Skillbox

Введение в нейронные сети

Skillbox

Введение в нейронные сети

Линейный классификатор. Повторение.

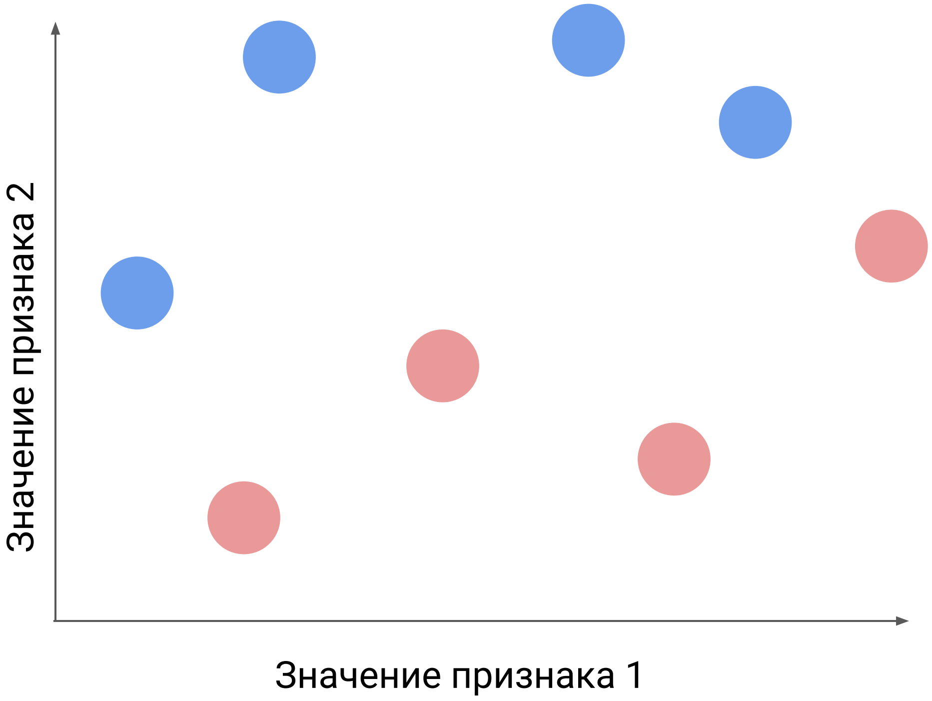
Задача классификации

Дано:

- Множество объектов, которые мы можем описать **признаками**

Skillbox

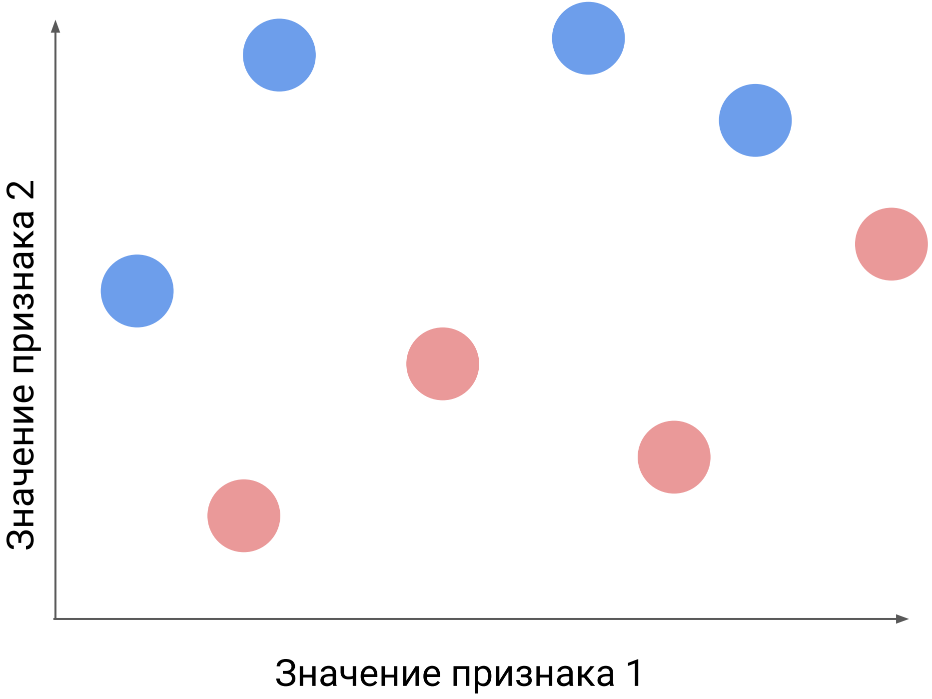
Задача классификации



Дано:

- Множество объектов, которые мы можем описать **признаками**
- Каждый объект имеет **класс**

Задача классификации



Дано:

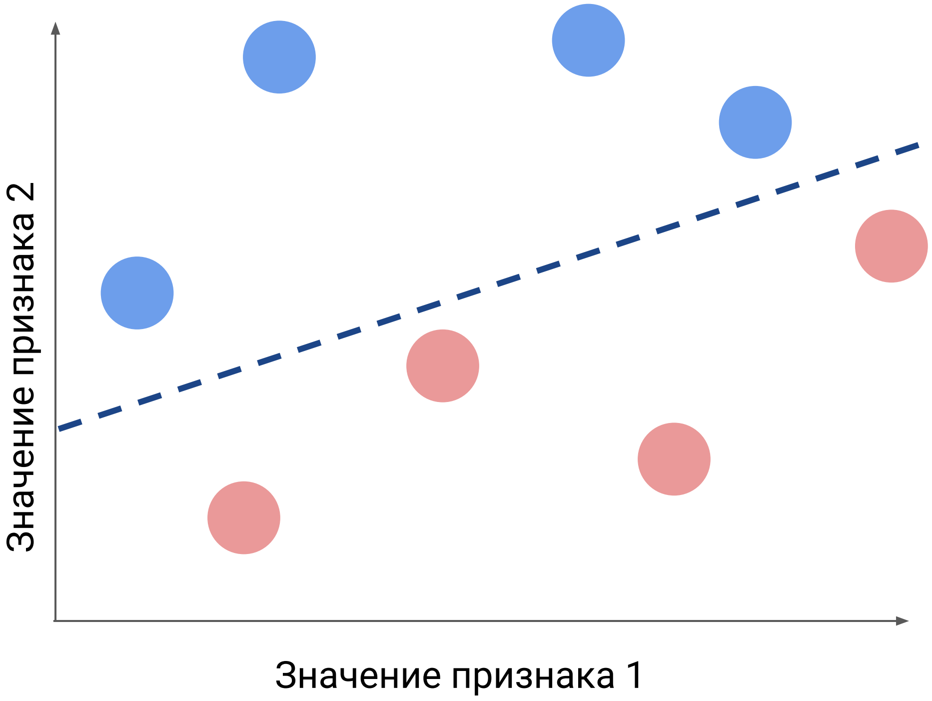
- Множество объектов, которые мы можем описать **признаками**
- Каждый объект имеет **класс**

Найти:

- **Алгоритм**, который правильно предсказывает класс для новых объектов

Skillbox

Задача классификации



Дано:

- Множество объектов, которые мы можем описать **признаками**
- Каждый объект имеет **класс**

Найти:

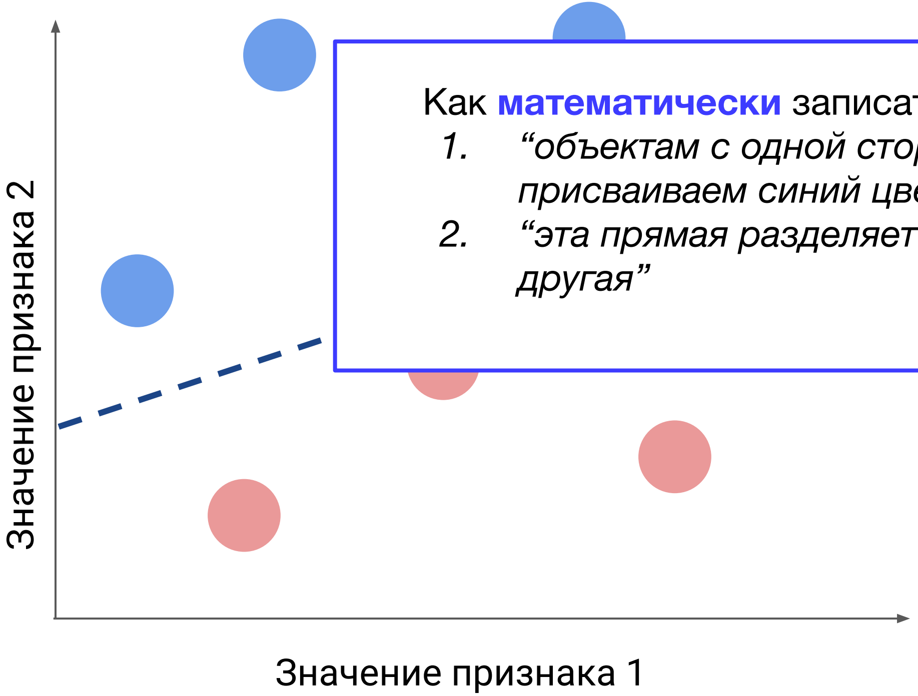
- **Алгоритм**, который правильно предсказывает класс для новых объектов

Для примера на слайде таким алгоритмом может быть:

- Все, что выше прямой -- синий класс
- Что ниже -- красный

Прямая -- разделяющая граница

Задача классификации



Как **математически** записать:

1. *“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”*
2. *“эта прямая разделяет объекты лучше, чем другая”*

Дано:

- Множество объектов, которые мы
- ... **признаками**
- ... имеет **класс**

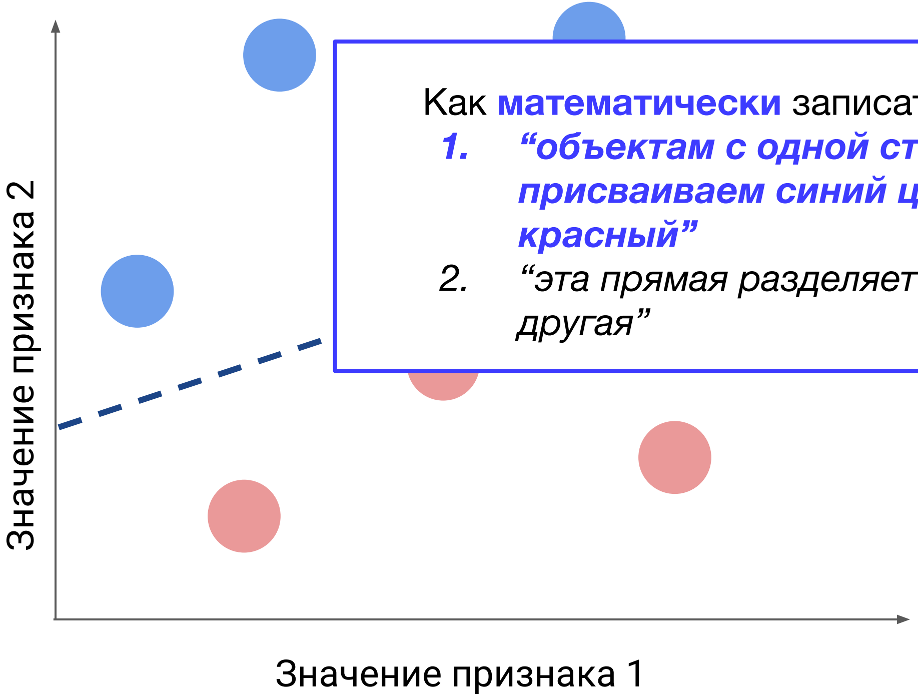
...рый правильно
...класс для новых

Для примера на слайде таким алгоритмом может быть:

- Все, что выше прямой -- синий класс
- Что ниже -- красный

Прямая -- разделяющая граница

Задача классификации



Как **математически** записать:

1. ***“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”***
2. *“эта прямая разделяет объекты лучше, чем другая”*

Дано:

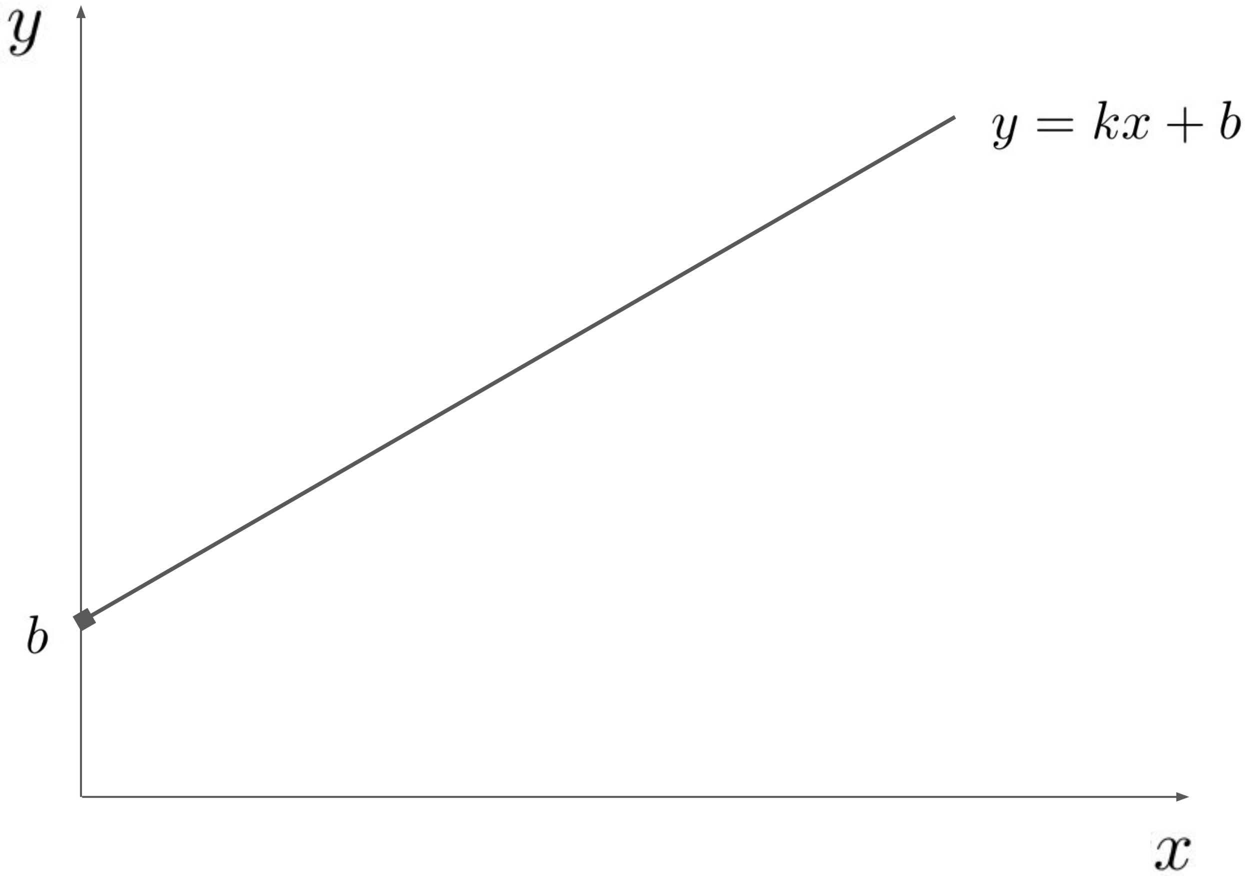
- Множество объектов, которые мы
признаками
имеет **класс**

рый правильно
класс для новых

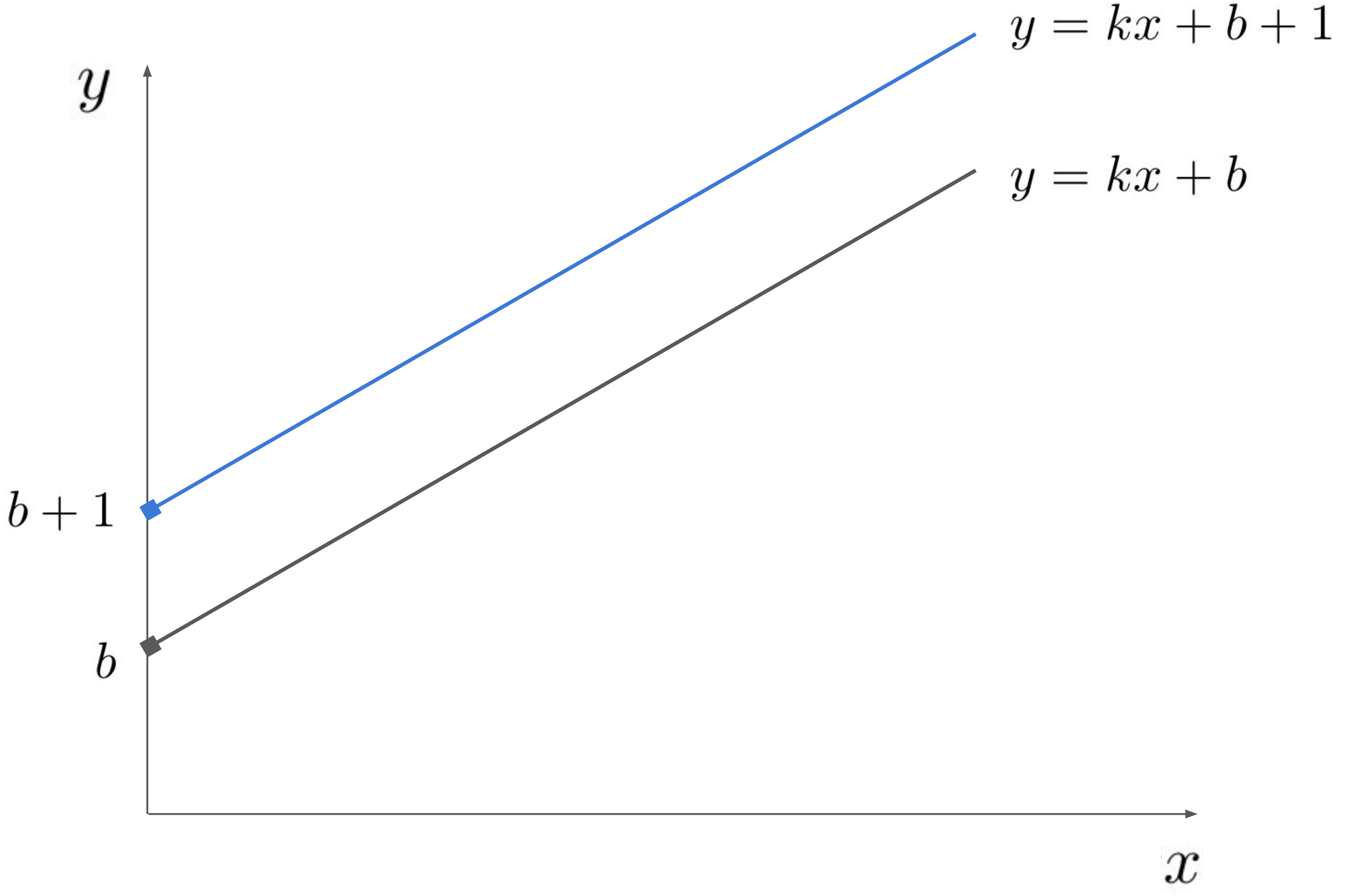
Для примера на слайде таким алгоритмом может быть:

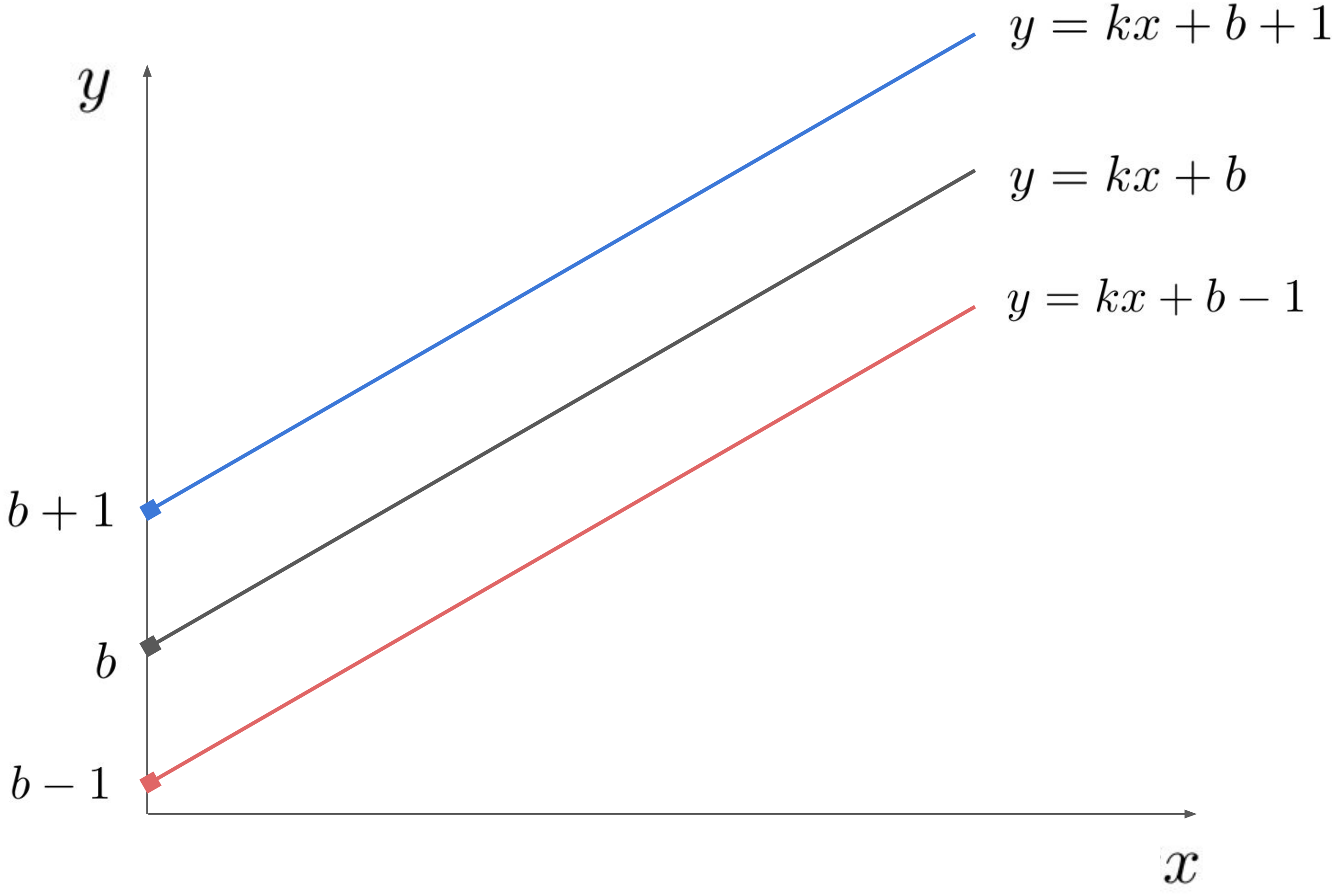
- Все, что выше прямой -- синий класс
- Что ниже -- красный

