходимо иметь тренировочный набор $\mathcal{D} = \{(\mathbf{x}^n, c^n) \mid n = 1, \ldots, N\}$, где c^n определяет метку класса образца $\mathbf{x}^n$)
- Целевое значение i-го выхода нейронной сети для n-го образца:

$$t_i^n = \begin{cases} 1, & i = c^n, \\ 0, & \text{иначе.} \end{cases}$$

Задача обучения состоит в подстройке весов $\mathbf{W}$ таким образом, чтобы ***функция потерь*** на тренировочном наборе была минимальной.

Обучение нейронной сети

- Для обучения нейронной сети необходимо иметь тренировочный набор $\mathcal{D} = \{(\mathbf{x}^n, c^n) \mid n = 1, \ldots, N\}$, где c^n определяет метку класса образца $\mathbf{x}^n$)
- Целевое значение i-го выхода нейронной сети для n-го образца:

$$t_i^n = \begin{cases} 1, & i = c^n, \\ 0, & \text{иначе.} \end{cases}$$

Задача обучения состоит в подстройке весов $\mathbf{W}$ таким образом, чтобы ***функция потерь*** на тренировочном наборе была минимальной

Квадратичная функция потерь

$$E = \frac{1}{2} \sum_{n=1}^{N} \sum_{i=1}^{M} [y_i(\mathbf{x}^n) - t_i^n]^2$$

Кросс-энтропия

$$E = -\sum_{n=1}^{N} \sum_{i=1}^{M} t_i^n \log y_i(\mathbf{x}^n)$$

Обучение нейронной сети

- Для обучения нейронной сети необходимо иметь тренировочный набор $\mathcal{D} = \{(\mathbf{x}^n, c^n) \mid n = 1, \ldots, N\}$, где c^n определяет метку класса образца $\mathbf{x}^n$)
- Целевое значение i-го выхода нейронной сети для n-го образца:

$$t_i^n = \begin{cases} 1, & i = c^n, \\ 0, & \text{иначе.} \end{cases}$$

Задача обучения состоит в подстройке весов $\mathbf{W}$ таким образом, чтобы ***функция потерь*** на тренировочном наборе была минимальной

Квадратичная функция потерь

$$E = \frac{1}{2} \sum_{n=1}^{N} \sum_{i=1}^{M} [y_i(\mathbf{x}^n) - t_i^n]^2$$

Кросс-энтропия

$$E = -\sum_{n=1}^{N} \sum_{i=1}^{M} t_i^n \log y_i(\mathbf{x}^n)$$

- Градиентный спуск

$$w_{i,j} \leftarrow w_{i,j} - \eta \frac{\partial E}{\partial w_{i,j}}$$