Прямая -- разделяющая граница

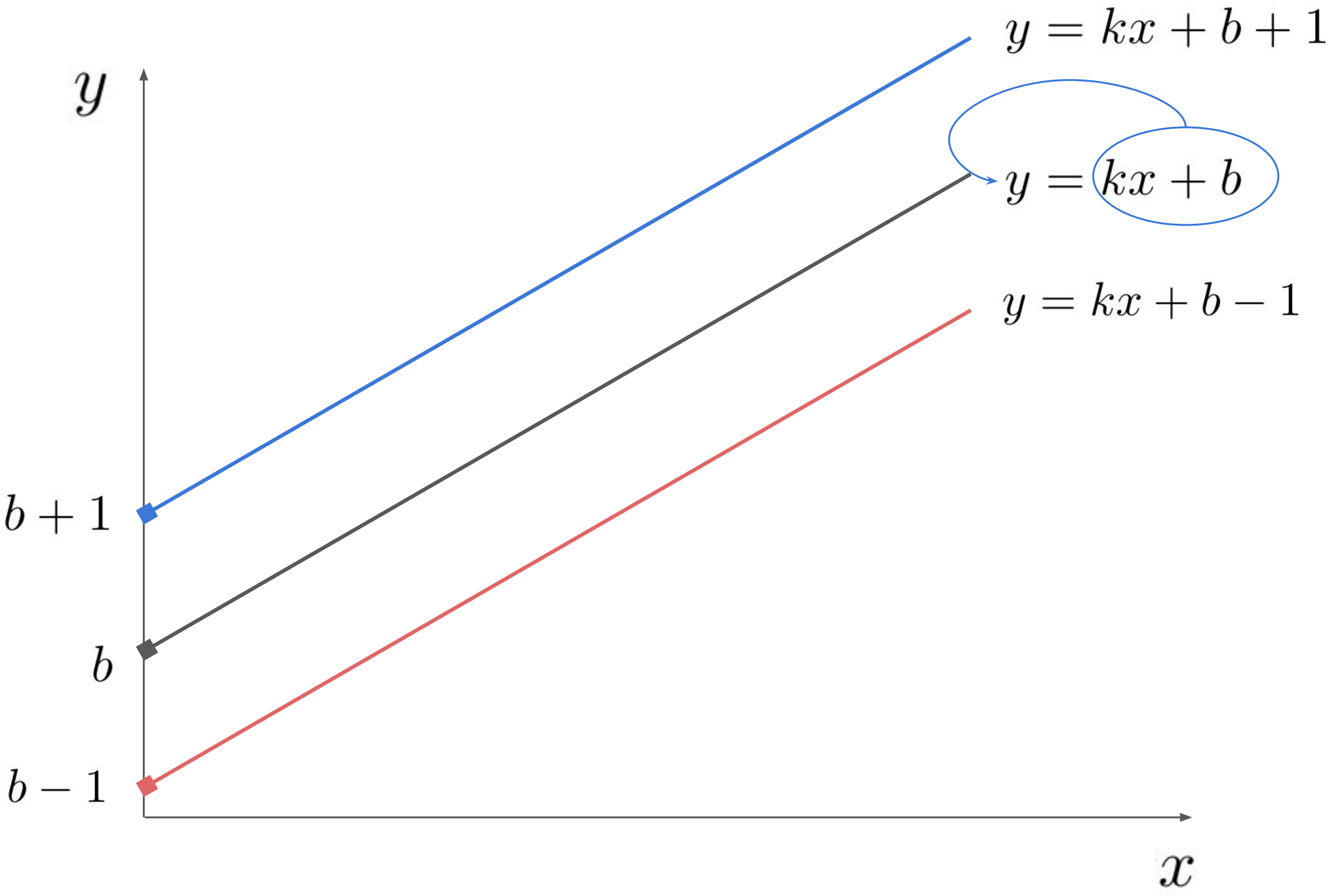

y
$y = kx + b$
b
x

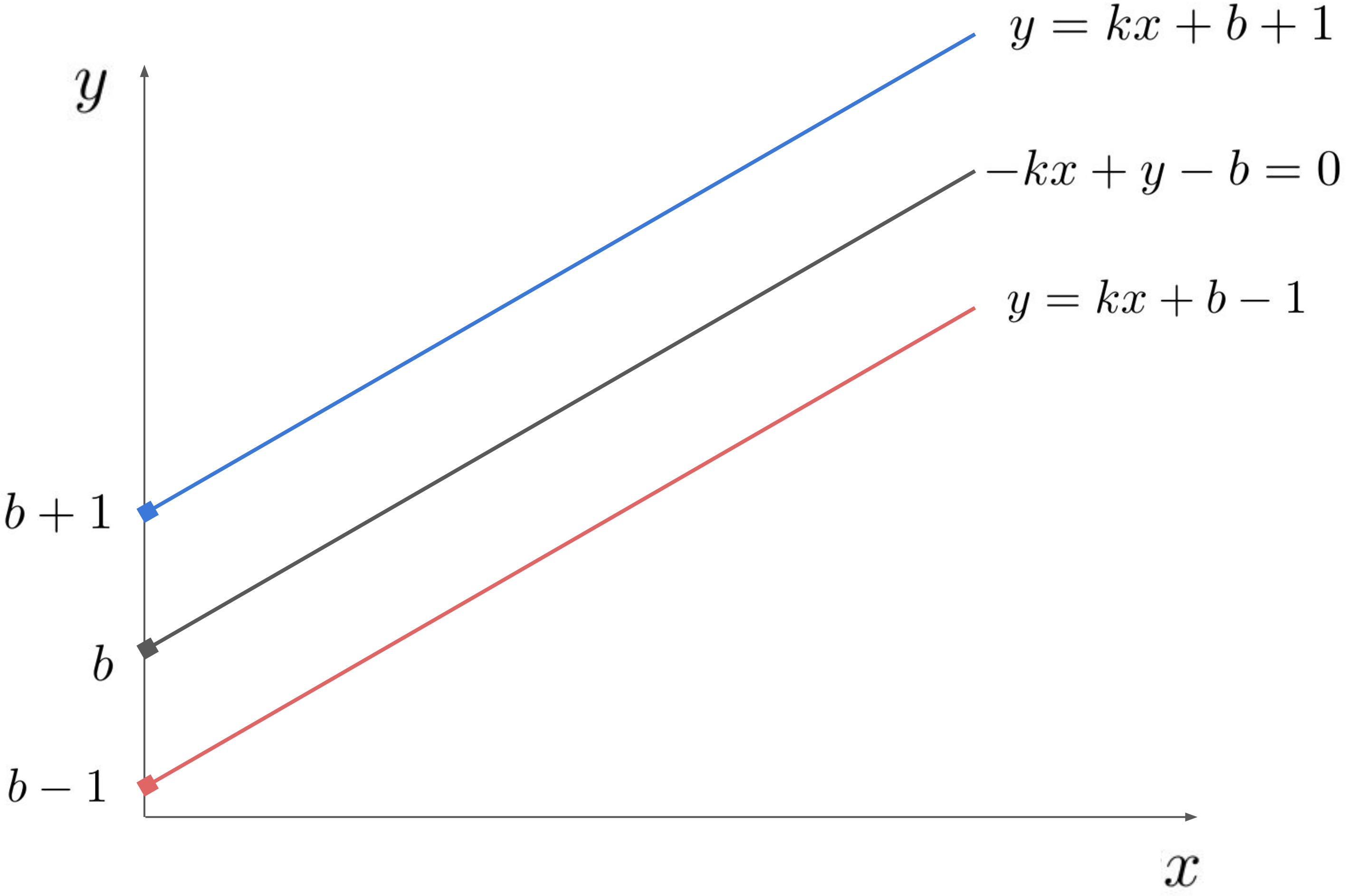
Skillbox



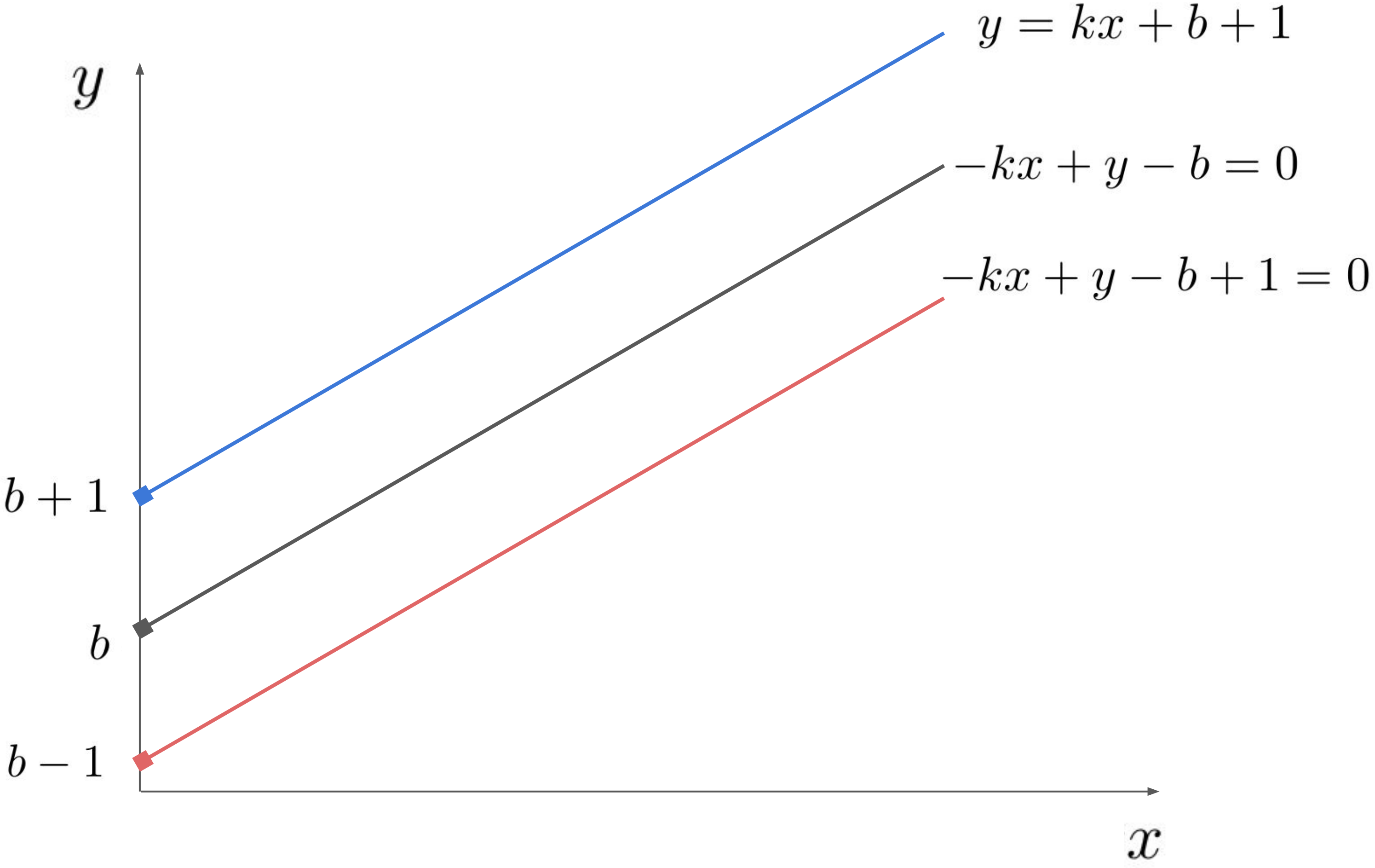


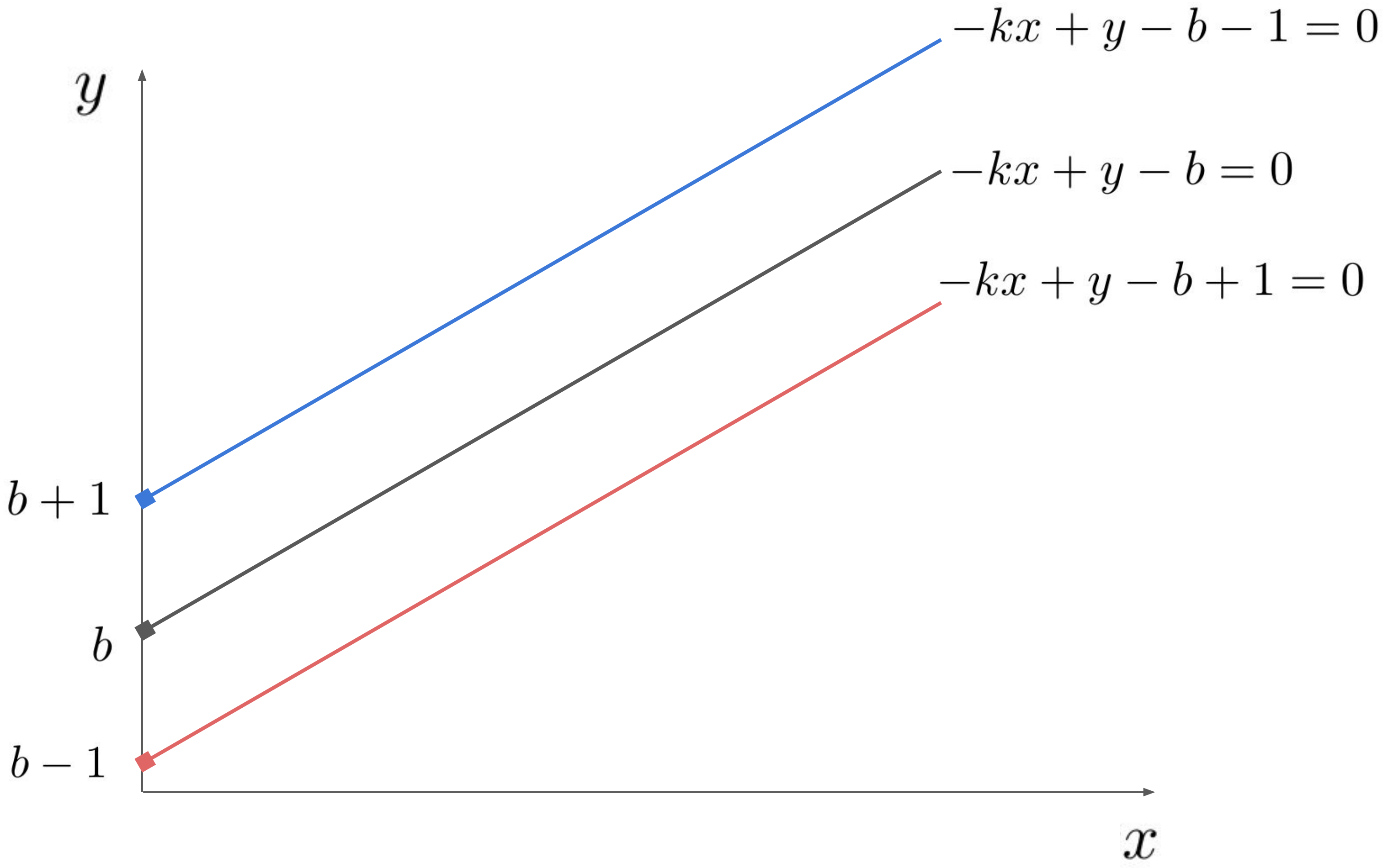
$y = kx + b + 1$
$y = kx + b$
$y = kx + b - 1$
y
$b + 1$
b
$b - 1$
x


$y = kx + b + 1$
$y = kx + b$
$y = kx + b - 1$
y
x
$b + 1$
b
$b - 1$

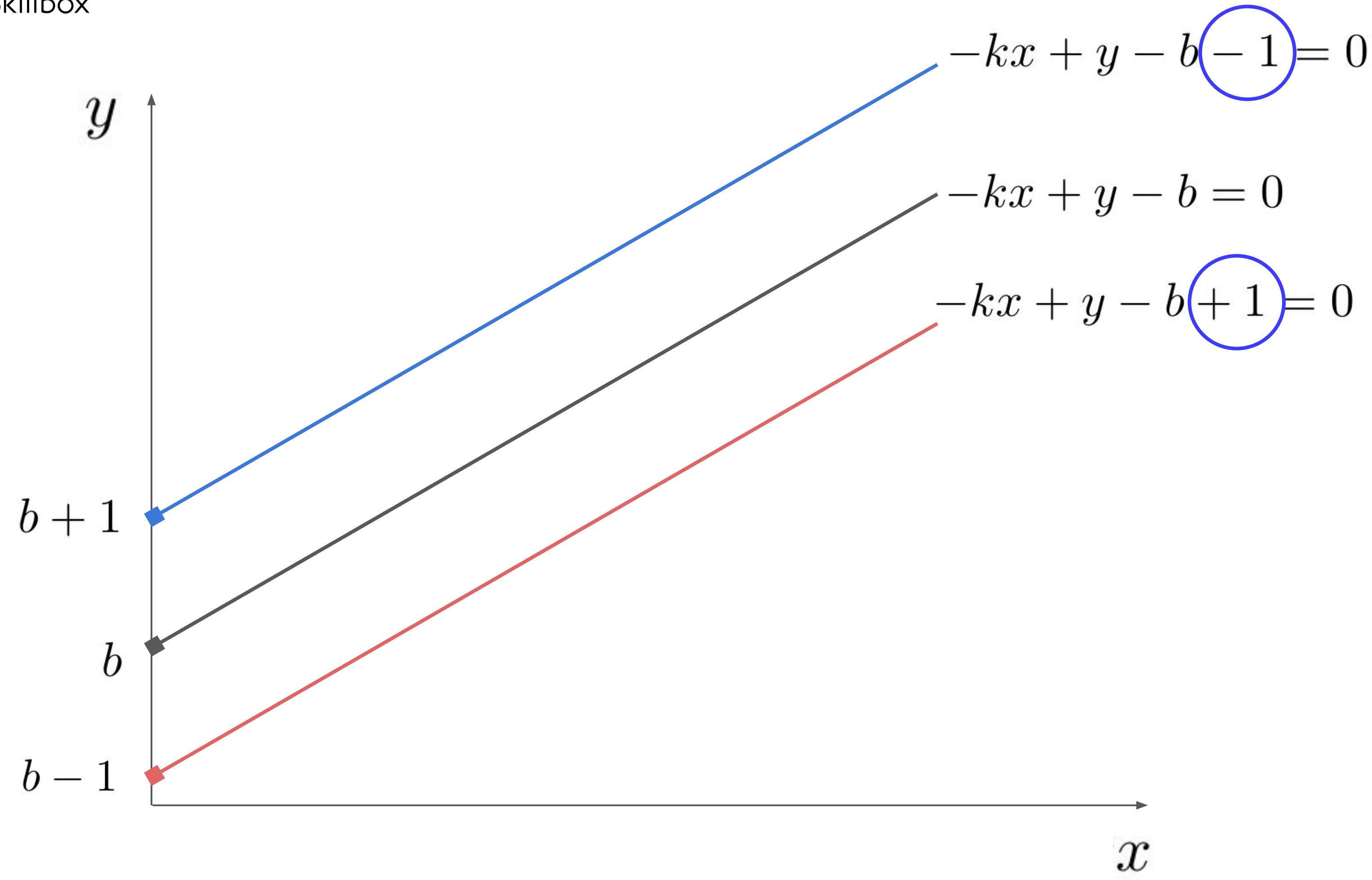


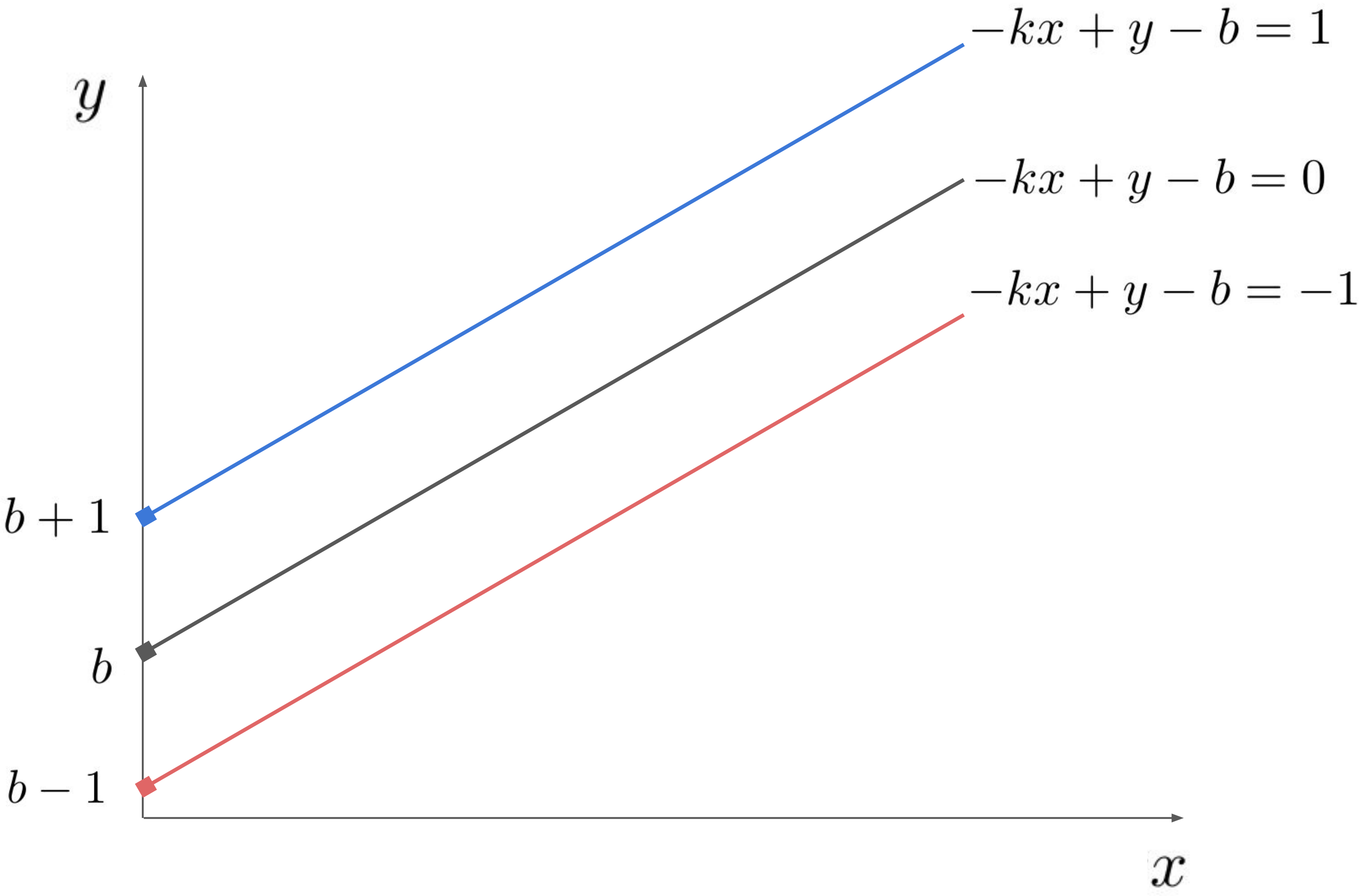
$y = kx + b + 1$
$-kx + y - b = 0$
$y = kx + b - 1$
y
$b + 1$
b
$b - 1$
x


$y = kx + b + 1$
$-kx + y - b = 0$
$-kx + y - b + 1 = 0$
y
$b + 1$
b
$b - 1$
x

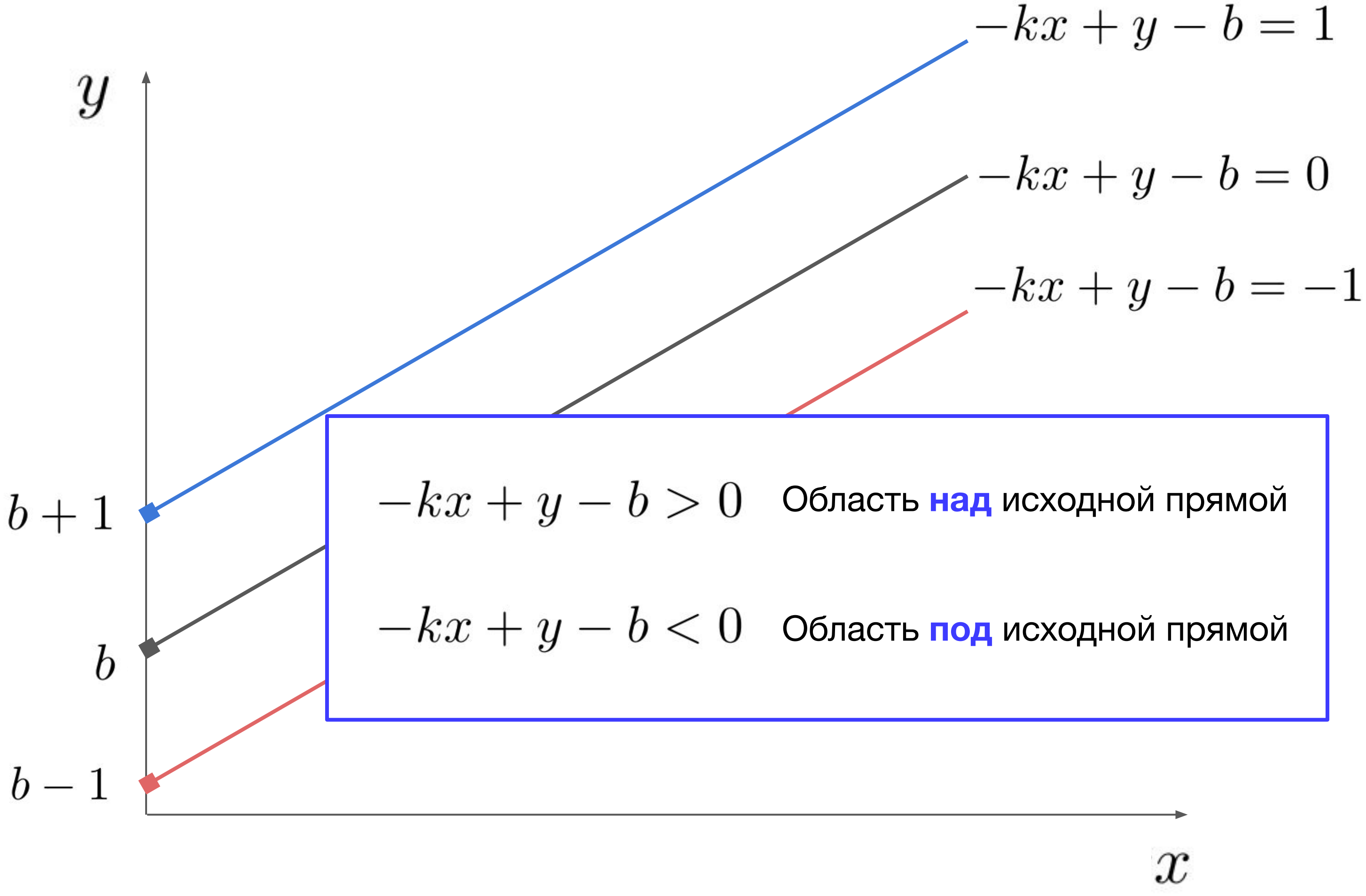

$-kx + y - b - 1 = 0$
$-kx + y - b = 0$
$-kx + y - b + 1 = 0$
y
$b + 1$
b
$b - 1$
x

Skillbox



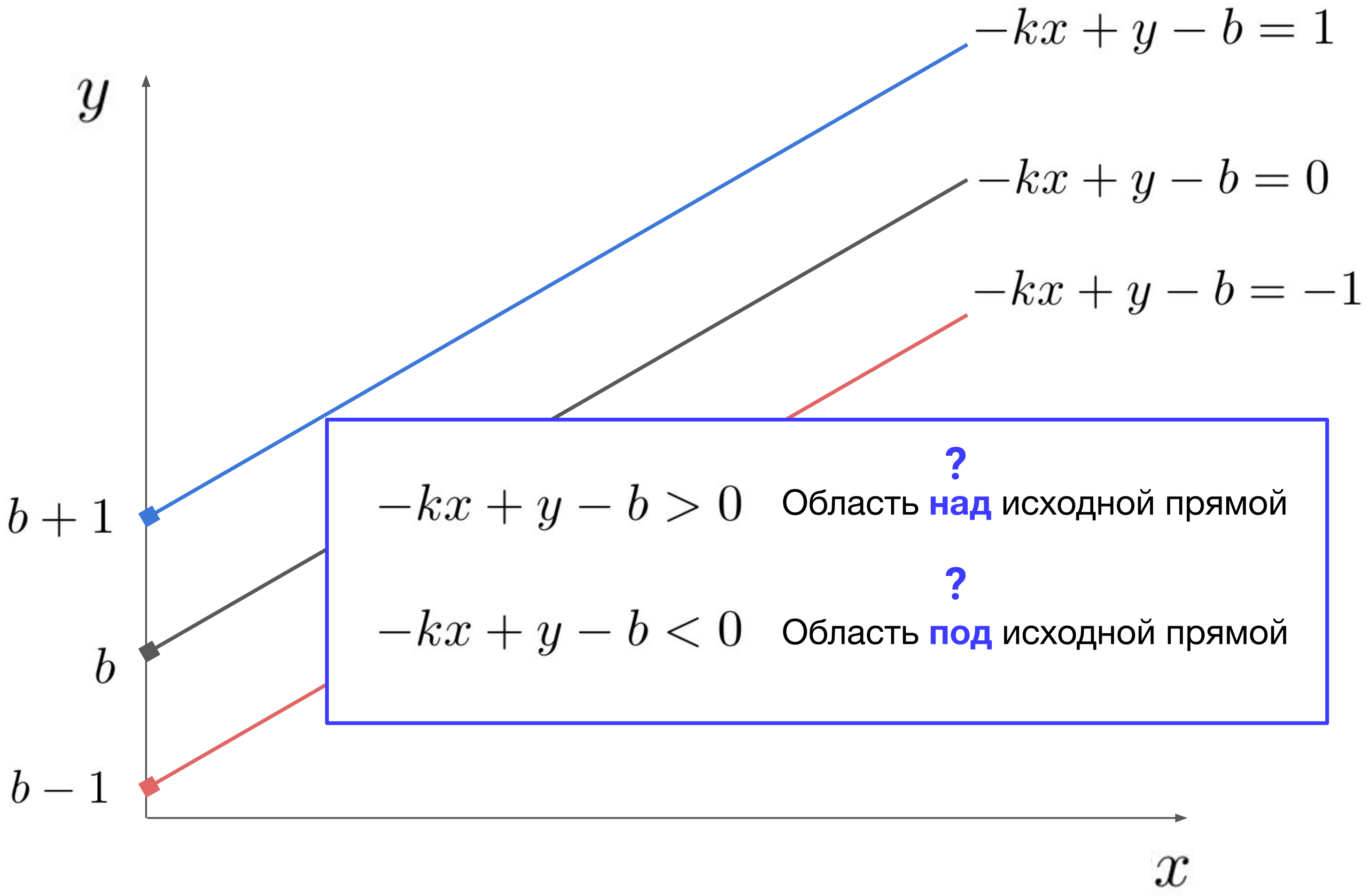


$-kx + y - b = 1$
y
$-kx + y - b = 0$
$-kx + y - b = -1$
$b + 1$
b
$b - 1$
x



$-kx + y - b > 0$ Область **над** исходной прямой

$-kx + y - b < 0$ Область **под** исходной прямой

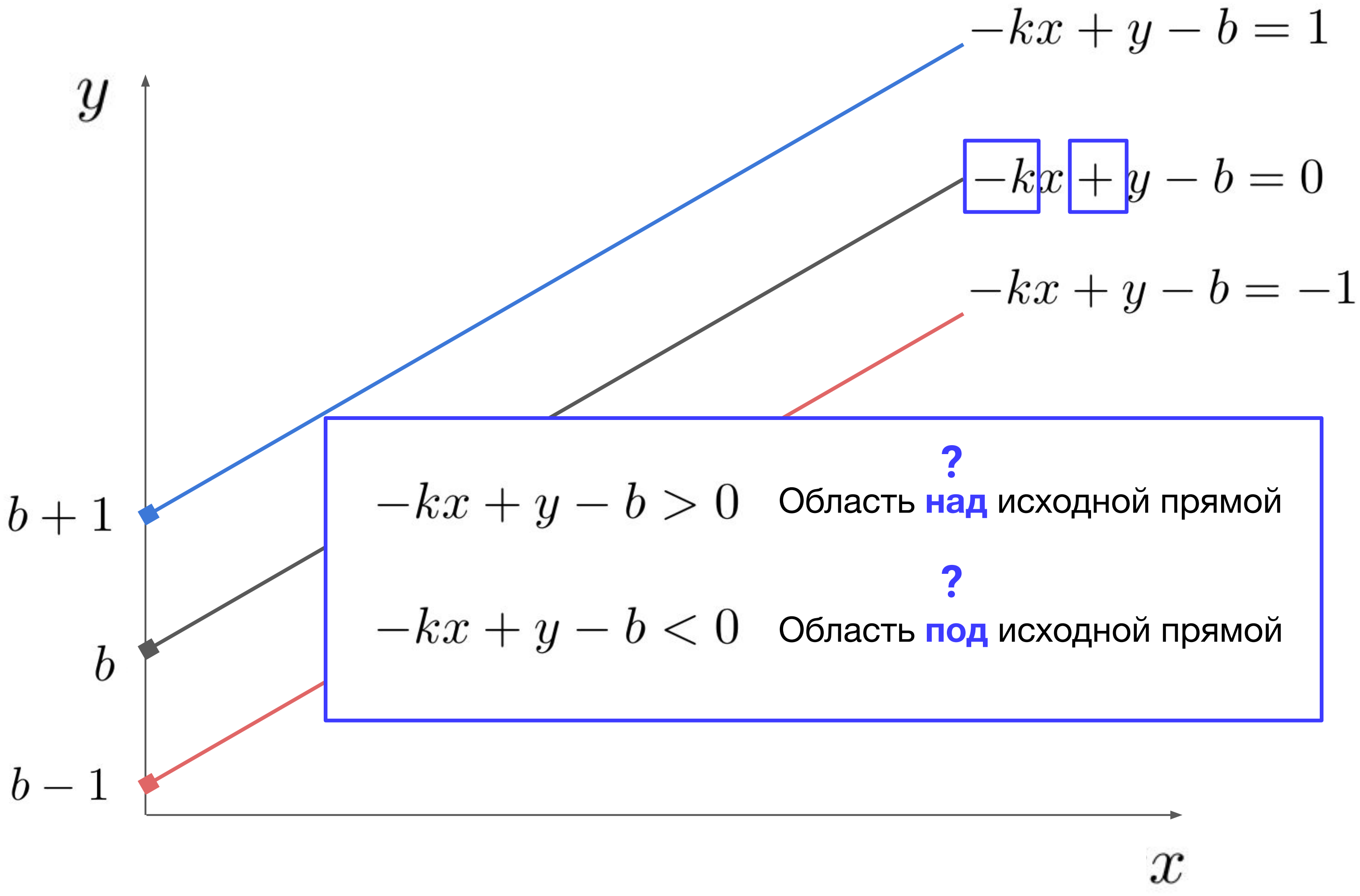


?

$-kx + y - b > 0$ Область **над** исходной прямой

?

$-kx + y - b < 0$ Область **под** исходной прямой

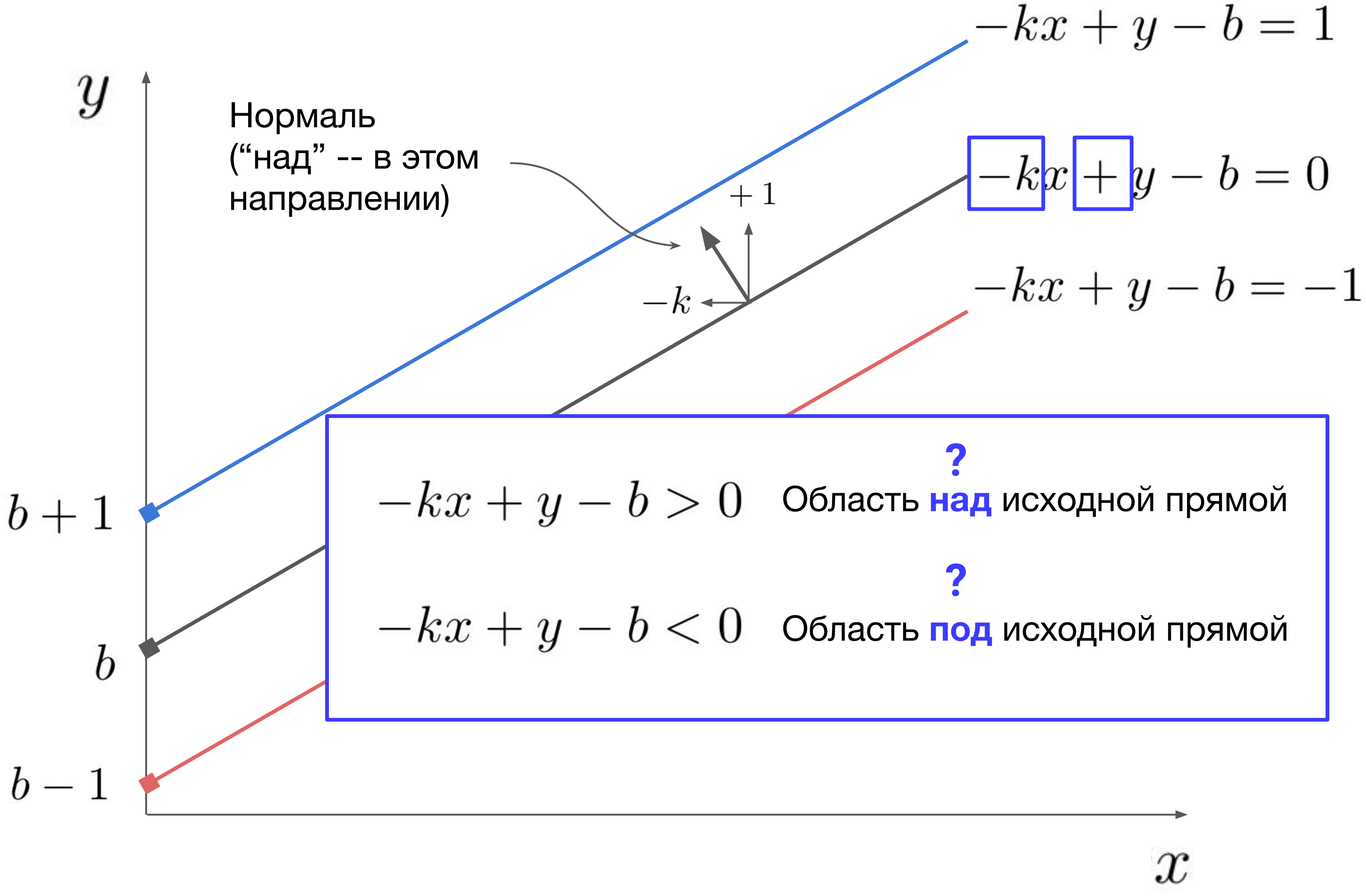


?

$-kx + y - b > 0$ Область **над** исходной прямой

?

$-kx + y - b < 0$ Область **под** исходной прямой



$-kx + y - b > 0$ Область **над** исходной прямой ?

$-kx + y - b < 0$ Область **под** исходной прямой ?

Линейный классификатор

Как **математически** записать:

1. ***“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”***

$$-kx + y - b > 0$$ Область **над** исходной прямой

$$-kx + y - b < 0$$ Область **под** исходной прямой

Линейный классификатор

Как **математически** записать:

1. ***“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”***

$$I[ax_1 + bx_2 + c > 0]$$

Линейный классификатор

Как **математически** записать:

1. ***“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”***

$$I[ax_1 + bx_2 + c > 0]$$

* I -- индикатор. Если утверждение внутри верно, он принимает значение 1, иначе -- 0.

Линейный классификатор

Как **математически** записать:

1. ***“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”***

$$I[ax_1 + bx_2 + c > 0]$$

* I -- индикатор. Если утверждение внутри верно, он принимает значение 1, иначе -- 0.

Признаки

Линейный классификатор

Как **математически** записать:

1. ***“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”***

$$I[\boxed{a}x_1 + \boxed{b}x_2 + \boxed{c} > 0]$$

* I -- индикатор. Если утверждение внутри верно, он принимает значение 1, иначе -- 0.

Признаки

Коэффициенты

Линейный классификатор

Как **математически** записать:

1. ***“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”***

$$I[ax_1 + bx_2 + cx_3 + \cdots + z > 0]$$

Линейный классификатор

Итог

- Вспомнили задачу классификации
- Записали линейный классификатор в общем виде
- Поняли, что означают его коэффиценты
    - Подробнее -- на практике
- И как он принимает решение

В следующем уроке: поймем как записать задачу оптимизации

Skillbox

Введение в нейронные сети

Функционал ошибки

Как **математически** записать:

1. *“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”*
2. *“эта прямая разделяет объекты лучше, чем другая”*

Skillbox

Как **математически** записать:

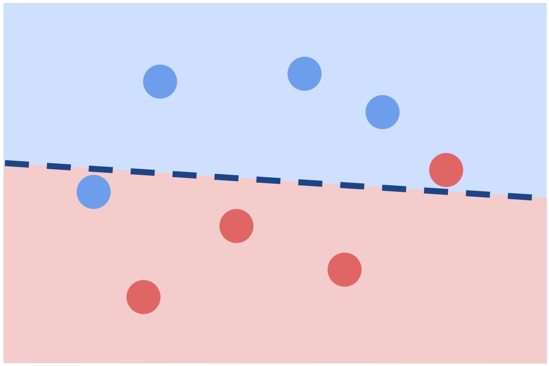
1. *“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”*
2. *“эта прямая разделяет объекты лучше, чем другая”*

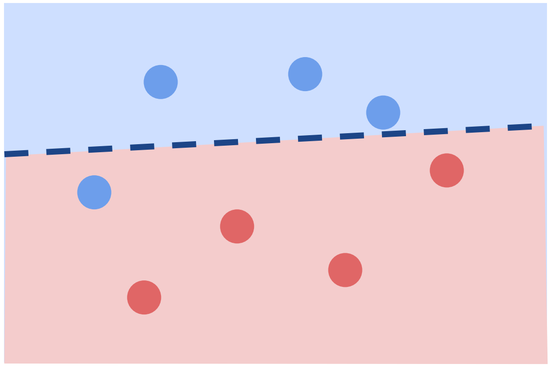
Прошлый урок

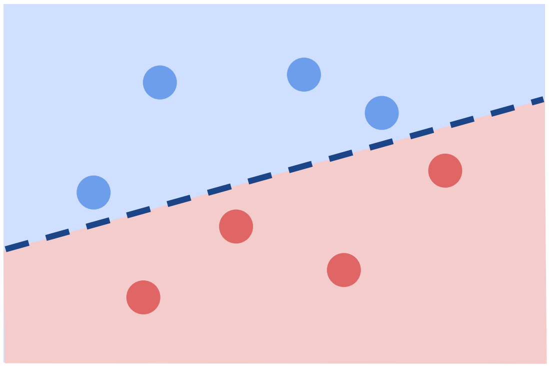
Как **математически** записать:

1. *“объектам с одной стороны от прямой присваиваем синий цвет, а с другой -- красный”*
2. ***“эта прямая разделяет объекты лучше, чем другая”***

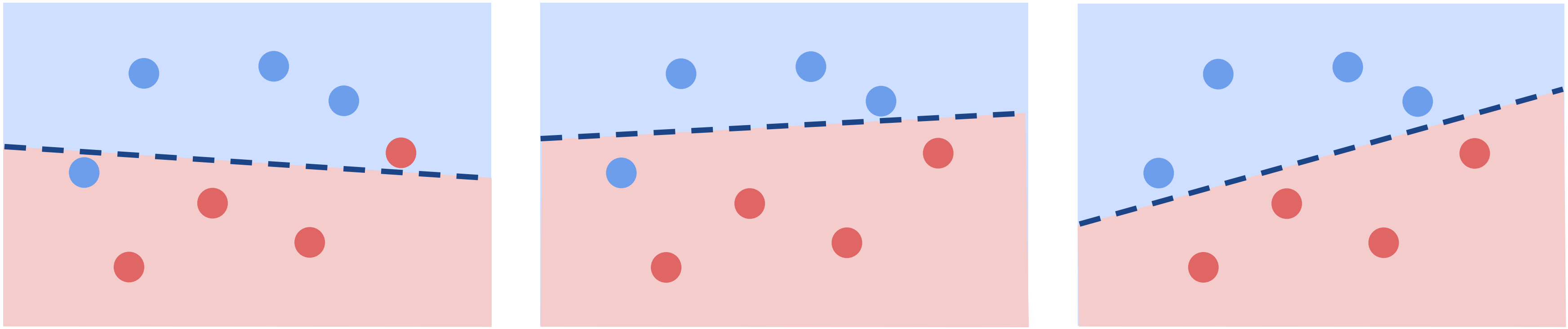
Какое разбиение лучше?







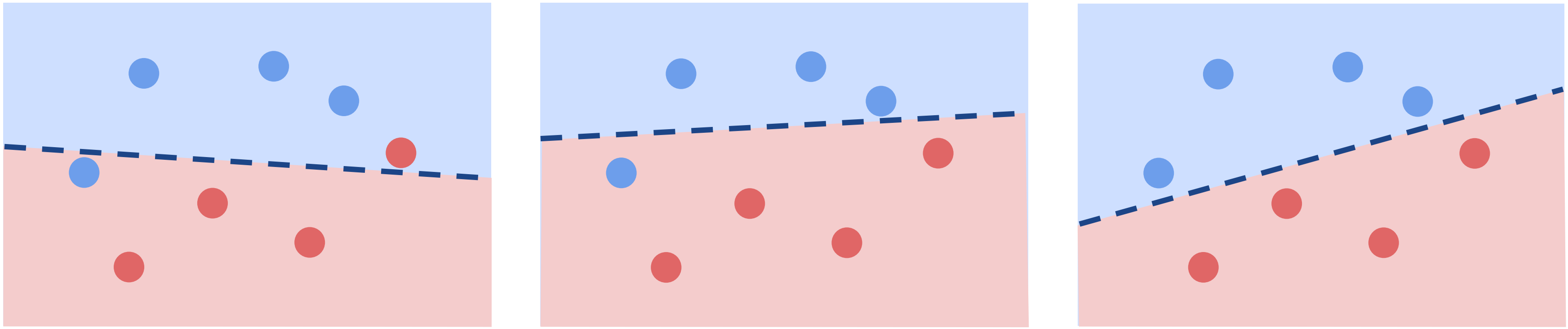
Какое разбиение лучше?



“Там, где меньше ошибок!”

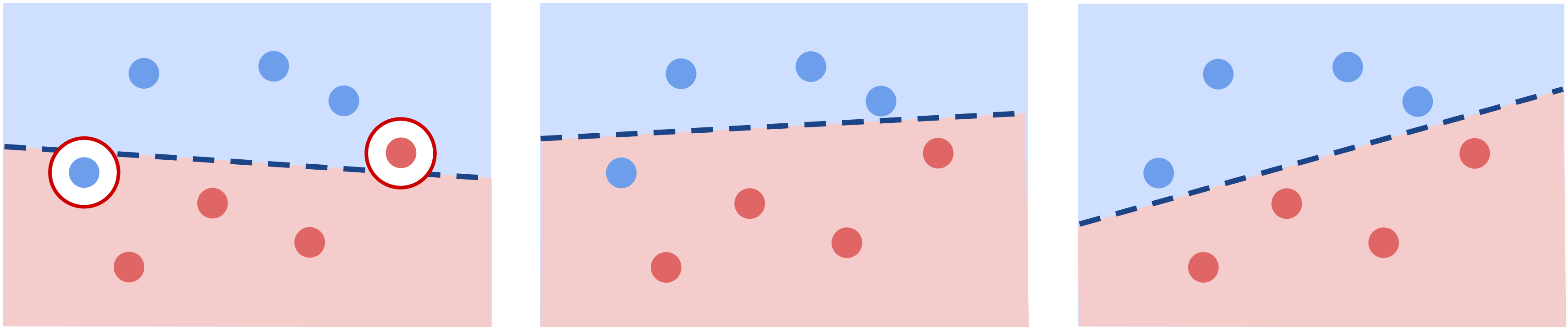
Skillbox

Какое разбиение лучше?



“Там, где меньше ошибок!” ~ “Там, где выше точность (accuracy)!”

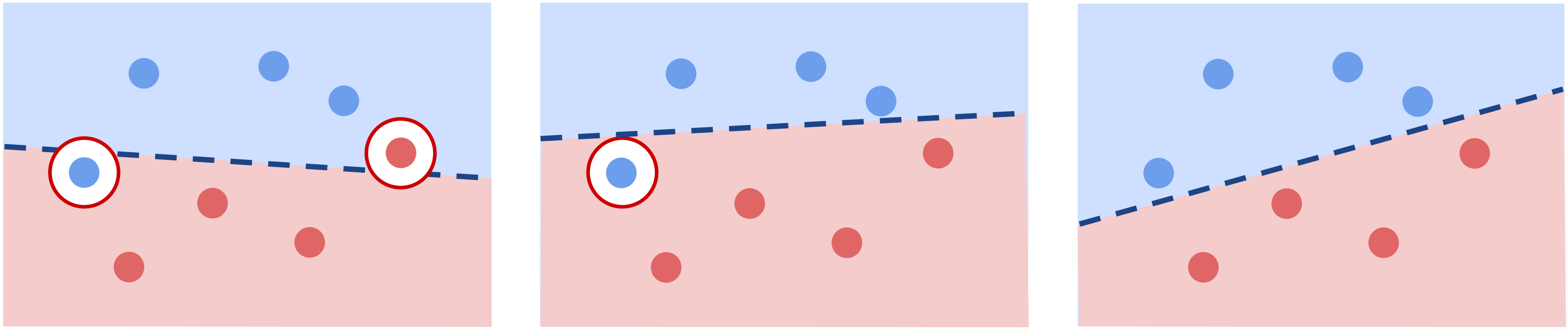
Какое разбиение лучше?



"Там, где меньше ошибок!" ~ "Там, где выше точность (accuracy)!"

Skillbox

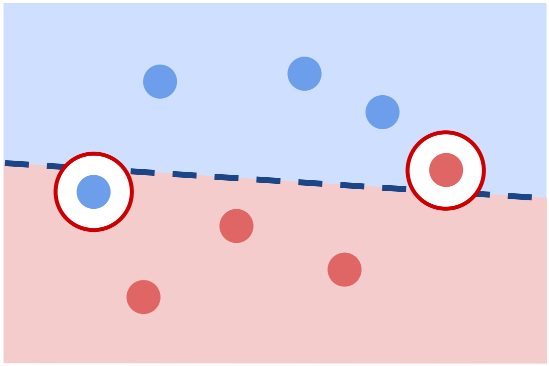
Какое разбиение лучше?



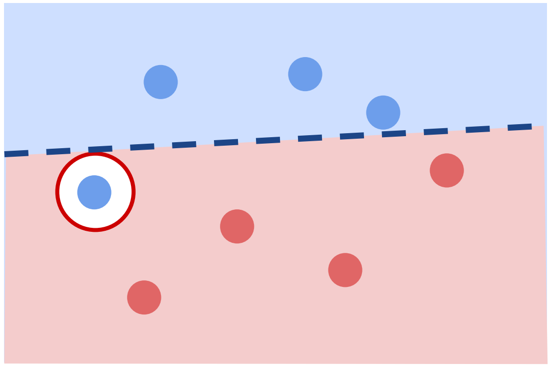
“Там, где меньше ошибок!” ~ “Там, где выше точность (accuracy)!”

Какое разбиение лучше?

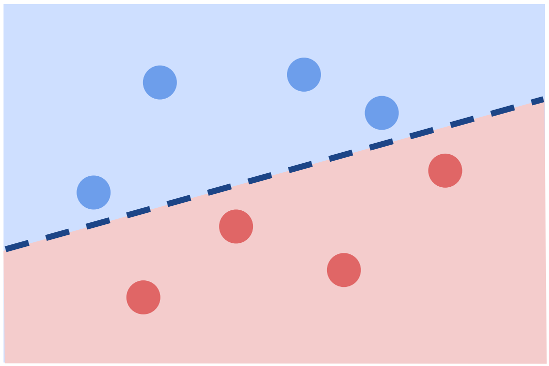
2 ошибки, 6/8 точность



1 ошибка, 7/8 точность



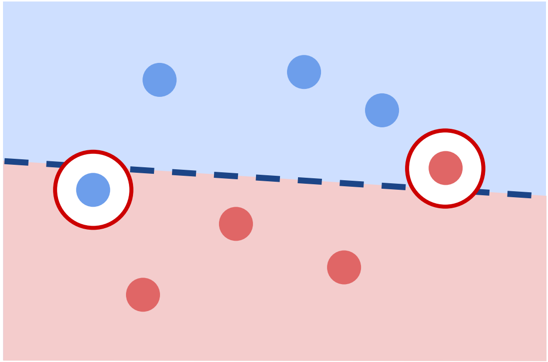
0 ошибок, 8/8 точность



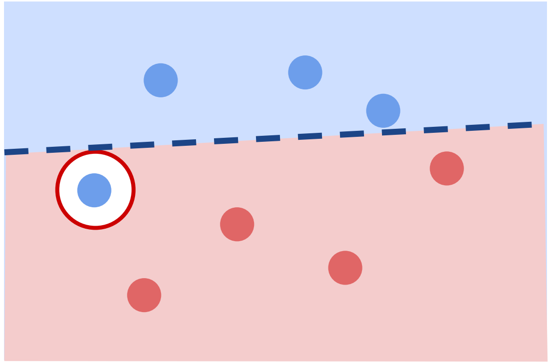
“Там, где меньше ошибок!” ~ “Там, где выше точность (accuracy)!”

Какое разбиение лучше?

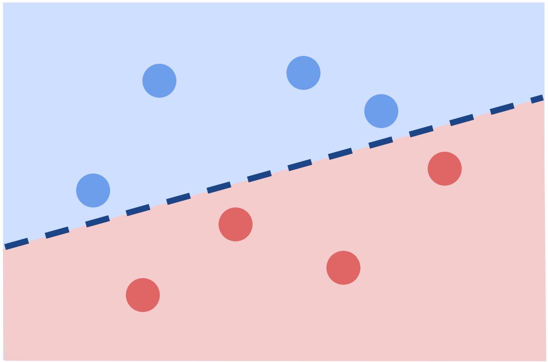
2 ошибки, 6/8 точность



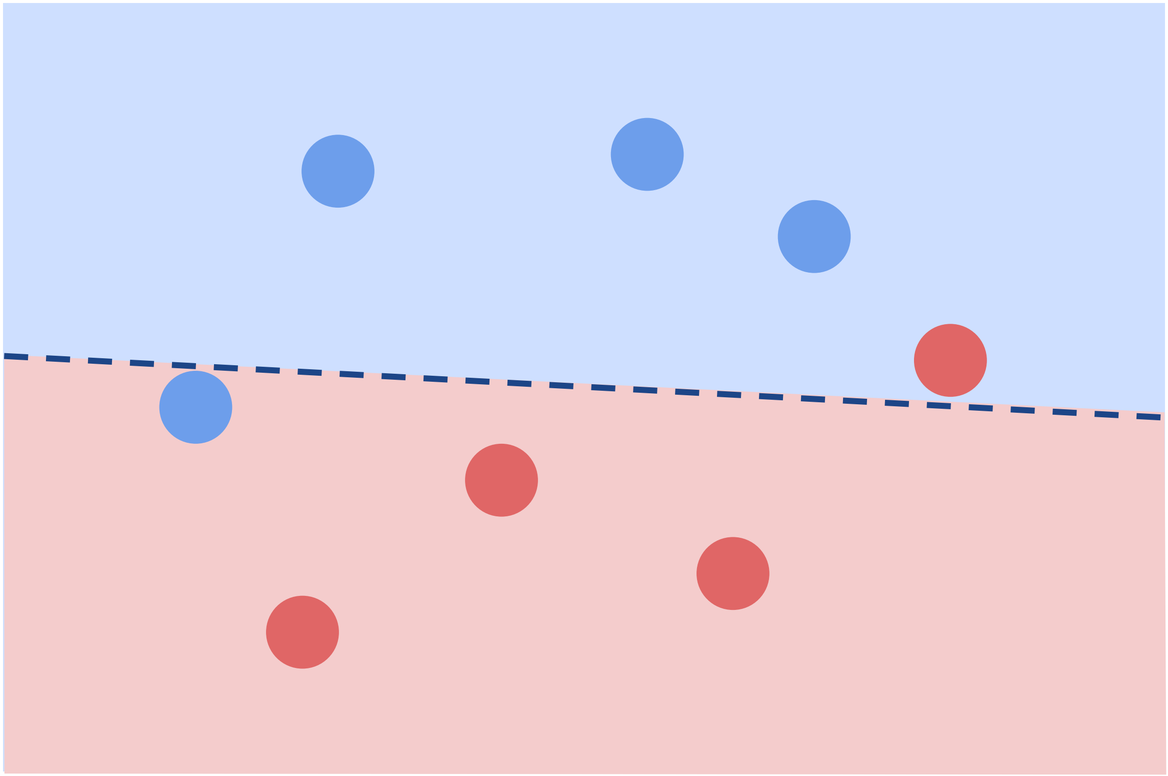
1 ошибка, 7/8 точность

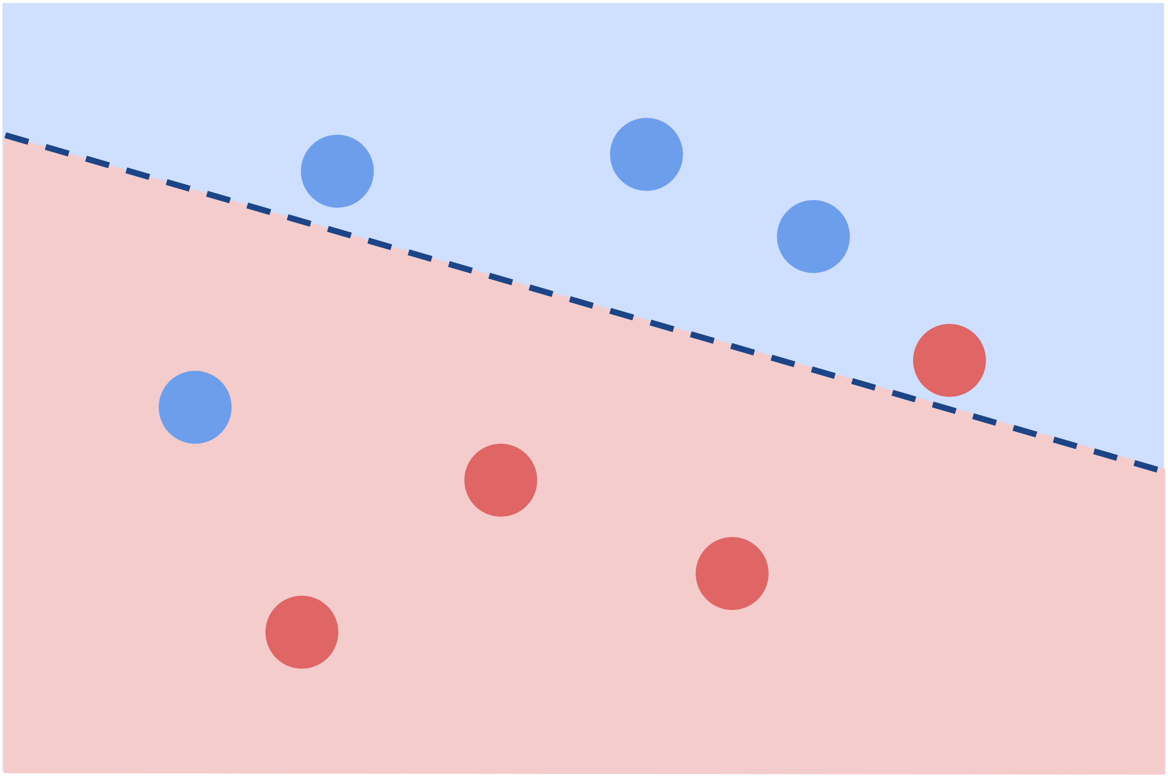


0 ошибок, 8/8 точность

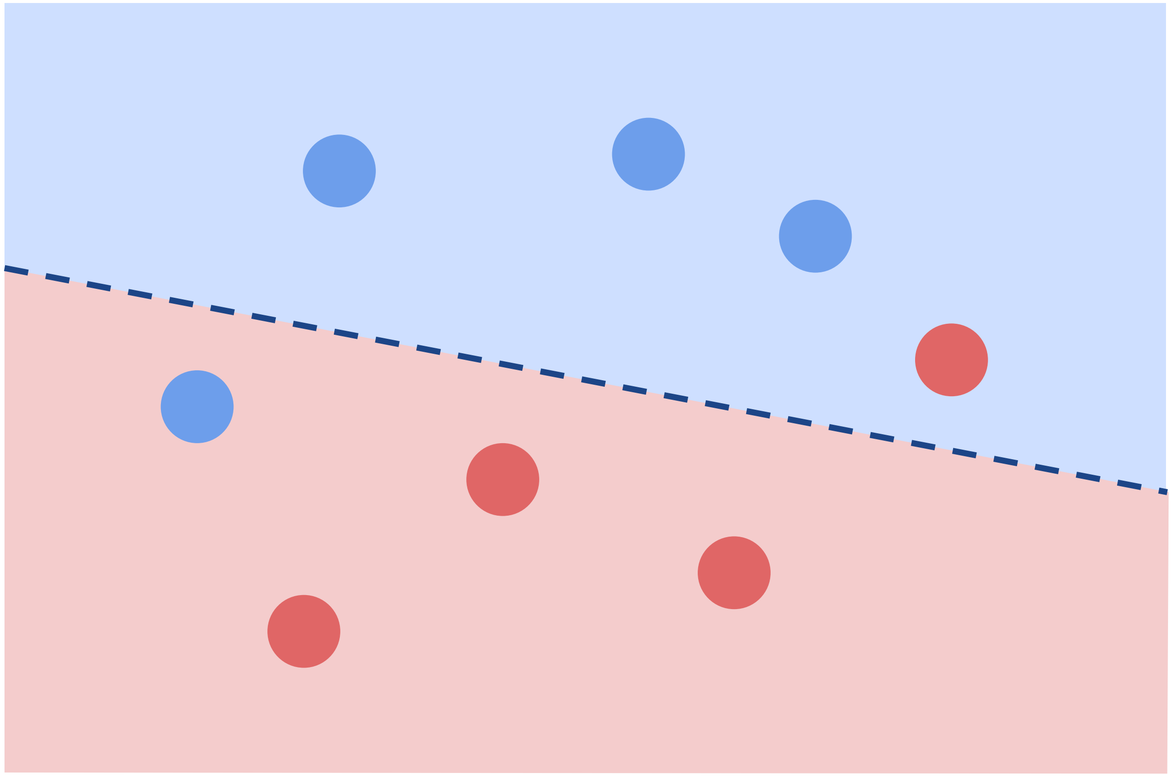


“Там, где **меньше** ошибок!” ~ “Там, где **выше** точность (accuracy)!”



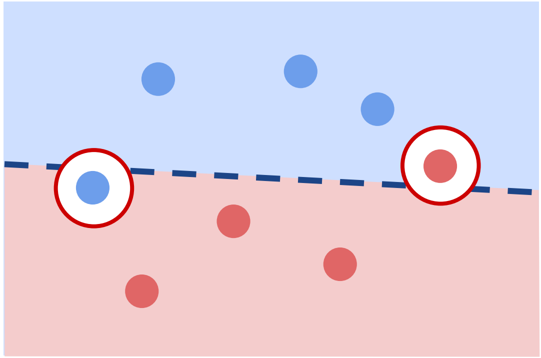


Skillbox

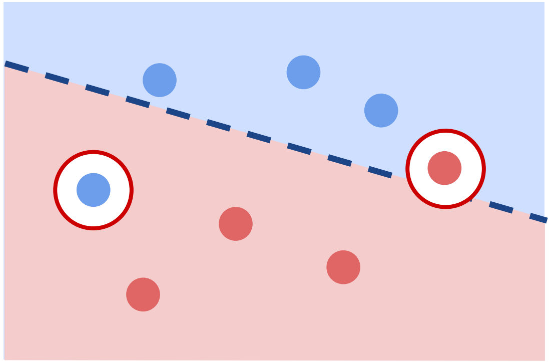


Какое разбиение лучше?

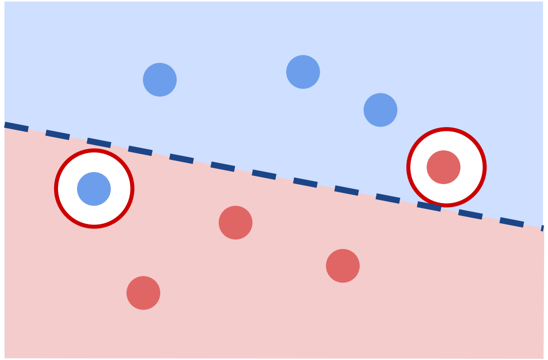
2 ошибки, 6/8 точность



2 ошибки, 6/8 точность



2 ошибки, 6/8 точность

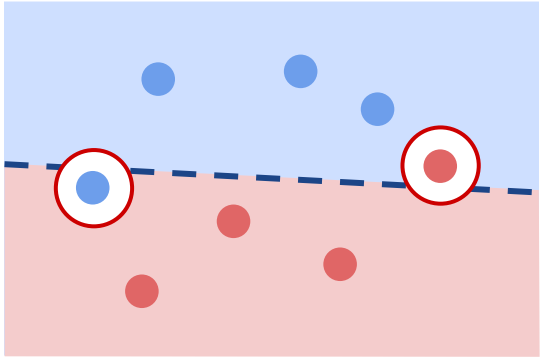


- Одинаковы с точки зрения нашего функционала ошибки.

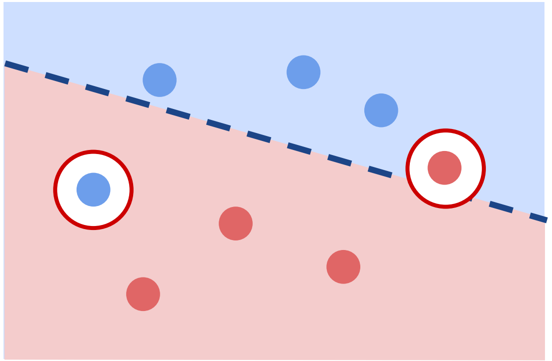
Skillbox

Какое разбиение лучше?

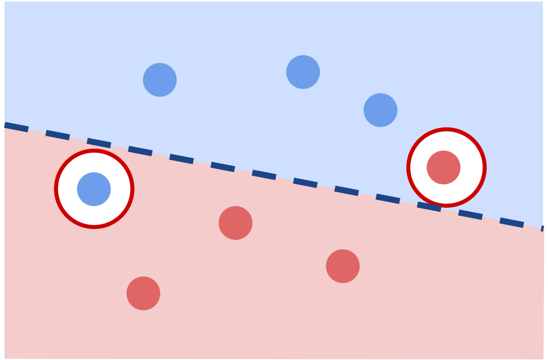
2 ошибки, 6/8 точность



2 ошибки, 6/8 точность



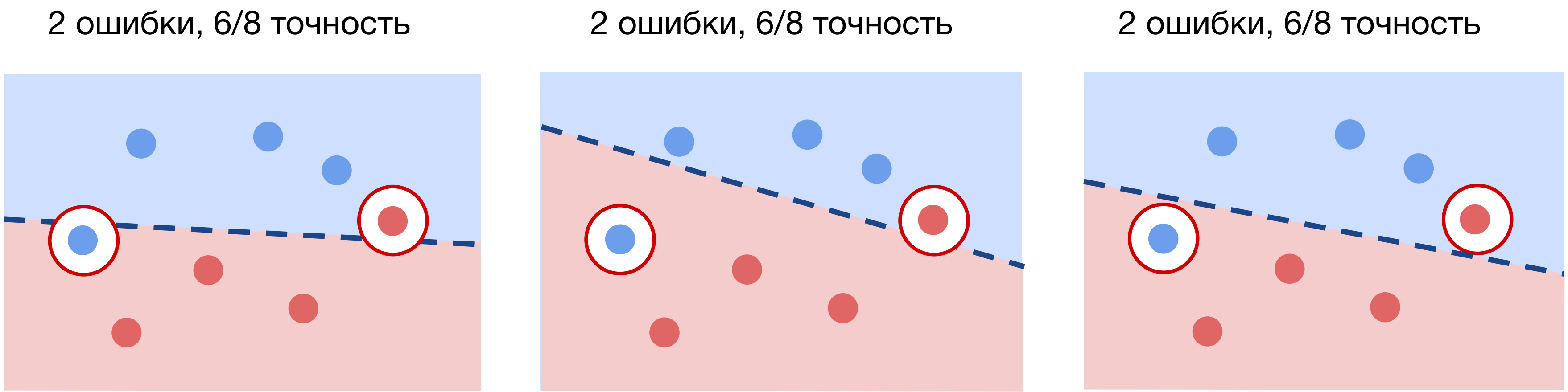
2 ошибки, 6/8 точность



- Одинаковы с точки зрения нашего функционала ошибки.
- В первом случае нужно довернуть прямую совсем чуть-чуть

Skillbox

Какое разбиение лучше?



- Одинаковы с точки зрения нашего функционала ошибки.
- В первом случае нужно довернуть прямую совсем чуть-чуть
- Нам **важно** чтобы наш функционал ошибки различал и такие случаи
    - **Небольшое изменение параметров приводит к небольшому изменению функционала** качества -- без резких скачков и константных областей.
    - Почему это важно?

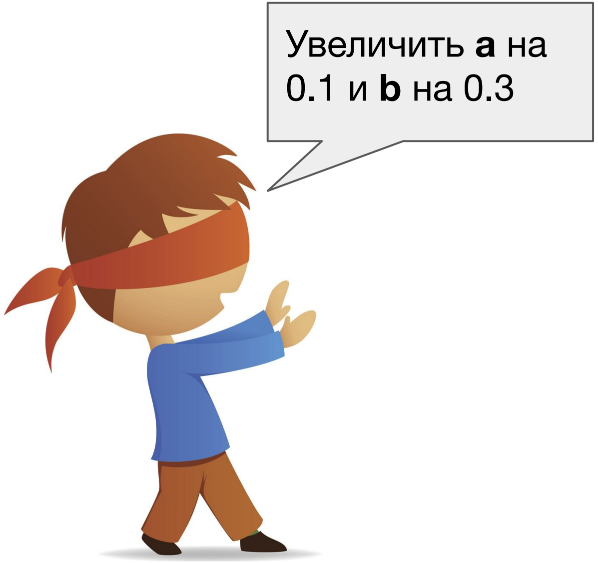
Skillbox

Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой

Функционал качества

Алгоритм оптимизации

Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой



Функционал качества

Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой

Увеличить **a** на 0.1 и **b** на 0.3

Холодно!

Функционал качества

Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой

Увеличить **a** на 0.1 и **b** на 0.3

Горячо!

Функционал качества

Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой

Функционал качества

Аналогия: Игра "горячо-холодно" в поисках идеальных параметров прямой

Функционал качества

- Точность или количество ошибок -- полезны для **нас**, но не для **алгоритма**

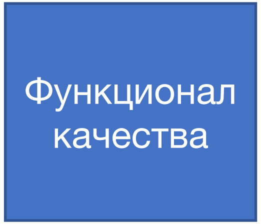
Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой

Функционал качества

- Точность или количество ошибок -- полезны для **нас**, но не для **алгоритма**
- Они не дают дополнительной информации куда “двигаться”

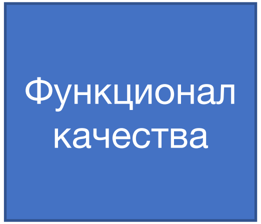
Skillbox

Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой



- Точность или количество ошибок -- полезны для **нас**, но не для **алгоритма**
- Они не дают дополнительной информации куда “двигаться”
- Алгоритм не знает куда идти чтобы найти параметры лучше, т.к. ответ функционала всегда -- “ничего не изменится”

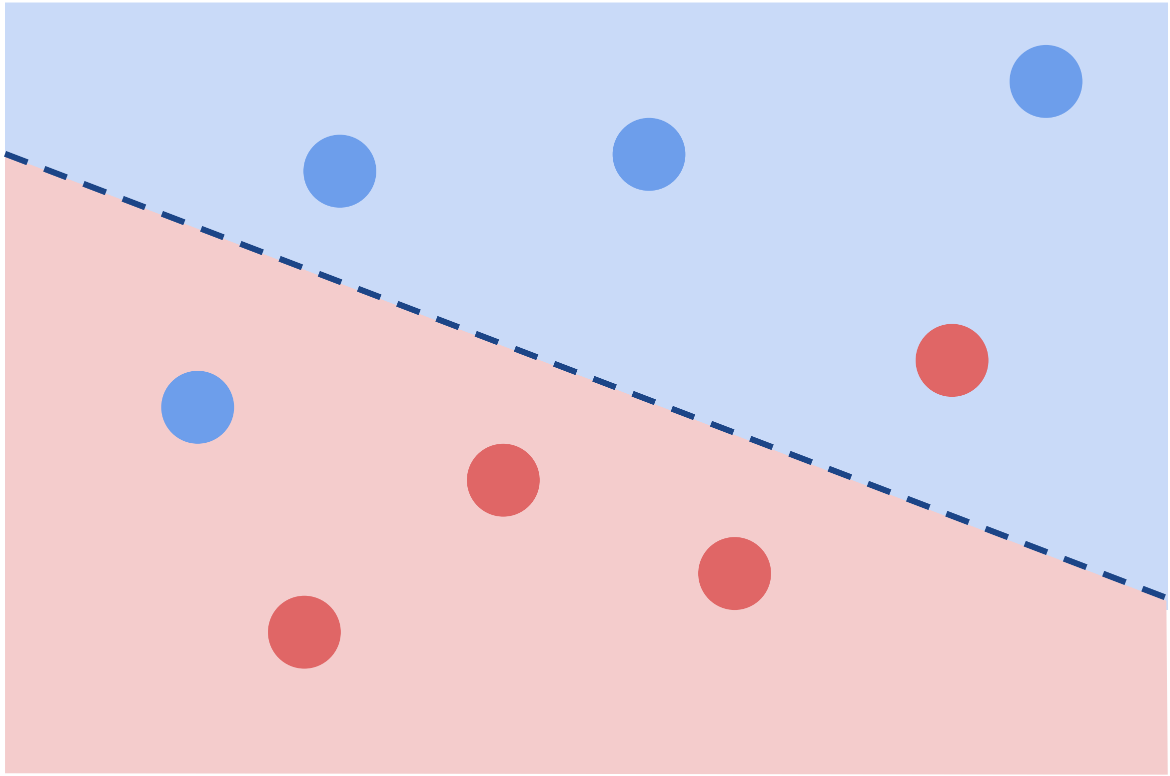
Аналогия: Игра ”горячо-холодно” в поисках идеальных параметров прямой



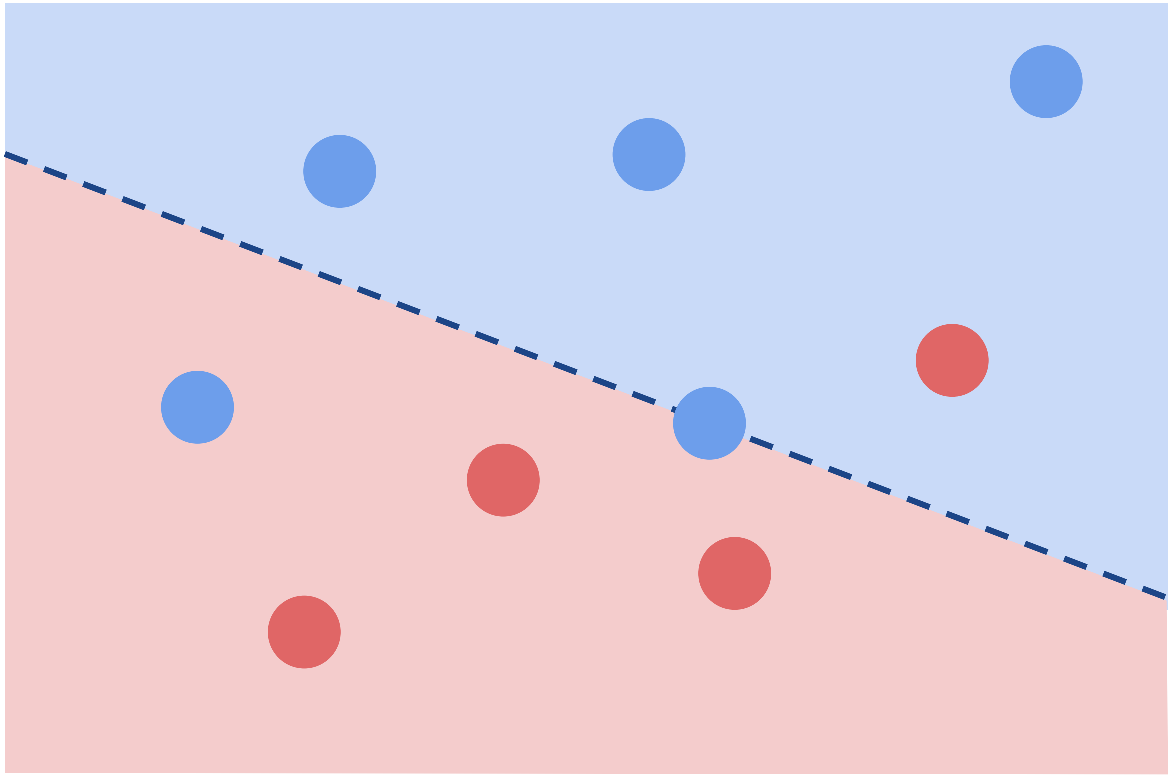
- Точность или количество ошибок -- полезны для **нас**, но не для **алгоритма**
- Они не дают дополнительной информации куда “двигаться”
- Алгоритм не знает куда идти чтобы найти параметры лучше, т.к. ответ функционала всегда -- “ничего не изменится”
- Давайте построим такую функцию!

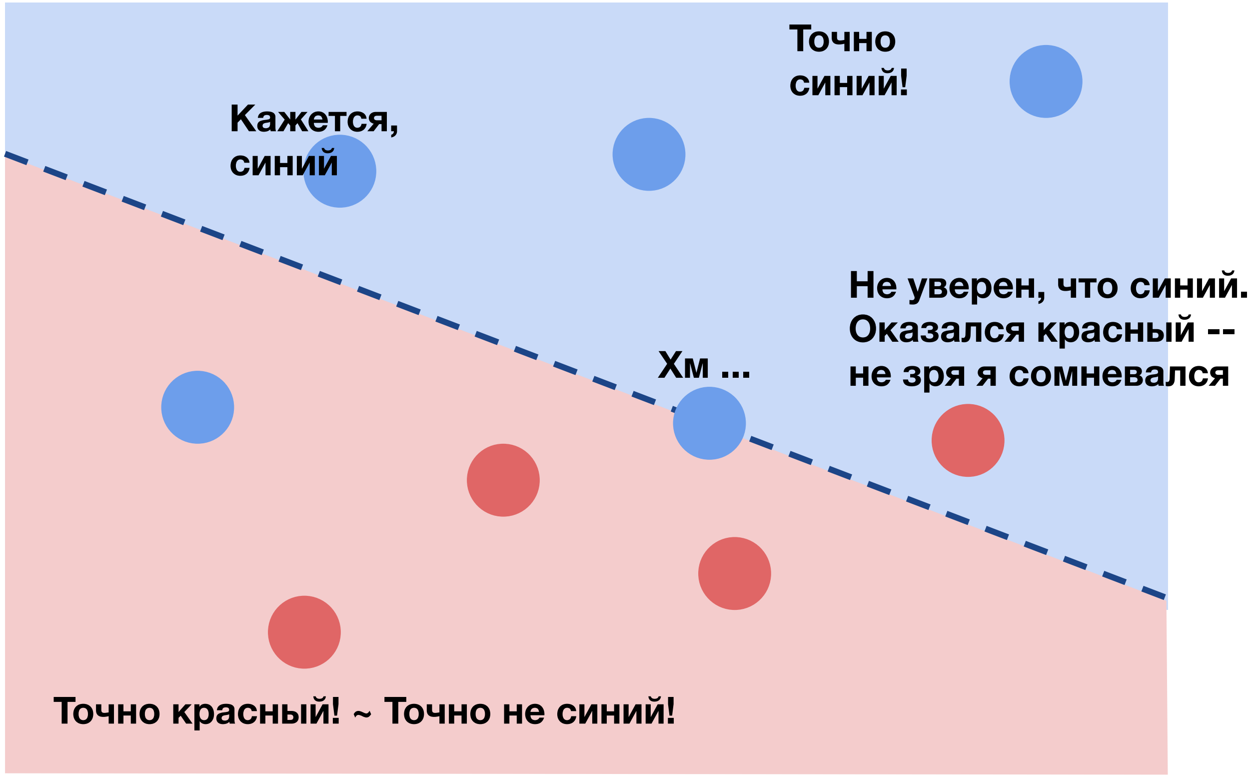
Skillbox

$$I[ax_1 + bx_2 + c > 0]$$

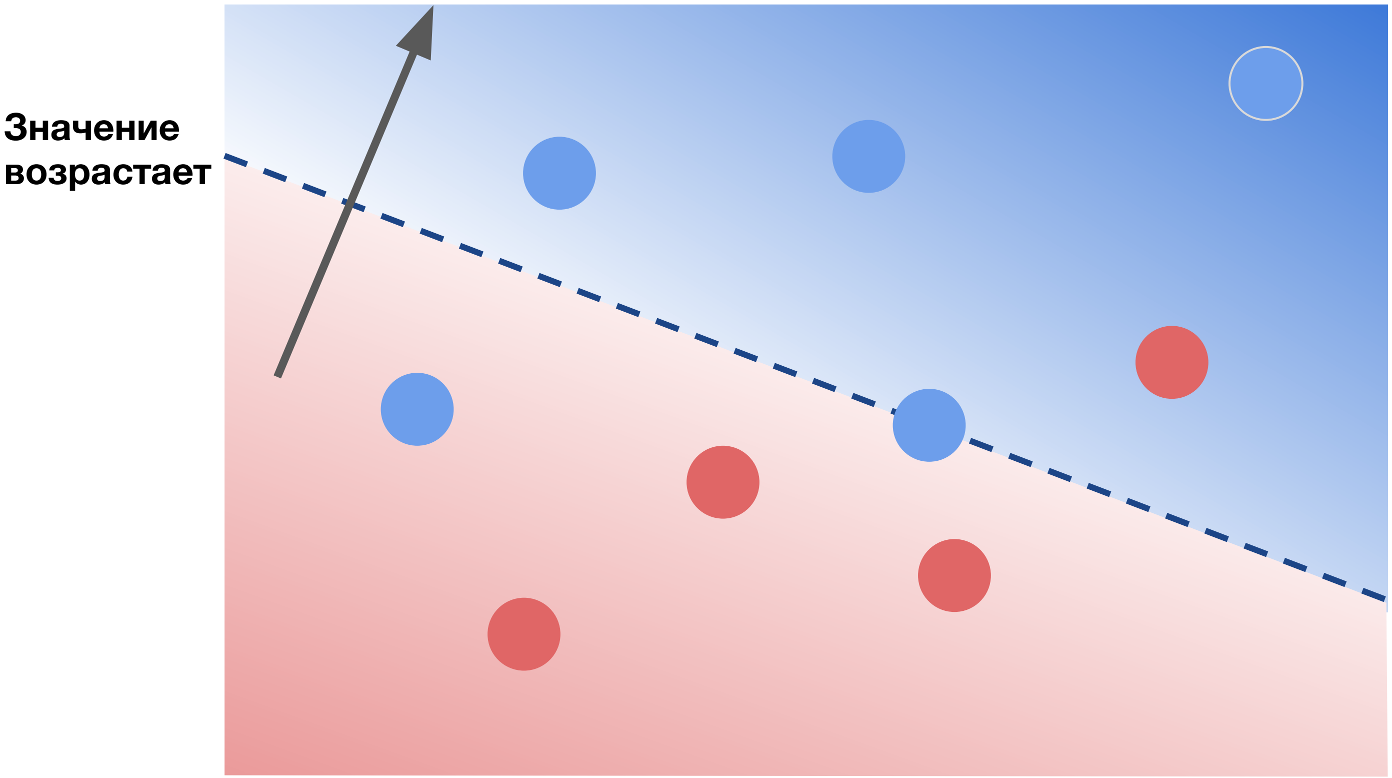


$$I[ax_1 + bx_2 + c > 0]$$




Точно
синий!
Кажется,
синий
Не уверен, что синий.
Оказался красный --
не зря я сомневался
Хм ...
Точно красный! ~ Точно не синий!

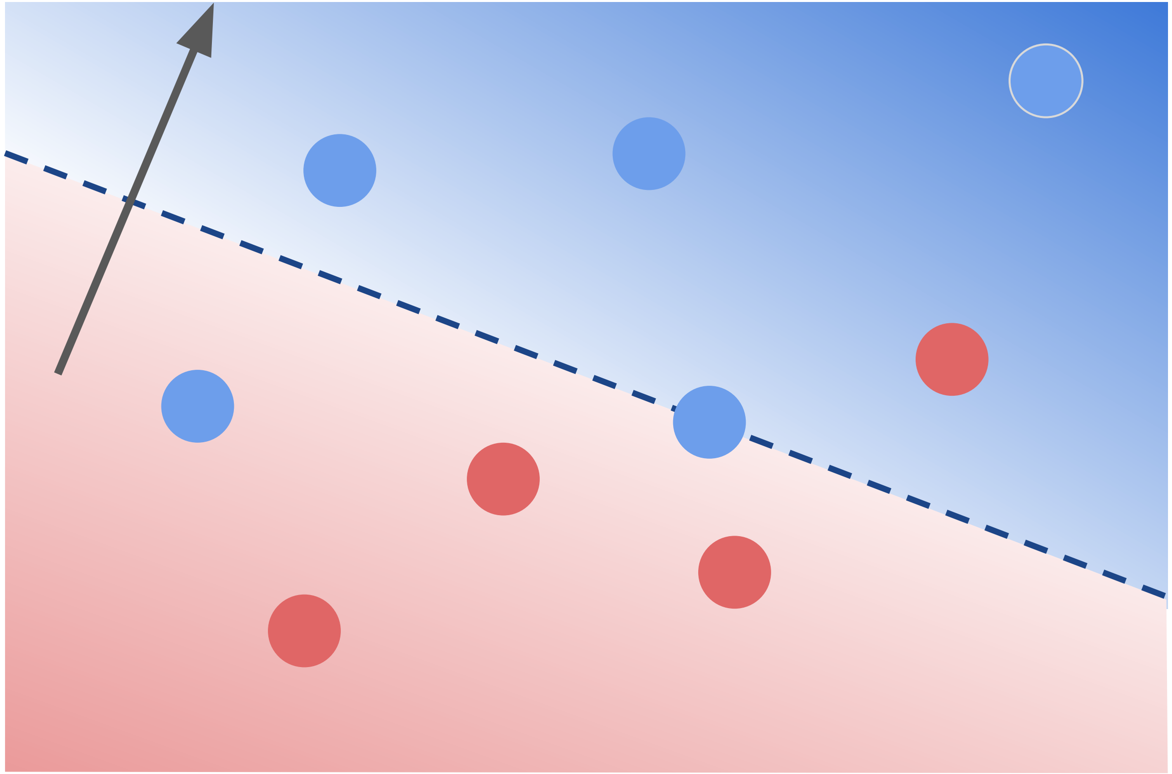
$$ax_1 + bx_2 + c$$



Skillbox

$$ax_1 + bx_2 + c$$

А что если, <u>относится</u> к этому как к вероятности?*

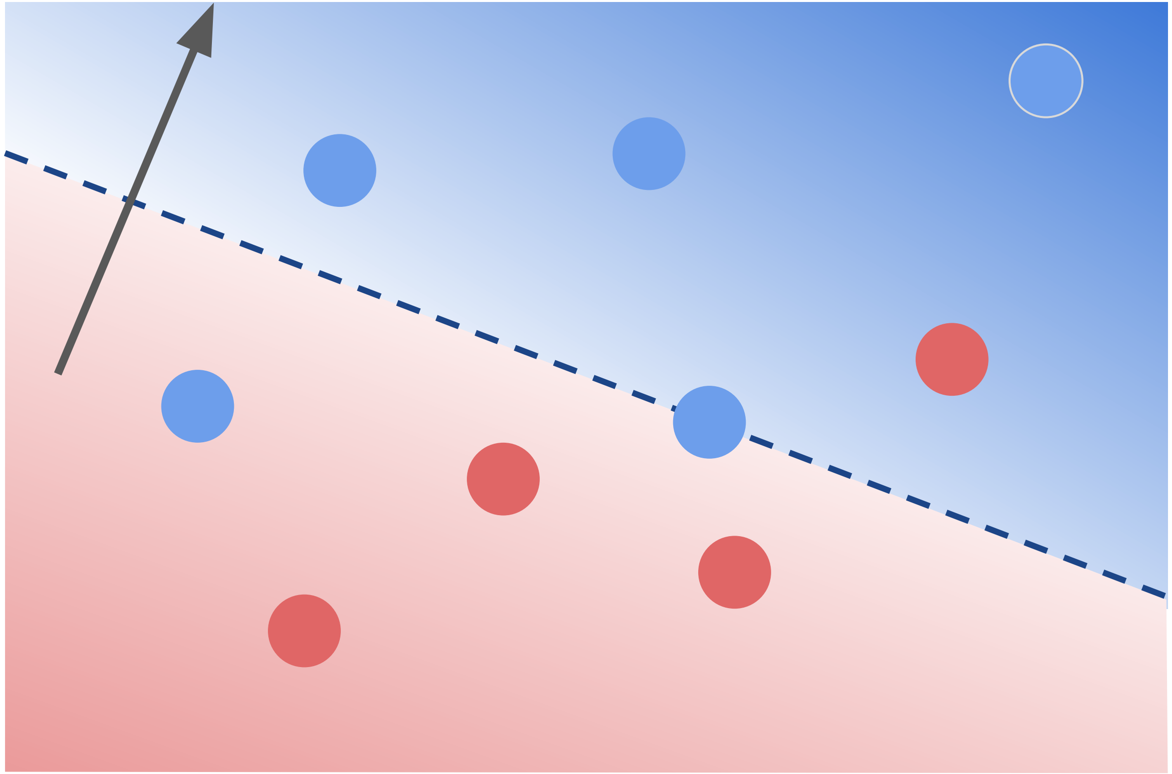


Skillbox

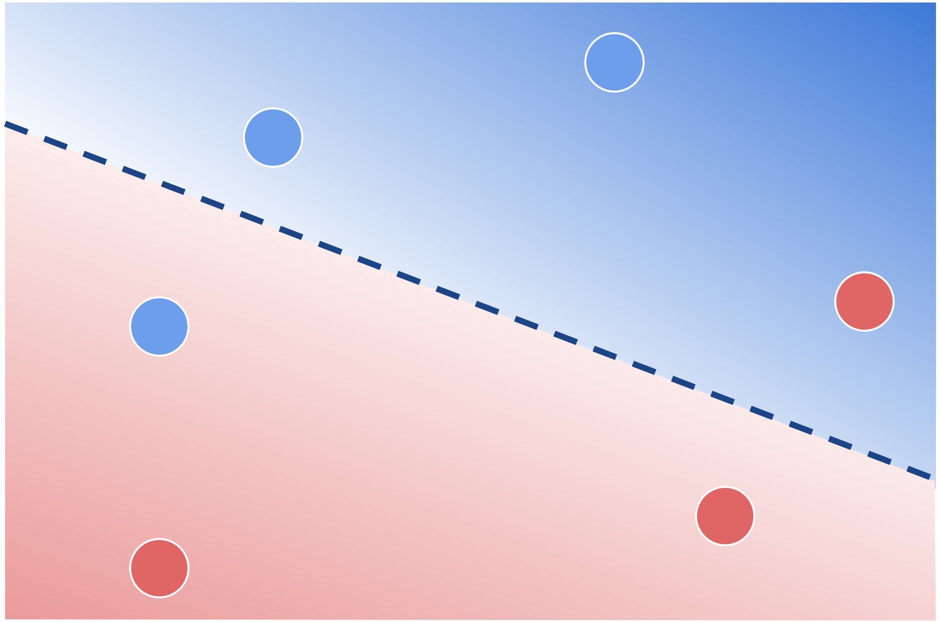
$$ax_1 + bx_2 + c$$

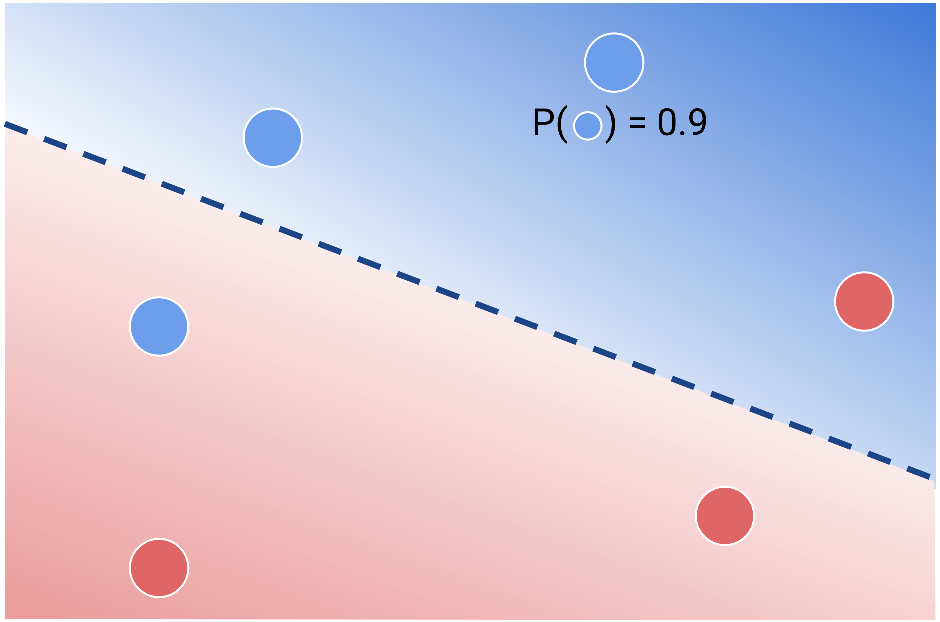
А что если, <u>относится</u> к этому как к вероятности?*

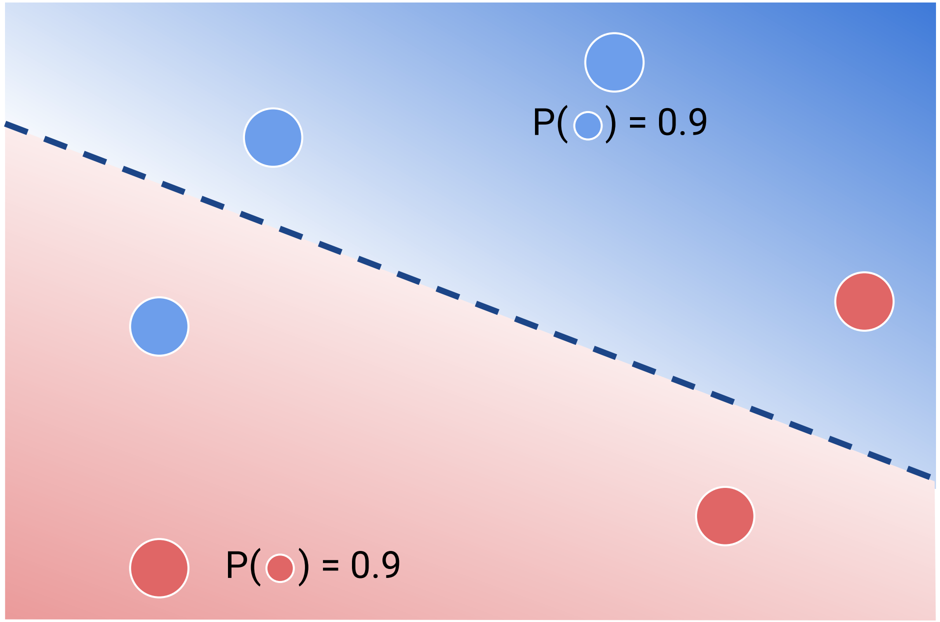
(не совсем строгое утверждение, но позволим его себе для простоты усвоения материала. В следующем уроке разберемся с этим подробнее)

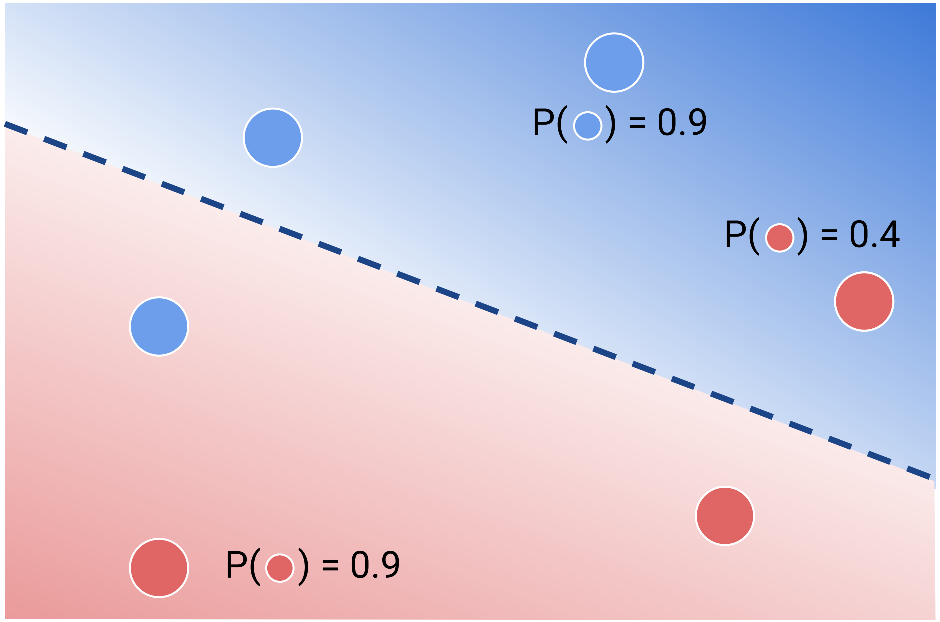


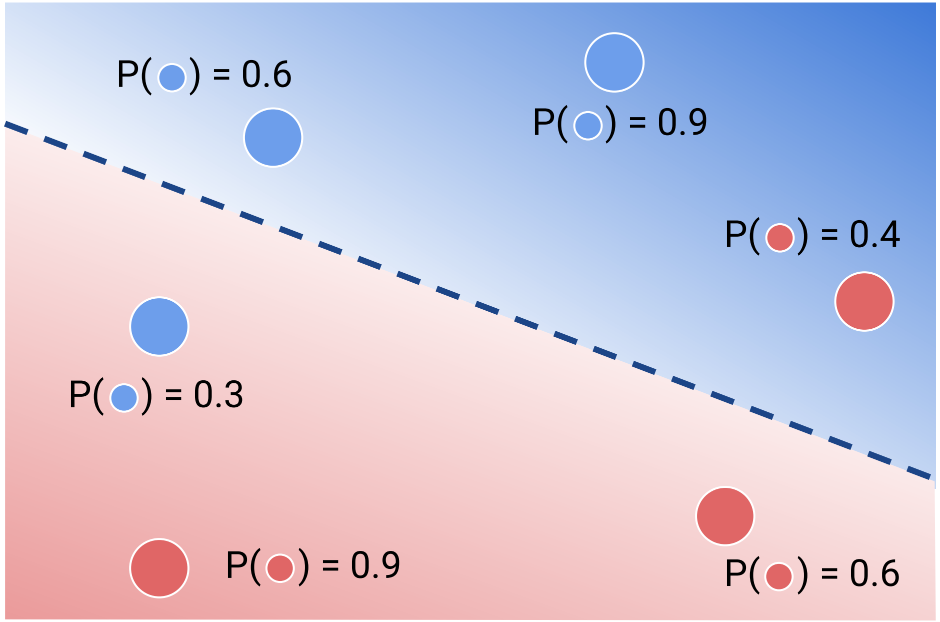
Skillbox



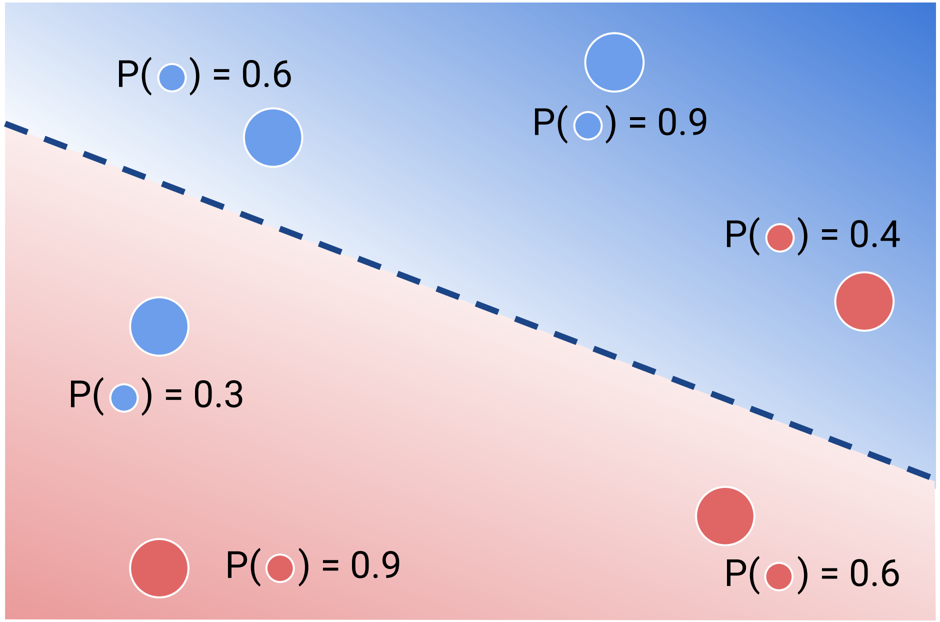

P( ) = 0.9


P( ) = 0.9
P( ) = 0.9

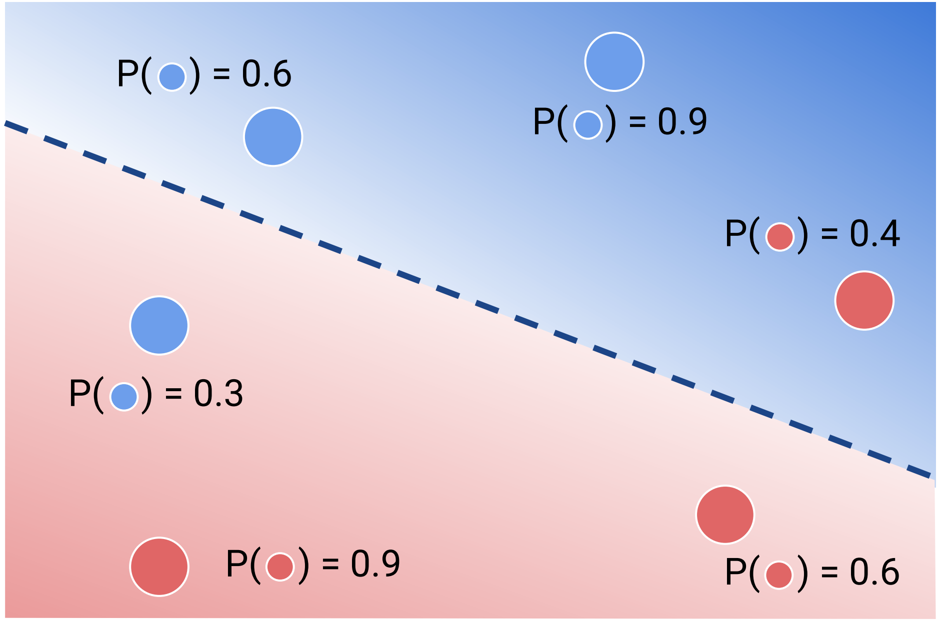

P( ) = 0.9
P( ) = 0.4
P( ) = 0.9


P( ) = 0.6
P( ) = 0.9
P( ) = 0.4
P( ) = 0.3
P( ) = 0.9
P( ) = 0.6

Skillbox



- Чем нам это помогло?



- Чем нам это помогло?
- Теперь мы можем посчитать вероятность нашей выборки!

Принцип максимума правдоподобия

- Пусть мы подбросили монету 100 раз и из них 60 раз выпал орел.
- **Какая у этой монеты вероятность выпадения орла?**
  - Интуитивно: 60%

Принцип максимума правдоподобия

- Пусть мы подбросили монету 100 раз и из них 60 раз выпал орел.
- **Какая у этой монеты вероятность выпадения орла?**
  - Интуитивно: 60%

- Если вероятность выпадения монетки равна p_0, **то с какой вероятностью мы могли получить такие исходы?**

Принцип максимума правдоподобия

- Пусть мы подбросили монету 100 раз и из них 60 раз выпал орел.
- **Какая у этой монеты вероятность выпадения орла?**
  - Интуитивно: 60%

- Если вероятность выпадения монетки равна p_0, **то с какой вероятностью мы могли получить такие исходы?**
  - Вероятность исходов при предположении о параметрах модели -- правдоподобие (likelihood)

Skillbox

Принцип максимума правдоподобия

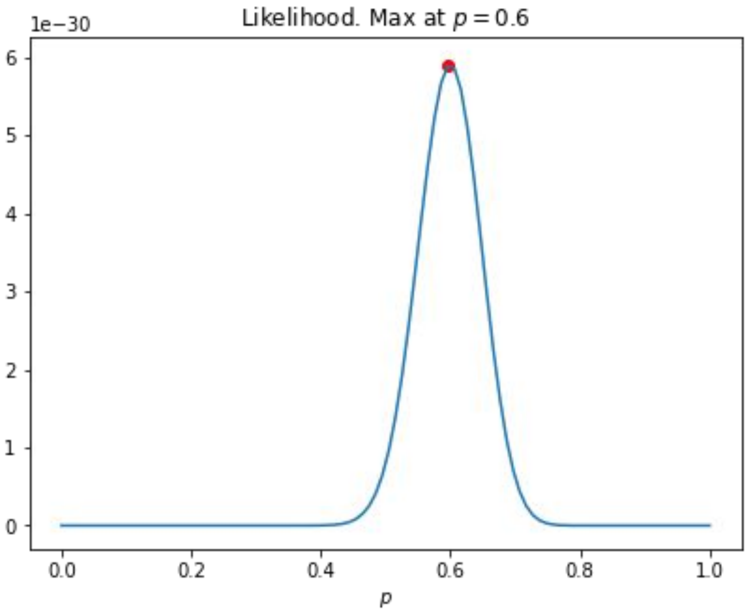
- Пусть мы подбросили монету 100 раз и из них 60 раз выпал орел.
- **Какая у этой монеты вероятность выпадения орла?**
  - Интуитивно: 60%

- Если вероятность выпадения монетки равна p_0, **то с какой вероятностью мы могли получить такие исходы?**
  - Вероятность исходов при предположении о параметрах модели -- правдоподобие (likelihood)

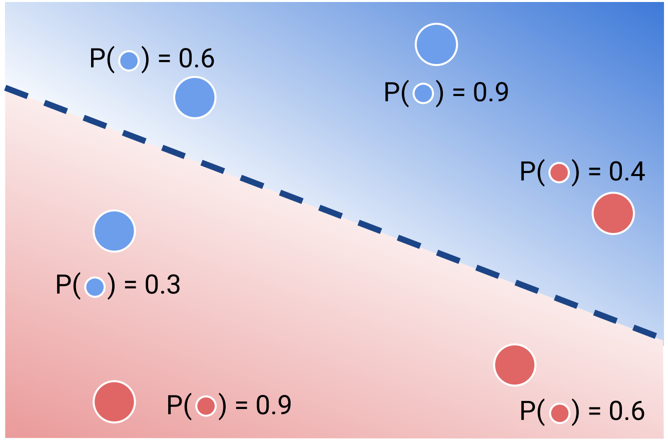
- Если мы переберем все p_0, то обнаружим его при p_0=60% значение правдоподобия максимально!

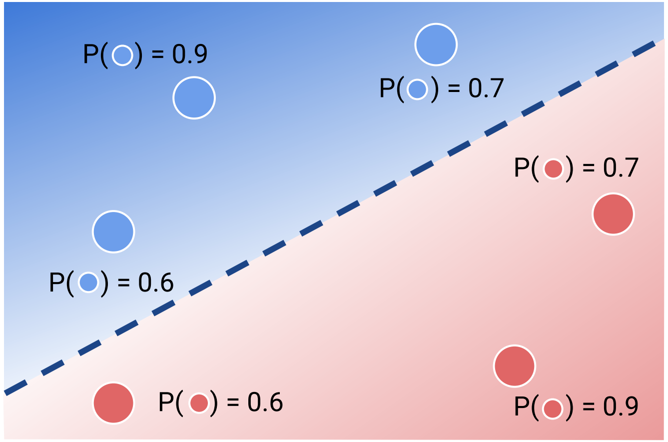
Принцип максимума правдоподобия

- Пусть мы подброс
орел.
- **Какая у этой мон**
  - Интуитивно:
- Если вероятность
вероятностью мы
  - Вероятность
модели -- пр
- Если мы перебере
значение правдопо

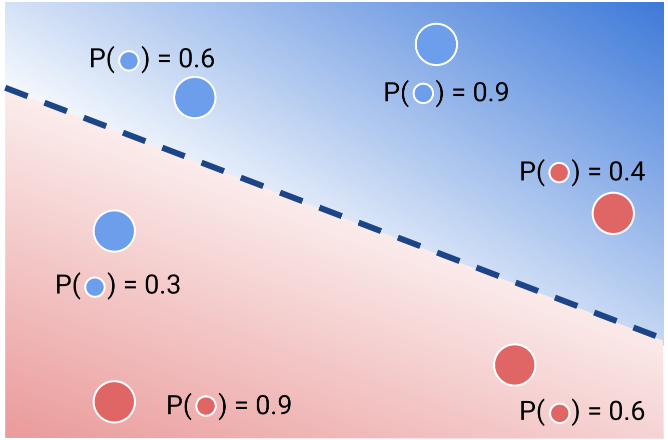


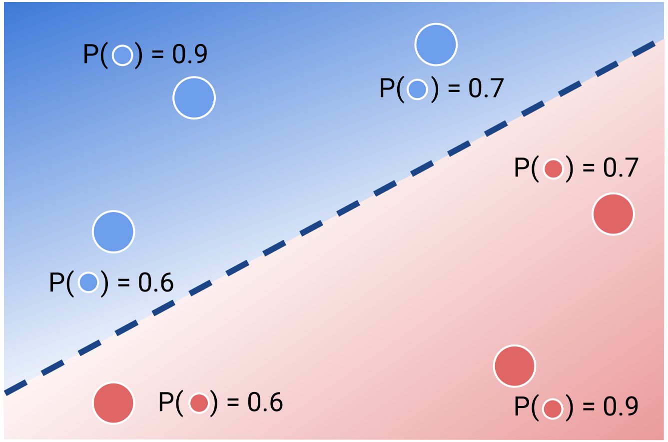
Skillbox



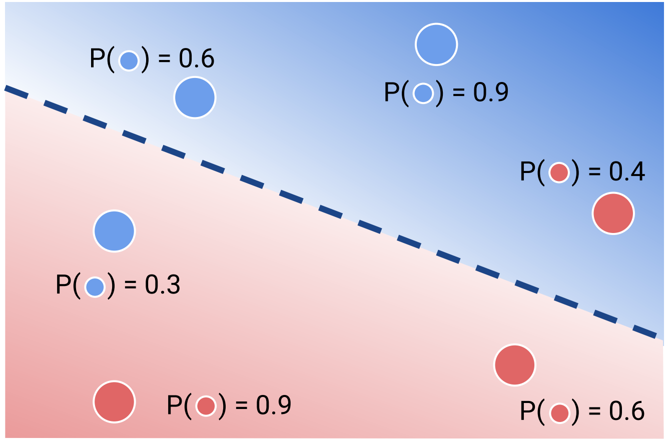


Skillbox

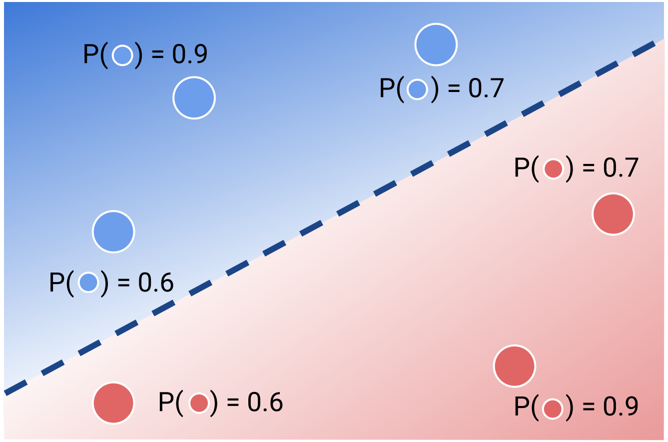




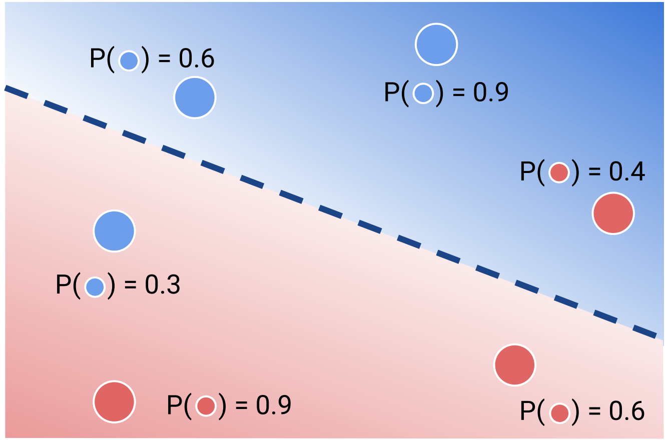
0.6*0.3*0.9*0.4*0.6*0.9 = 0.03

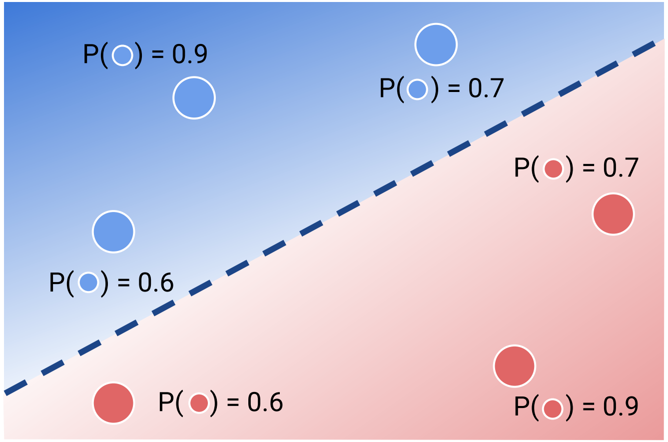


$0.6*0.3*0.9*0.4*0.6*0.9 = 0.03$



$0.9*0.6*0.7*0.6*0.9*0.7 = 0.14$

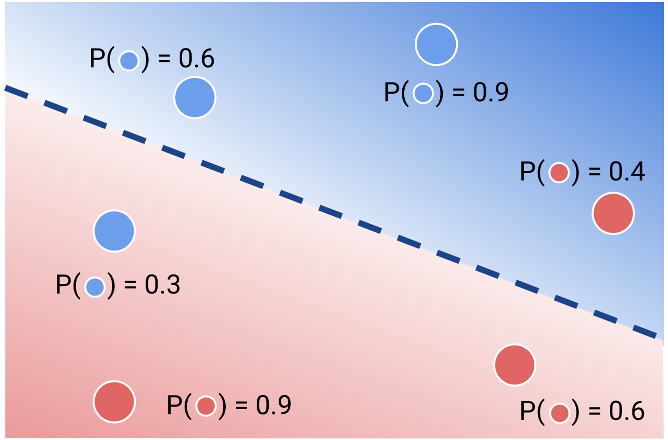




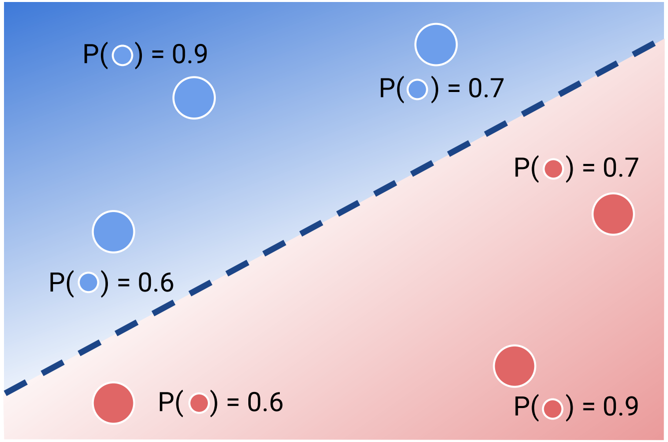
0.6*0.3*0.9*0.4*0.6*0.9 = 0.03

0.9*0.6*0.7*0.6*0.9*0.7 = 0.14

Чем **больше**, тем **лучше**

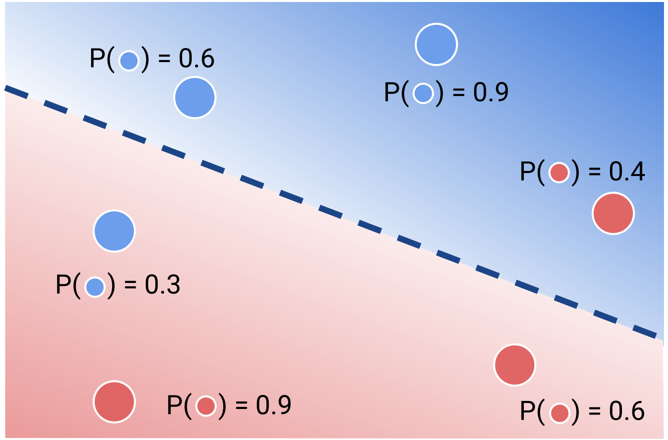


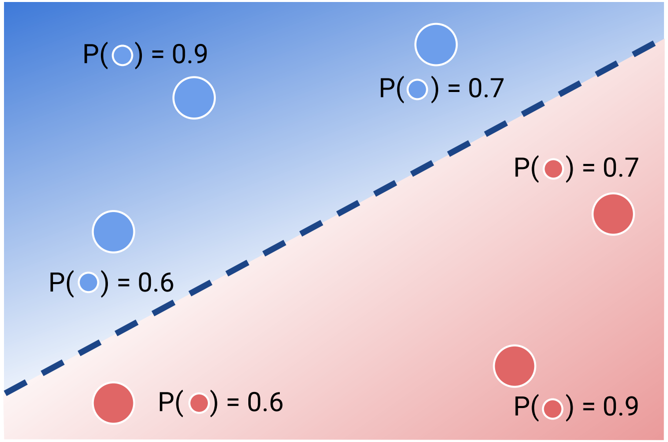
0.6*0.3*0.9*0.4*0.6*0.9 = 0.03



0.9*0.6*0.7*0.6*0.9*0.7 = 0.14

$$log(ab) = log(a) + log(b)$$





0.6*0.3*0.9*0.4*0.6*0.9 = 0.03

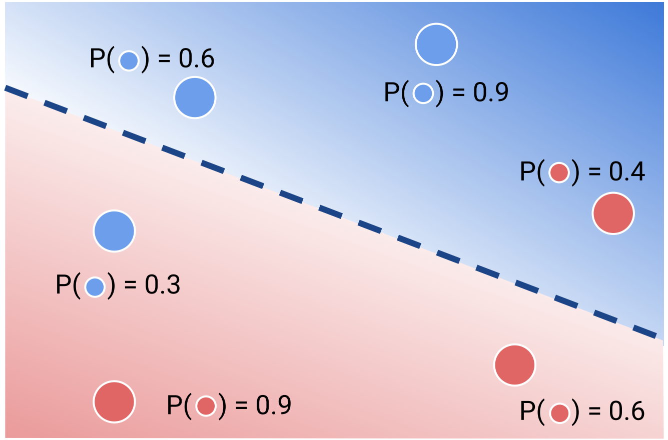
0.9*0.6*0.7*0.6*0.9*0.7 = 0.14

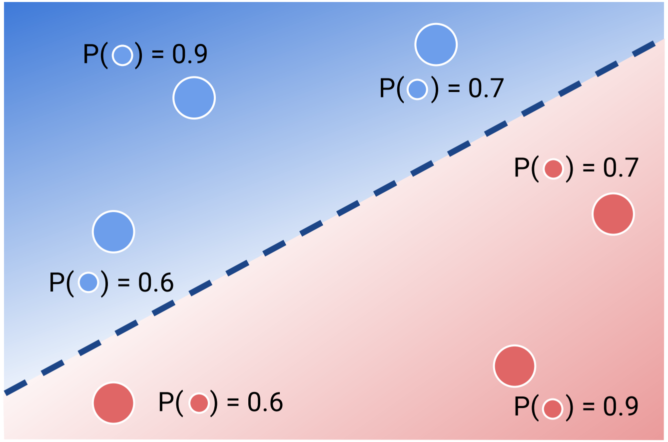
$$log(ab) = log(a) + log(b)$$

log(0.6) + log(0.3) + log(0.9) + +log(0.4) + log(0.6) + log(0.9) = -1.45

log(0.9) + log(0.6) + log(0.7) + +log(0.6) + log(0.9) + log(0.7) = -0.84

Чем **больше**, тем **лучше**. **Задача максимизации “качества”**





0.6*0.3*0.9*0.4*0.6*0.9 = 0.03

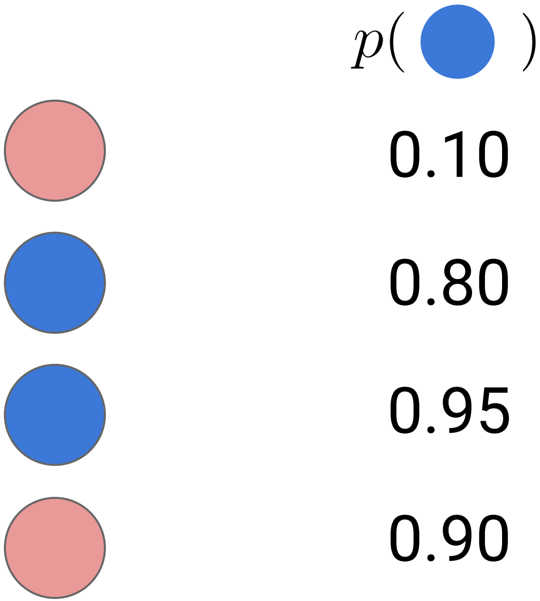
0.9*0.6*0.7*0.6*0.9*0.7 = 0.14

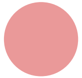
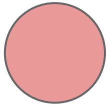
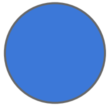
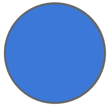
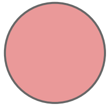
$$log(ab) = log(a) + log(b)$$

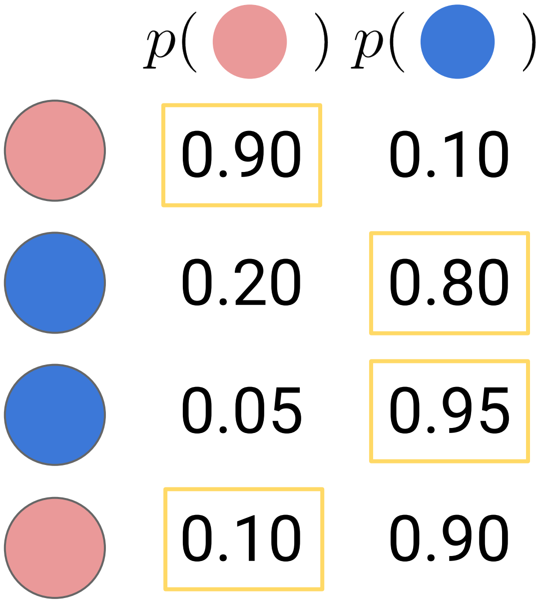
-log(0.6) - log(0.3) - log(0.9) -
-log(0.4) + log(0.6) - log(0.9) = 1.45

-log(0.9) - log(0.6) - log(0.7) -
-log(0.6) - log(0.9) - log(0.7) = 0.84

Чем **меньше**, тем **лучше**. **Задача минимизации ошибки**

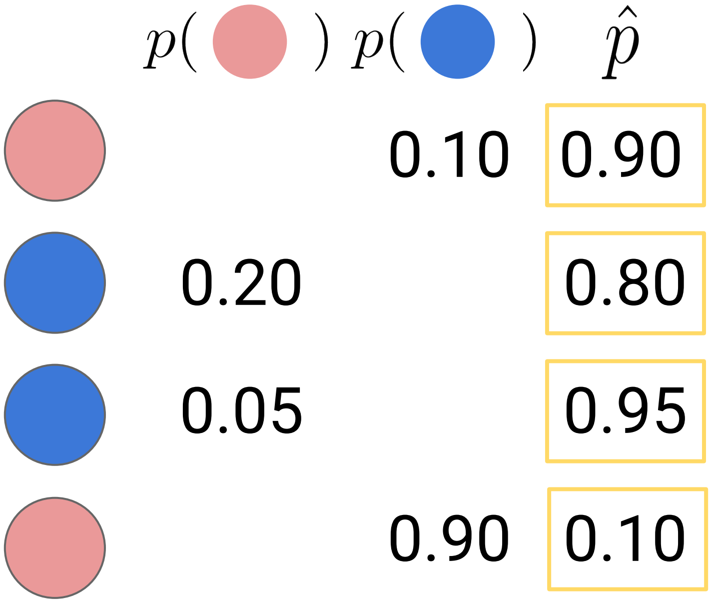

$p(\quad)$
0.10
0.80
0.95
0.90

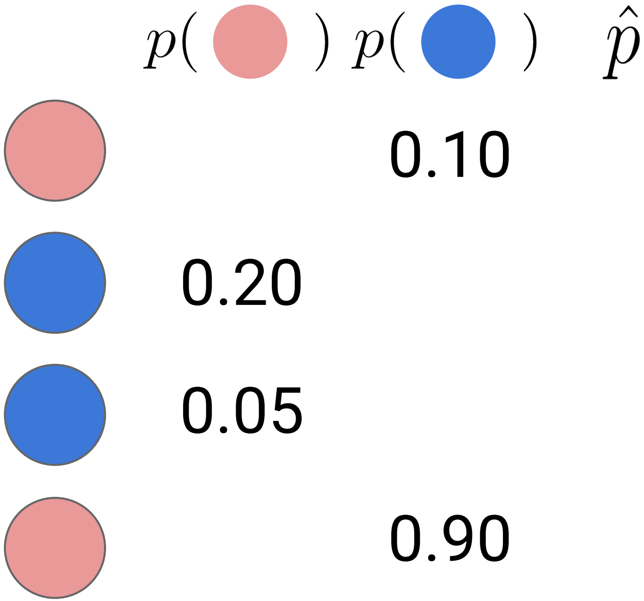
	$p($$)$	$p($$)$
	0.90	0.10
	0.20	0.80
	0.05	0.95
	0.10	0.90



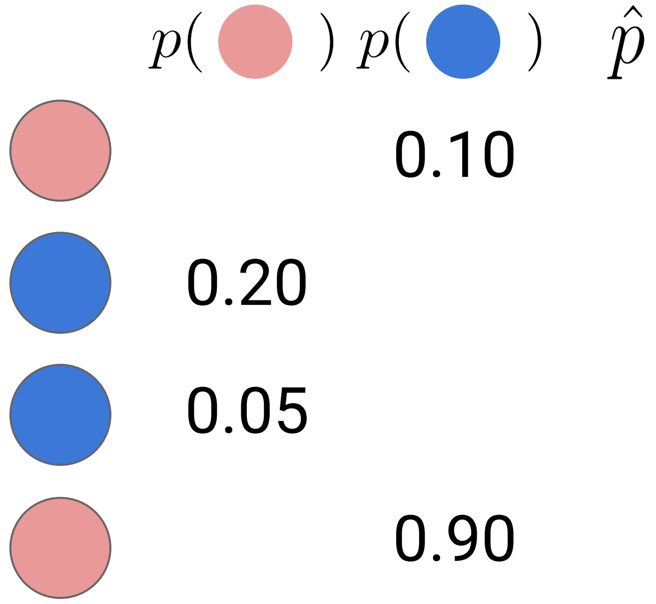
Выбираем вероятность правильного класса
Это **НЕ** обязательно максимум в строчке -- алгоритм мог ошибится

	$p(\ )$	$p(\ )$	$\hat{p}$
	$\boxed{0.90}$	0.10	
	0.20	$\boxed{0.80}$	
	0.05	$\boxed{0.95}$	
	$\boxed{0.10}$	0.90	


$p($ $)$ $p($ $)$ $\hat{p}$
0.10 0.90
0.20 0.80
0.05 0.95
0.90 0.10



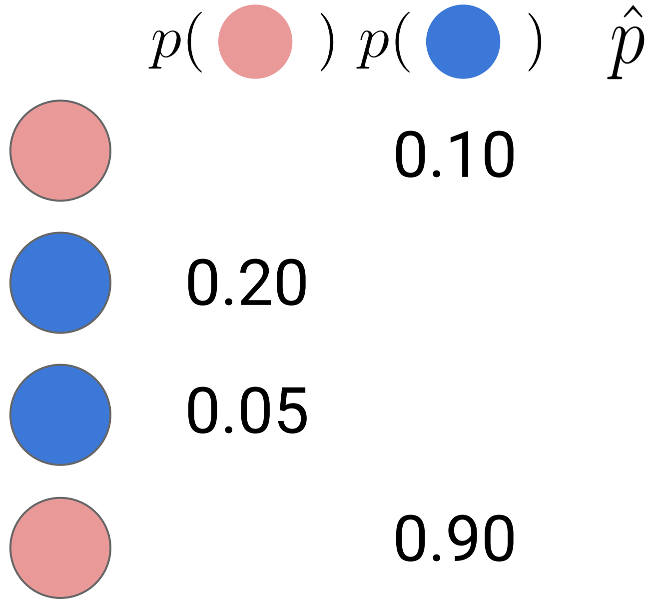
$$\boxed{0.90} * \boxed{0.80} * \boxed{0.95} * \boxed{0.10}$$



$$\boxed{0.90} * \boxed{0.80} * \boxed{0.95} * \boxed{0.10}$$

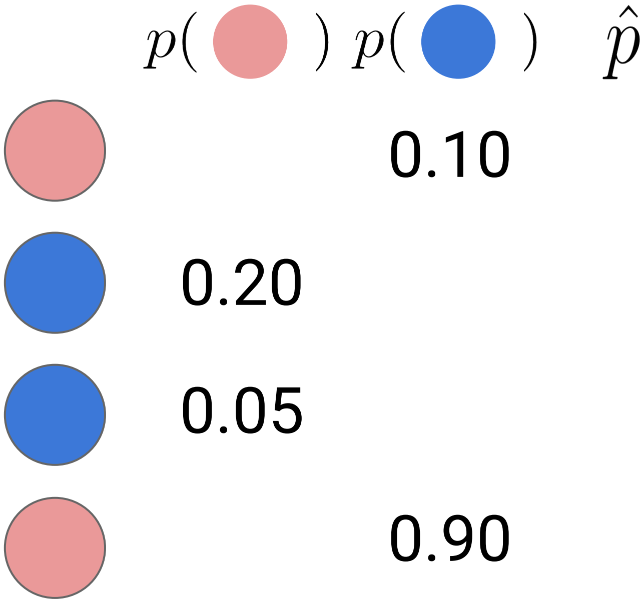
$$likelihood(\boldsymbol{p}, \boldsymbol{y}) = \prod_{i=1}^{N} \hat{p}(p_i, y_i),$$

$$\hat{p}(p_i, y_i) = \begin{cases} p_i & \text{если } y_i = 1 \\ 1 - p_i & \text{если } y_i = 0 \end{cases}$$



$$likelihood(\boldsymbol{p}, \boldsymbol{y}) = \prod_{i=1}^{N} \hat{p}(p_i, y_i),$$

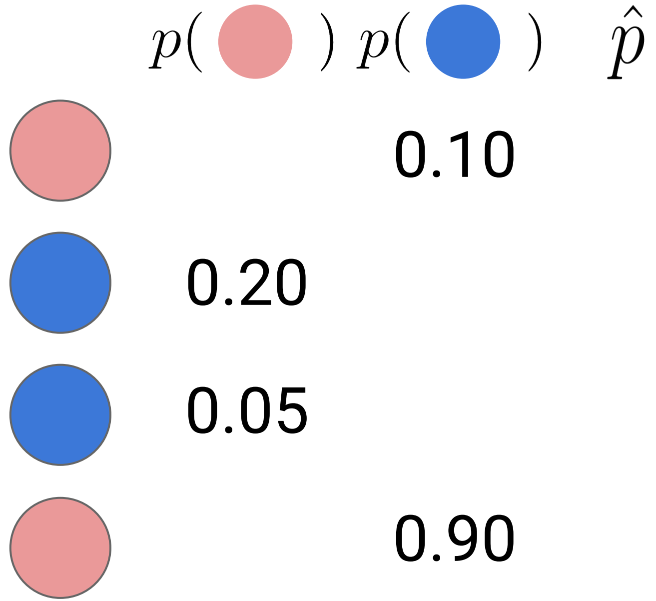
$$\hat{p}(p_i, y_i) = \begin{cases} p_i & \text{если } y_i = 1 \\ 1 - p_i & \text{если } y_i = 0 \end{cases}$$



$$likelihood(\boldsymbol{p}, \boldsymbol{y}) = \prod_{i=1}^{N} \hat{p}(p_i, y_i),$$

$$\hat{p}(p_i, y_i) = \begin{cases} p_i & \text{если } y_i = 1 \\ 1 - p_i & \text{если } y_i = 0 \end{cases}$$

$$log(ab) = log(a) + log(b)$$

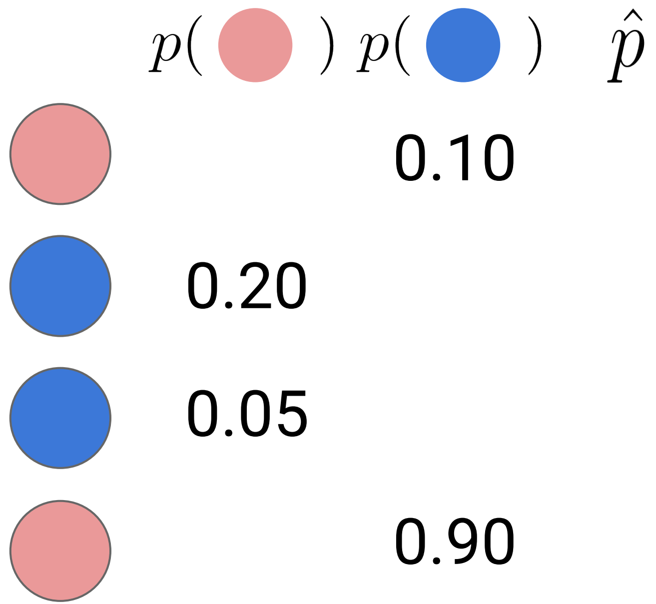


$$likelihood(\boldsymbol{p}, \boldsymbol{y}) = \prod_{i=1}^{N} \hat{p}(p_i, y_i),$$

$$\hat{p}(p_i, y_i) = \begin{cases} p_i & \text{если } y_i = 1 \\ 1 - p_i & \text{если } y_i = 0 \end{cases}$$

$$log(ab) = log(a) + log(b)$$

$$LogLikelihood(\boldsymbol{p}, \boldsymbol{y}) = \sum_{i=1}^{N} log(\hat{p}(p_i, y_i))$$



$$likelihood(\boldsymbol{p}, \boldsymbol{y}) = \prod_{i=1}^{N} \hat{p}(p_i, y_i),$$

$$\hat{p}(p_i, y_i) = \begin{cases} p_i & \text{если } y_i = 1 \\ 1 - p_i & \text{если } y_i = 0 \end{cases}$$

$log(ab) = log(a) + log(b)$

$$LogLikelihood(\boldsymbol{p}, \boldsymbol{y}) = \sum_{i=1}^{N} log(\hat{p}(p_i, y_i))$$

* -1 и среднее вместо суммы

$$NLL(\boldsymbol{p}, \boldsymbol{y}) = -\frac{1}{N} \sum_{i=1}^{N} log(\hat{p}(p_i, y_i))$$

Negative log likelihood (NLL)

- Negative log likelihood (NLL) -- функционал ошибки для задачи классификации

$$NLL(\boldsymbol{p}, \boldsymbol{y}) = -\frac{1}{N}\sum_{i=1}^{N} log(\hat{p}(p_i, y_i))$$

Negative log likelihood (NLL)

- Negative log likelihood (NLL) -- функционал ошибки для задачи классификации

$$NLL(\boldsymbol{p}, \boldsymbol{y}) = -\frac{1}{N}\sum_{i=1}^{N} log(\hat{p}(p_i, y_i))$$

- Aka:
  - Кросс энтропия
  - Бинарная кросс энтропия
  - Логлосс
- **Чем меньше ошибка, тем лучше наша модель**

Задача оптимизации

$$NLL(\boldsymbol{p}, \boldsymbol{y}) = -\frac{1}{N}\sum_{i=1}^{N} log(\hat{p}(p_i, y_i)) \rightarrow \min_{\text{model parameters}}$$

Задача оптимизации

$$NLL(\boldsymbol{p}, \boldsymbol{y}) = -\frac{1}{N}\sum_{i=1}^{N} log(\hat{p}(p_i, y_i)) \rightarrow \min_{\text{model parameters}}$$

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(\boldsymbol{x}_i, \boldsymbol{\theta})) \rightarrow \min_{\boldsymbol{\theta}}$$

Итог

- Мы поняли почему точность -- хороший показатель для нас, но не всегда полезный для модели
- Узнали, что такое принцип максимума правдоподобия
- Применив его построили функционал ошибки для нашего алгоритма
    - Предположив, что умеем предсказывать вероятности
- **Сформулировали задачу оптимизации**

Итог

- Мы поняли почему точность -- хороший показатель для нас, но не всегда полезный для модели
- Узнали, что такое принцип максимума правдоподобия
- Применив его построили функционал ошибки для нашего алгоритма
    - Предположив, что умеем предсказывать вероятности
- **Сформулировали задачу оптимизации**

В следующем уроке:

- А как же заставить модель предсказывать вероятности?

$$likelihood(p) = p^m (1 - p)^{n-m}$$

$$likelihood(p) = p^m(1-p)^{n-m}$$

$$LogLikelihood(p) = m\ln(p) + (n-m)\ln(1-p)$$

$$likelihood(p) = p^m (1-p)^{n-m}$$

$$LogLikelihood(p) = m\ln(p) + (n-m)\ln(1-p)$$

$$LogLikelihood(p)' = \frac{m}{p} - \frac{(n-m)}{(1-p)}$$

$$likelihood(p) = p^m (1-p)^{n-m}$$

$$LogLikelihood(p) = m\ln(p) + (n-m)\ln(1-p)$$

$$LogLikelihood(p)' = \frac{m}{p} - \frac{(n-m)}{(1-p)}$$

$$\frac{m}{p} = \frac{(n-m)}{(1-p)}$$

$$likelihood(p) = p^m (1-p)^{n-m}$$

$$LogLikelihood(p) = m\ln(p) + (n-m)\ln(1-p)$$

$$LogLikelihood(p)' = \frac{m}{p} - \frac{(n-m)}{(1-p)}$$

$$\frac{m}{p} = \frac{(n-m)}{(1-p)}$$

$$m(1-p) = p(n-m)$$

$$likelihood(p) = p^m (1-p)^{n-m}$$

$$LogLikelihood(p) = m\ln(p) + (n-m)\ln(1-p)$$

$$LogLikelihood(p)' = \frac{m}{p} - \frac{(n-m)}{(1-p)}$$

$$\frac{m}{p} = \frac{(n-m)}{(1-p)}$$

$$m(1-p) = p(n-m)$$

$$m - mp = pn - pm$$

$$likelihood(p) = p^m (1-p)^{n-m}$$

$$LogLikelihood(p) = m\ln(p) + (n-m)\ln(1-p)$$

$$LogLikelihood(p)' = \frac{m}{p} - \frac{(n-m)}{(1-p)}$$

$$\frac{m}{p} = \frac{(n-m)}{(1-p)}$$

$$m(1-p) = p(n-m)$$

$$m - mp = pn - pm$$

$$p = \frac{m}{n}$$

Skillbox

Введение в нейронные сети

Нейрон и логистическая регрессия

За прошлые два урока мы:

- Мы ввели модель линейного классификатора

$$I[ax_1 + bx_2 + cx_3 + \cdots + z > 0]$$

За прошлые два урока мы:

- Мы ввели модель линейного классификатора

$$I[ax_1 + bx_2 + cx_3 + \cdots + z > 0]$$

- Увидели, что можем получать из него “уверенность” -- чем дальше от прямой, тем классификатор более уверен в классе

$$ax_1 + bx_2 + cx_3 + \cdots + z$$

Skillbox

За прошлые два урока мы:

- Мы ввели модель линейного классификатора

$$I[ax_1 + bx_2 + cx_3 + \cdots + z > 0]$$

- Увидели, что можем получать из него “уверенность” -- чем дальше от прямой, тем классификатор более уверен в классе

$$ax_1 + bx_2 + cx_3 + \cdots + z$$

- Записали задачу оптимизации для подбора его коэффициентов

$$NLL(\boldsymbol{p}, \boldsymbol{y}) = -\frac{1}{N}\sum_{i=1}^{N} log(\hat{p}(p_i, y_i)) \to \min_{\text{model parameters}}$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = p_+$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = p_+$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

В чем тут проблема?

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \ 0.3$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \ 0.8$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \textbf{-34.8}$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \textbf{-34.8}$$

$$p_+ = p(y =$$

Значение

$$ax_1 + bx_2 + cx_3 \cdots + z$$

никак не ограничено!

А вероятность должна быть от 0 до 1!

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \textbf{?}$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \text{?}$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

$$Odds_+ = \frac{p_+}{1 - p_+}$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \textbf{?}$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

$$Odds_+ = \frac{p_+}{1 - p_+} \in [0, \inf)$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \textbf{?}$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

$$Odds_+ = \frac{p_+}{1 - p_+} \in [0, \inf)$$

$$logit_+ = \ln \frac{p_+}{1 - p_+}$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \textbf{?}$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

$$Odds_+ = \frac{p_+}{1 - p_+} \in [0, \inf)$$

$$logit_+ = \ln \frac{p_+}{1 - p_+} \in (-\inf, \inf)$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \; logit_+$$

$$p_+ = p(y = 1, \boldsymbol{x})$$

$$Odds_+ = \frac{p_+}{1 - p_+} \in [0, \inf)$$

$$logit_+ = \ln \frac{p_+}{1 - p_+} \in (-\inf, \inf)$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \ logit_+$$

$$logit_+ = \ln \frac{p_+}{1 - p_+} \in (-\inf, \inf)$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \; logit_+$$

$$logit_+ = \ln \frac{p_+}{1 - p_+} \in (-\inf, \inf) \cdots\cdots\cdots\cdots\cdots\rightarrow \; p_+ = \frac{1}{1 + e^{-logit_+}}$$

Как предсказать вероятность?

$$ax_1 + bx_2 + cx_3 \cdots + z = \; logit_+$$

$$logit_+ = \ln \frac{p_+}{1 - p_+} \in (-\inf, \inf) \quad \cdots\cdots\cdots\cdots\cdots\cdots\blacktriangleright \quad p_+ = \frac{1}{1 + e^{-logit_+}}$$

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

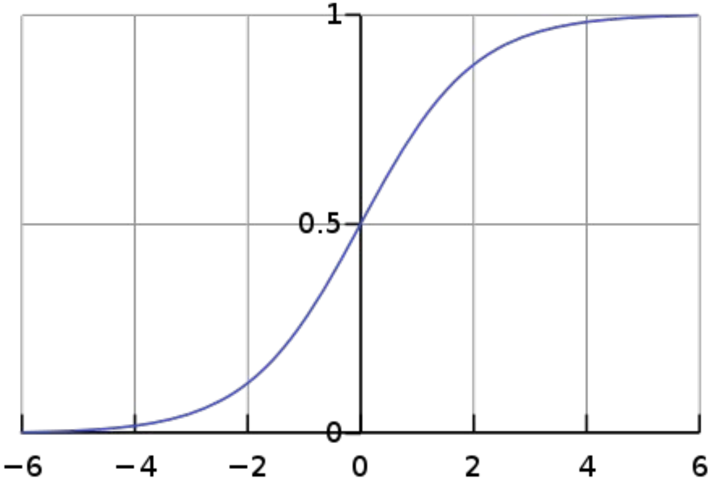
Как предсказать вероятность?

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

$$logit_+ = \ln \frac{p_+}{1 - p_+} \in (-\inf, \inf) \cdots\cdots\cdots\cdots\cdots\blacktriangleright \quad p_+ = \frac{1}{1 + e^{-logit_+}}$$

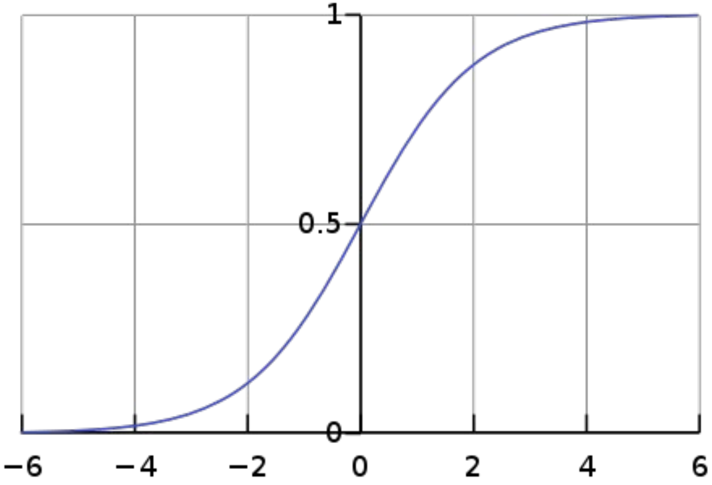
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$ -- сигмоида (sigmoid)

Как предсказать вероятность?



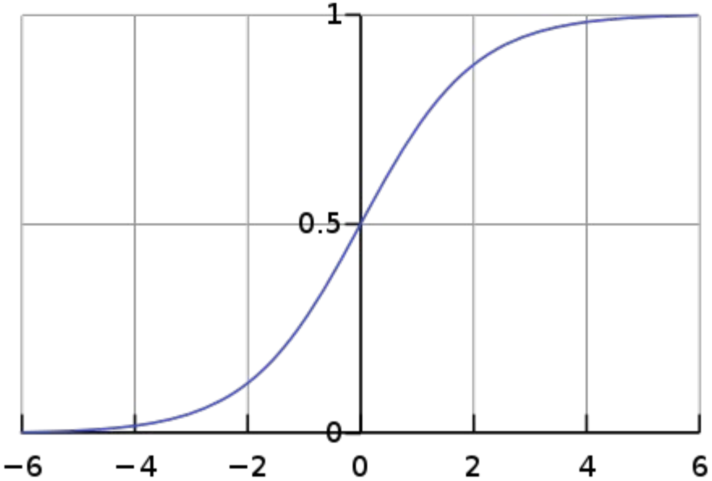
$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

Как предсказать вероятность?



$$\sigma\left(\boxed{ax_1 + bx_2 + cx_3 \cdots + z}\right) = p_+$$

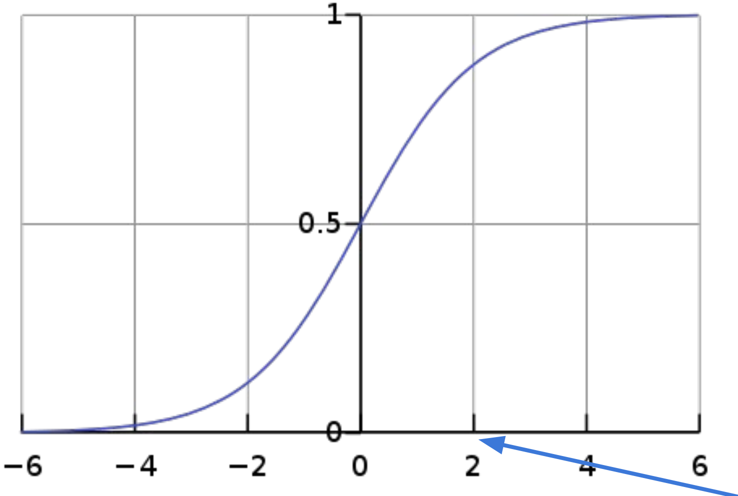
Как предсказать вероятность?



$$\sigma\Big(\underbrace{\boxed{ax_1 + bx_2 + cx_3 \cdots + z}}_{}\Big) = p_+$$

=2

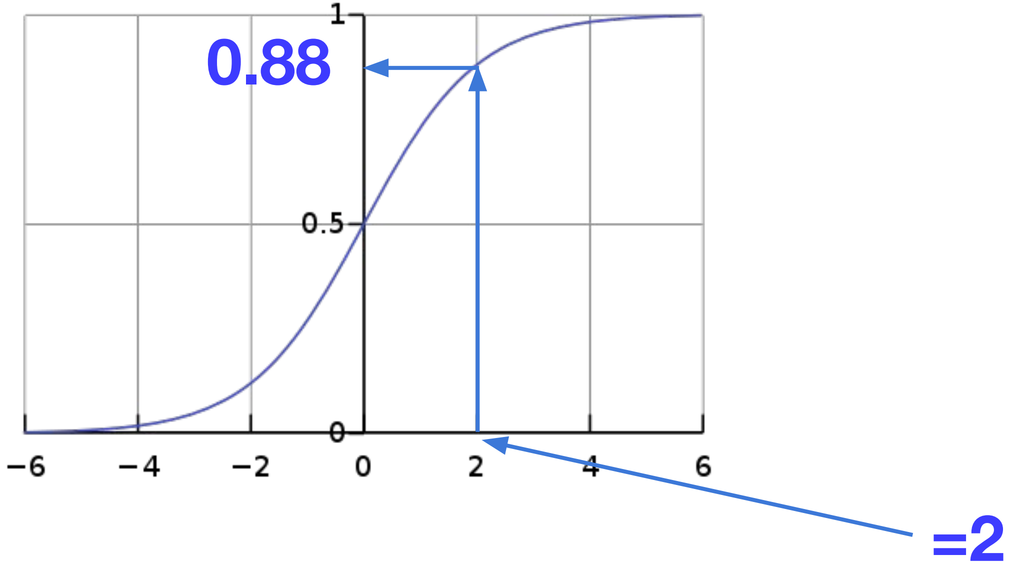
Как предсказать вероятность?



=2

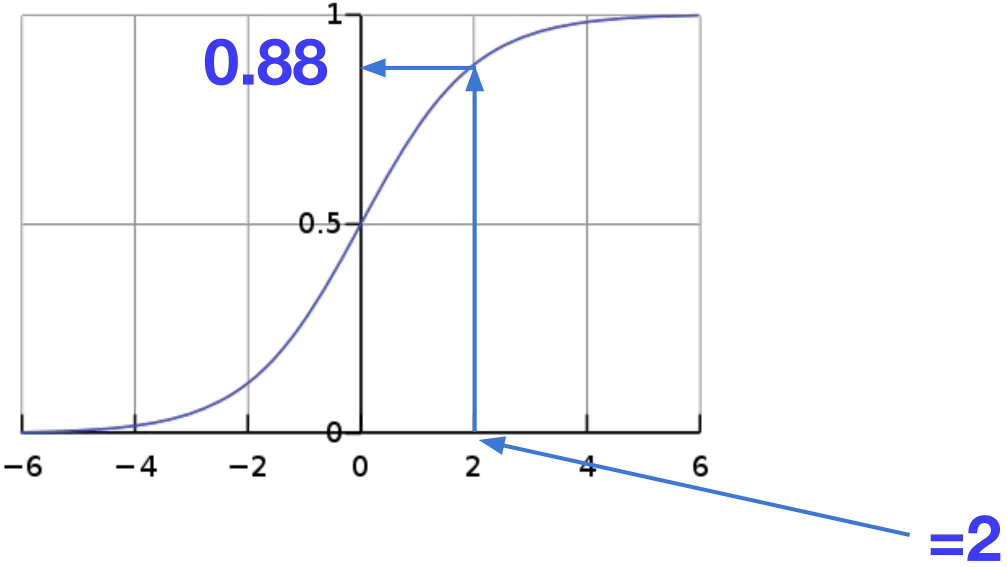
$$\sigma\left(\boxed{ax_1 + bx_2 + cx_3 \cdots + z}\right) = p_+$$

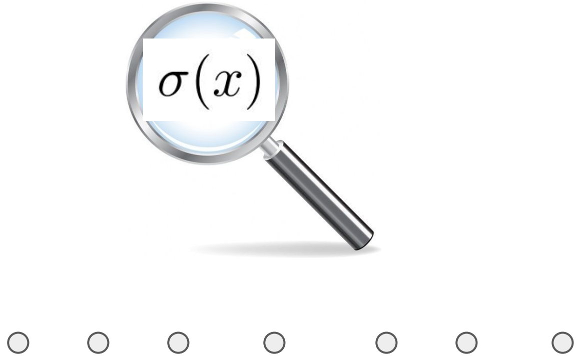
Как предсказать вероятность?



$$\sigma(\boxed{ax_1 + bx_2 + cx_3 \cdots + z}) = p_+$$

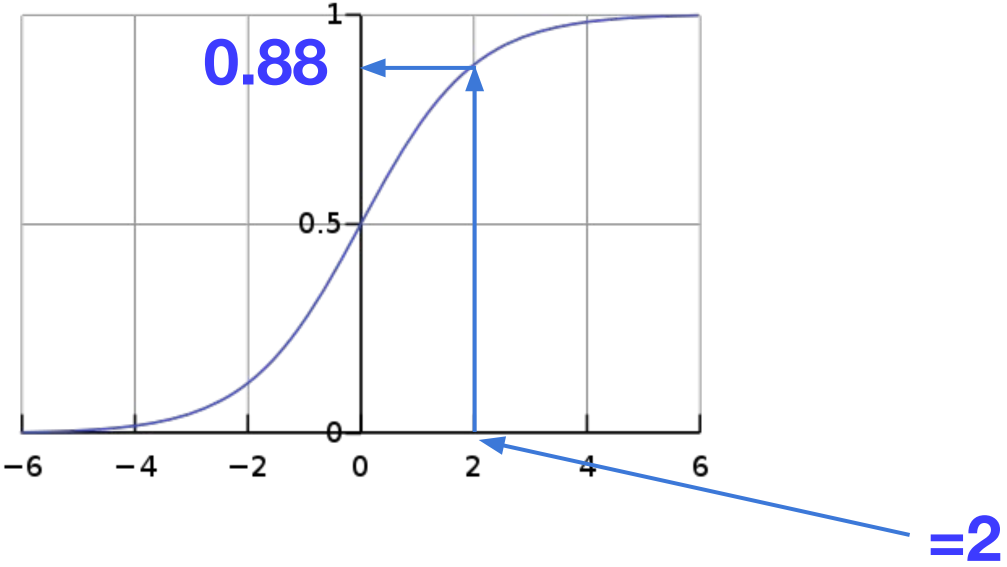
Как предсказать вероятность?

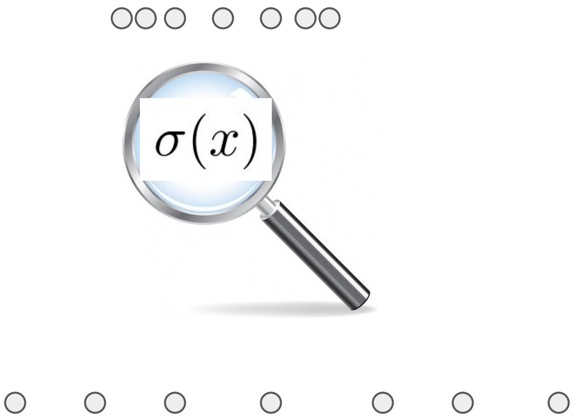




$$\sigma\left(\boxed{ax_1 + bx_2 + cx_3 \cdots + z}\right) = p_+$$

Как предсказать вероятность?





$$\sigma\left(\boxed{ax_1 + bx_2 + cx_3 \cdots + z}\right) = p_+$$

Логистическая регрессия

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

- Модель логистической регрессии
  - Хоть и регрессия -- это задача классификации (не путайтесь!)
  - Т.е. задача предсказания класса, а не числа (как в обычной регрессии)

Логистическая регрессия

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

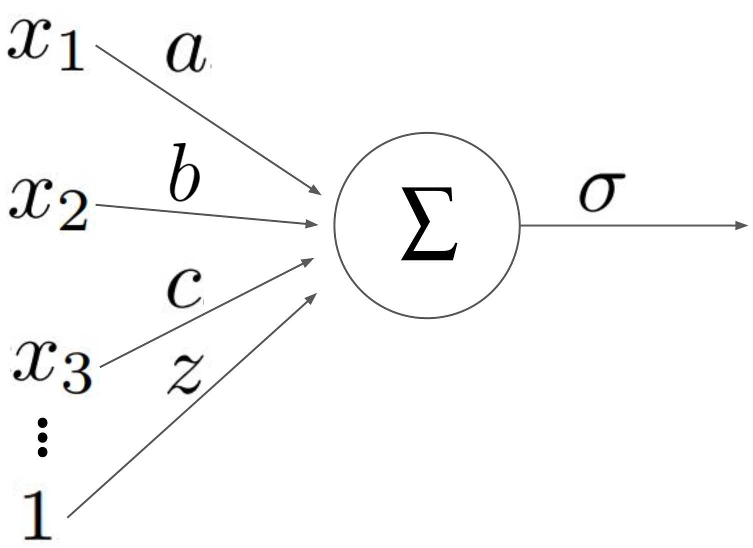
- Модель логистической регрессии
  - Хоть и регрессия -- это задача классификации (не путайтесь!)
  - Т.е. задача предсказания класса, а не числа (как в обычной регрессии)
- А при чем здесь нейрон?

Нейрон

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

Нейрон

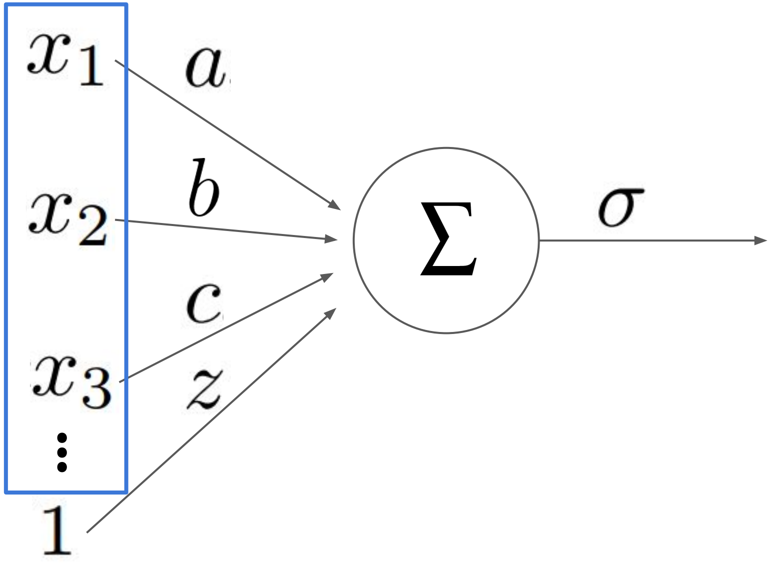
$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$



Нейрон

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

Вход нейрона (признаки, inputs)

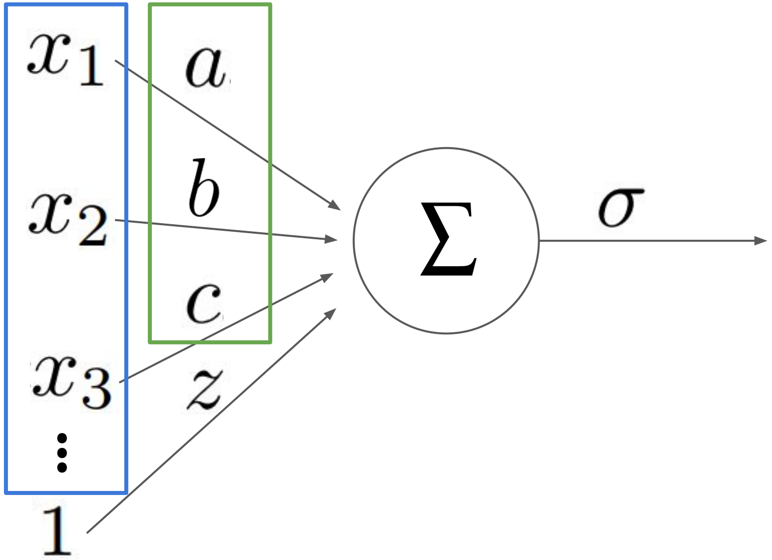


Нейрон

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)



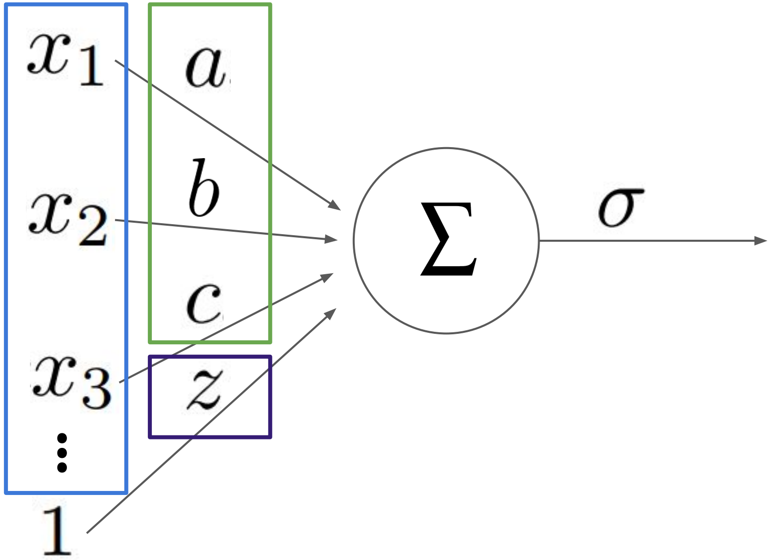
Нейрон

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)



Нейрон

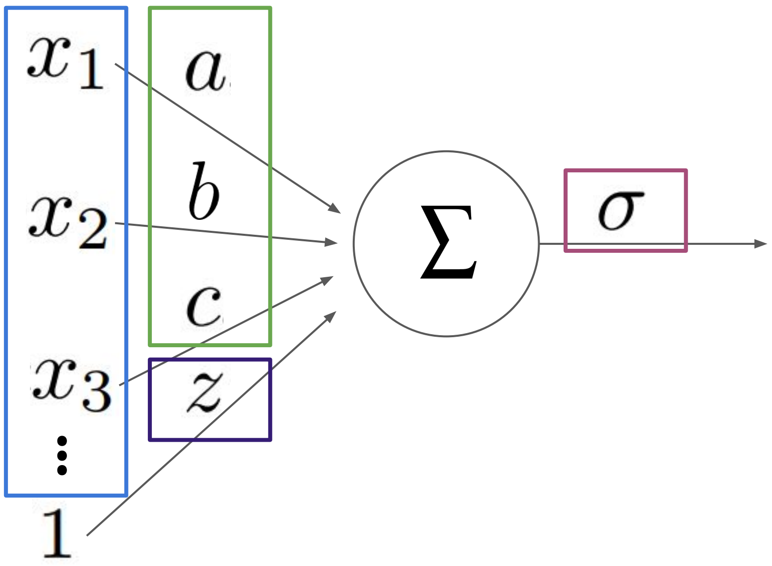
$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)

Функция активации



Нейрон

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

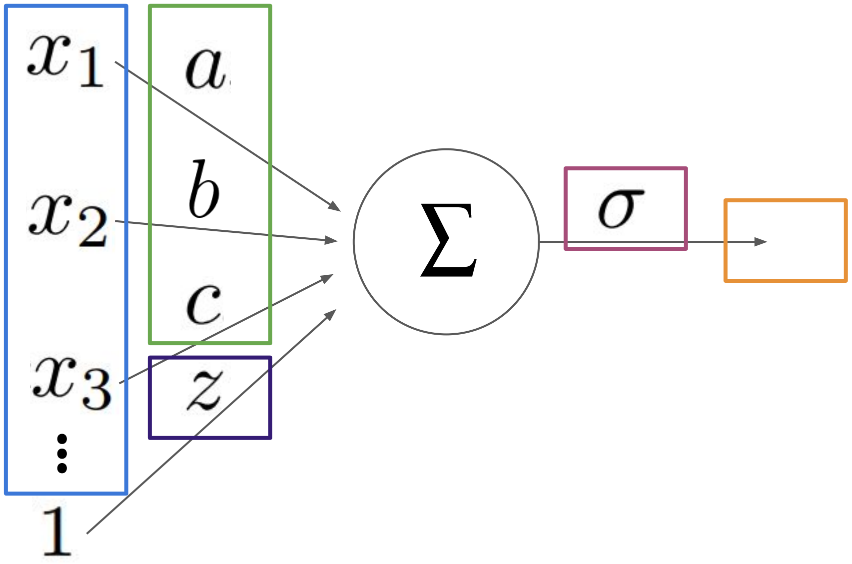
Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)

Функция активации

Выход нейрона



Нейрон

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

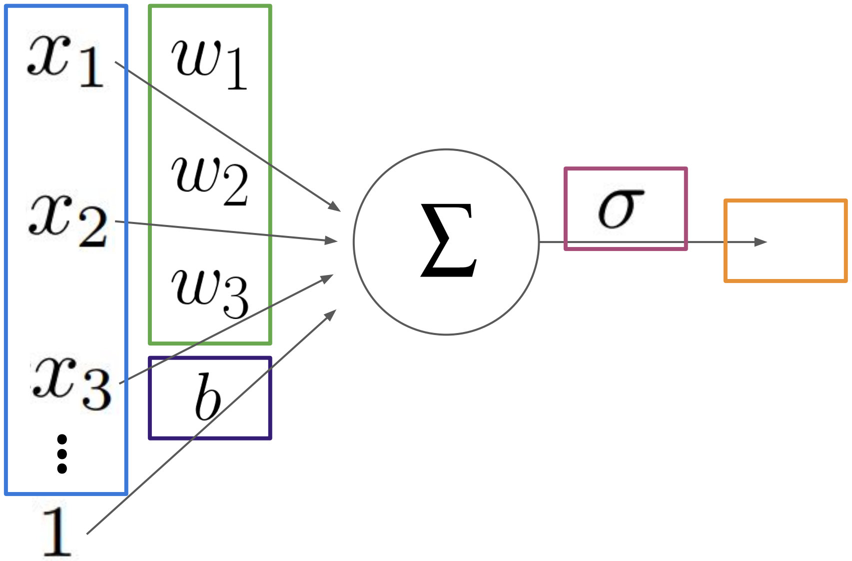
Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)

Функция активации

Выход нейрона



Нейрон

$$\sigma(ax_1 + bx_2 + cx_3 \cdots + z) = p_+$$

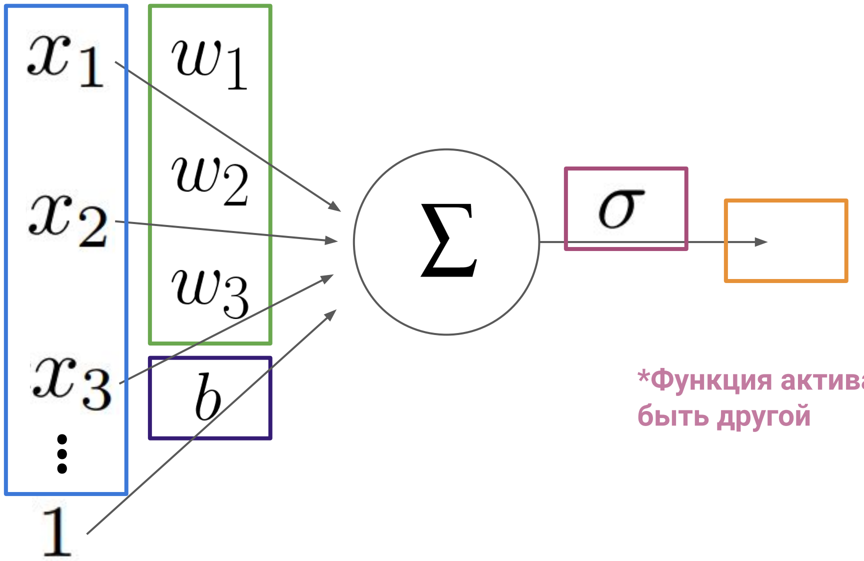
Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)

Функция активации

Выход нейрона



*Функция активации может быть другой

Skillbox

Нейрон

$$\sigma(\boldsymbol{wx} + b) = p_+$$

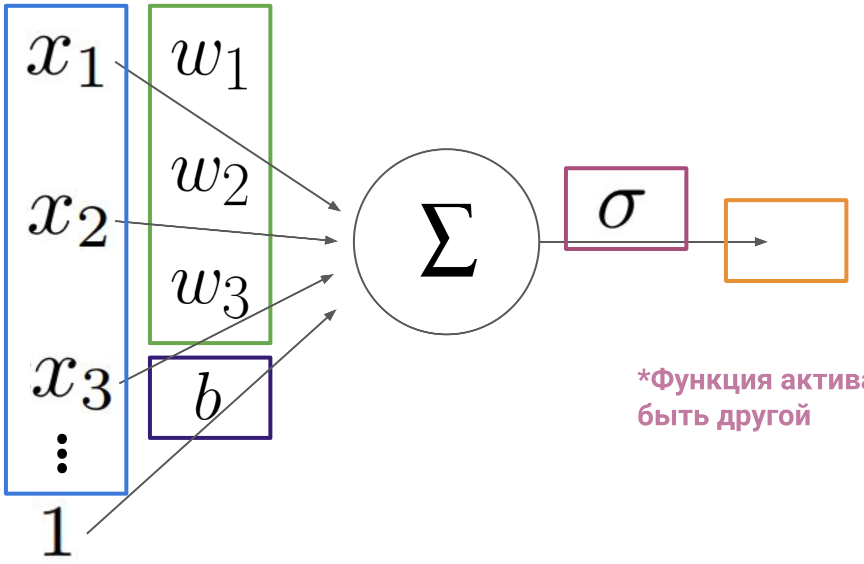
Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)

Функция активации

Выход нейрона



*Функция активации может быть другой

Итог

- Мы поняли, как заставить наш классификатор предсказывать вероятность
  - Именно это мы от него требовали в прошлом уроке, чтобы получить функцию потерь!
  - Для этого нам пригодилась сигмоида

Итог

- Мы поняли, как заставить наш классификатор предсказывать вероятность
    - Именно это мы от него требовали в прошлом уроке, чтобы получить функцию потерь!
    - Для этого нам пригодилась сигмоида
- Мы вспомнили, что такое логистическая регрессия

Итог

- Мы поняли, как заставить наш классификатор предсказывать вероятность
    - Именно это мы от него требовали в прошлом уроке, чтобы получить функцию потерь!
    - Для этого нам пригодилась сигмоида
- Мы вспомнили, что такое логистическая регрессия
- Поняли в чем ее отличие от нейрона:
    - Отличий нет, если функция активации -- сигмоида

Skillbox

Нейрон

$$\sigma(\boldsymbol{wx} + b) = p_+$$

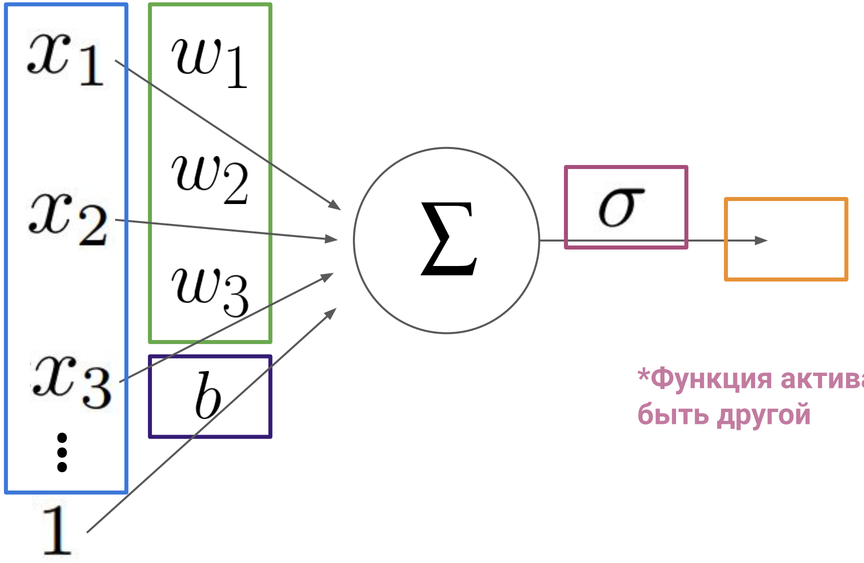
Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)

Функция активации

Выход нейрона

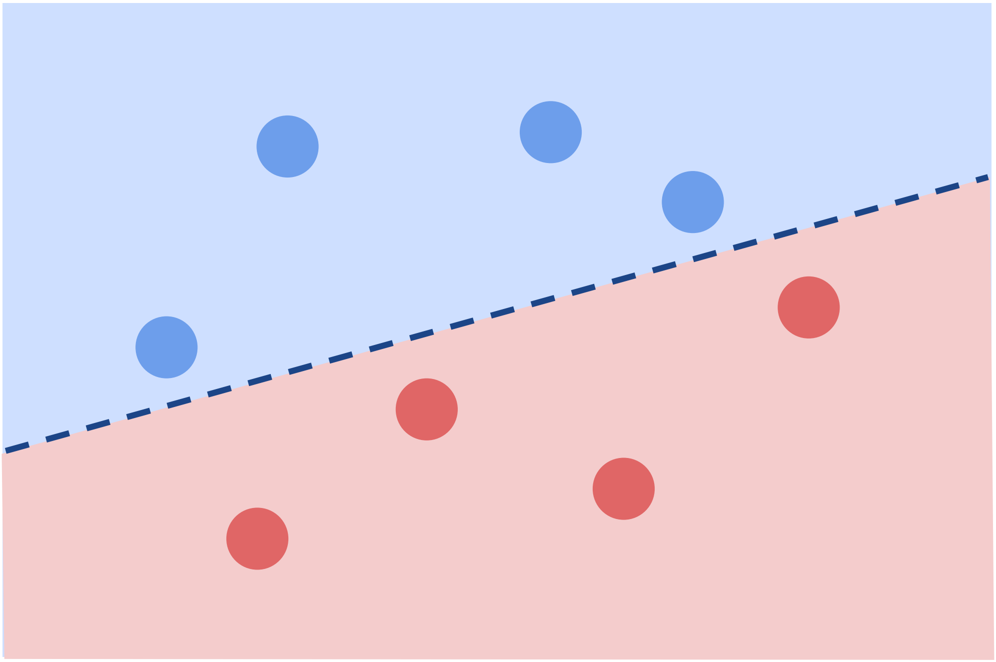


*Функция активации может быть другой

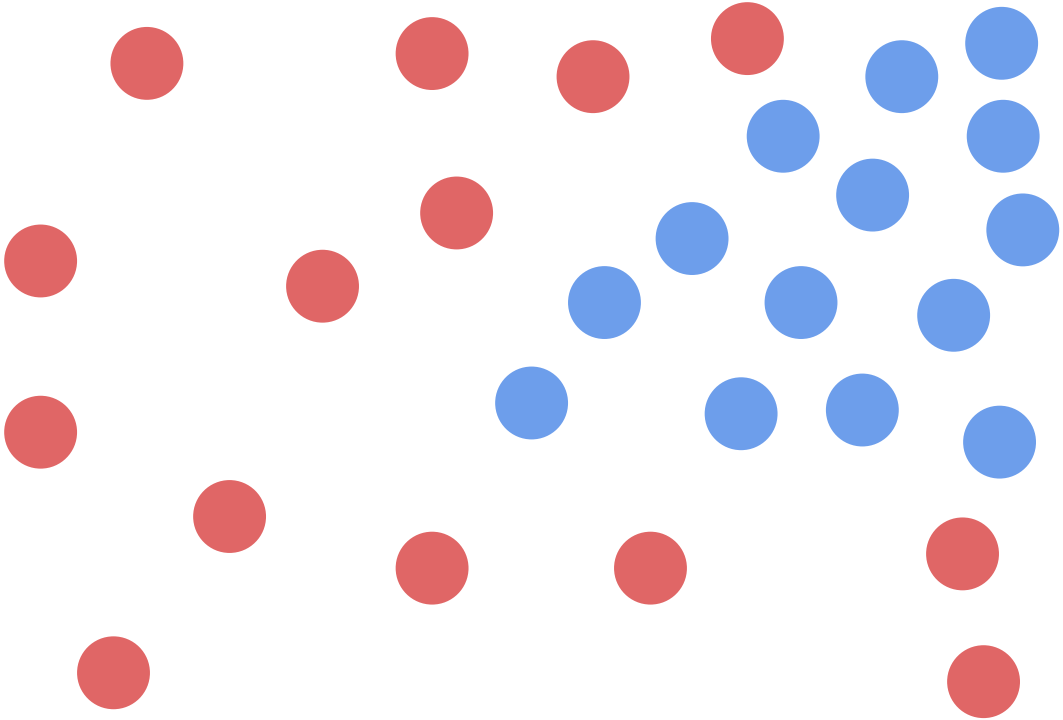
Skillbox

Введение в нейронные сети

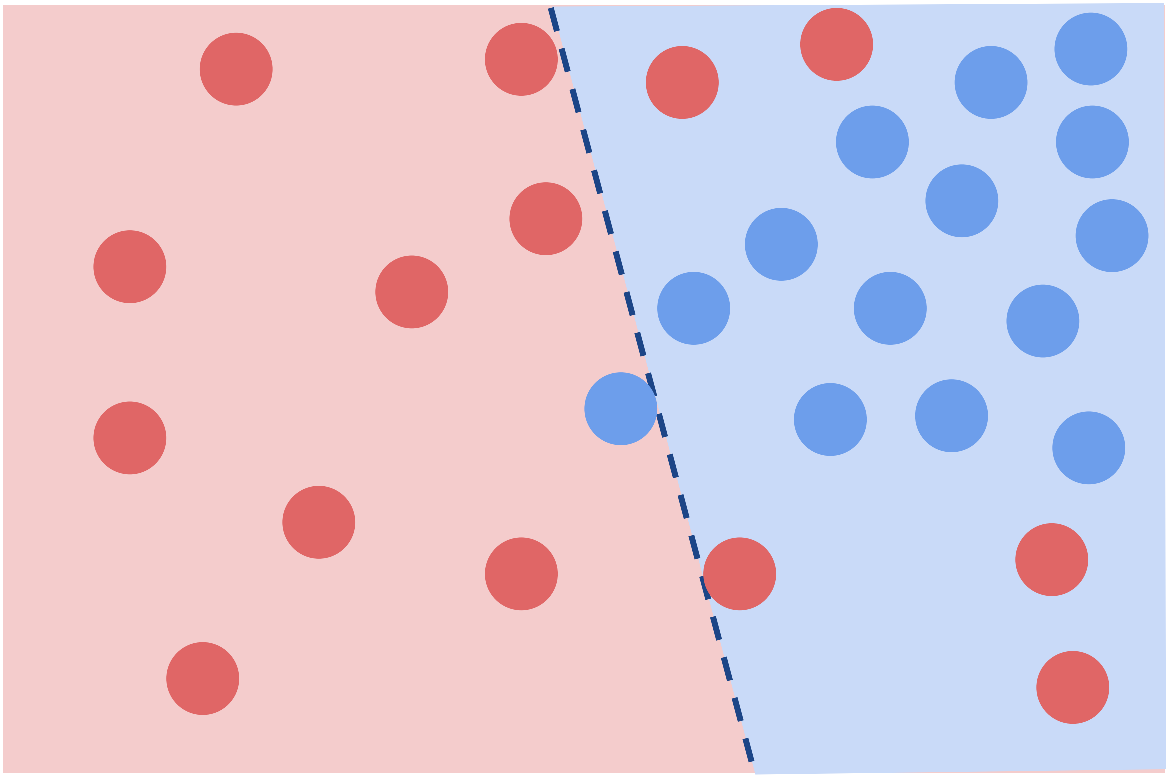
Нейронная сеть

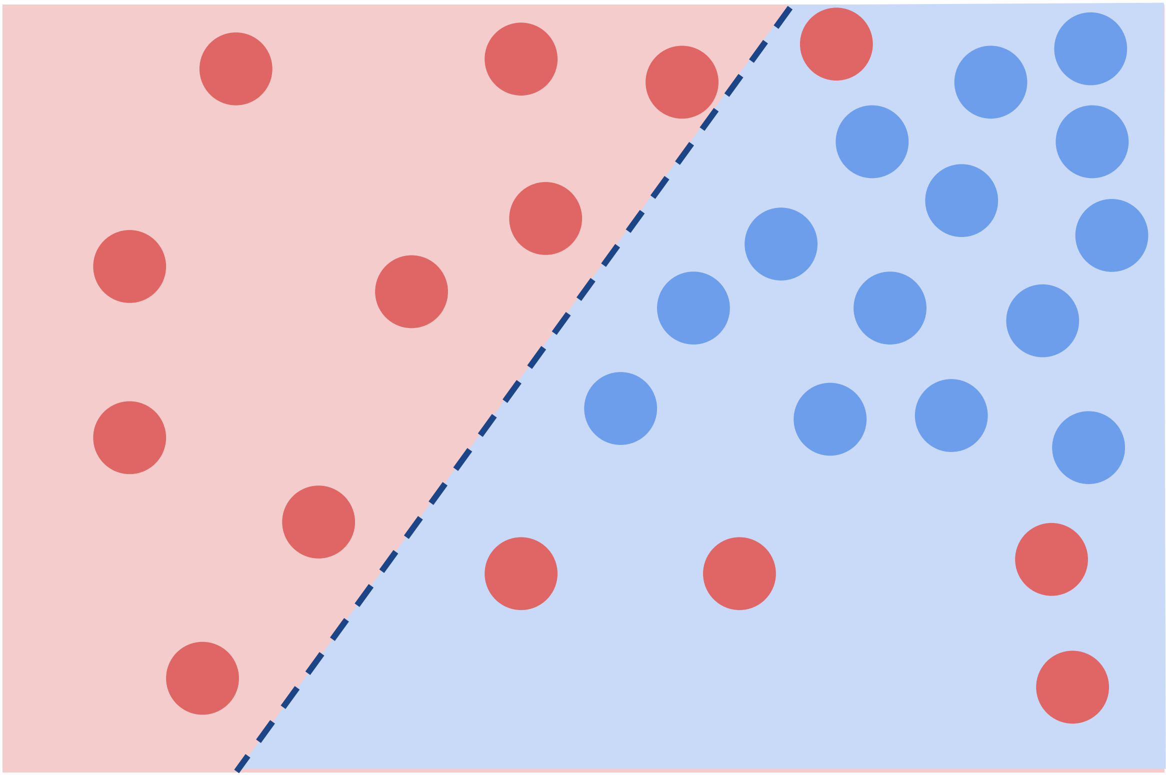


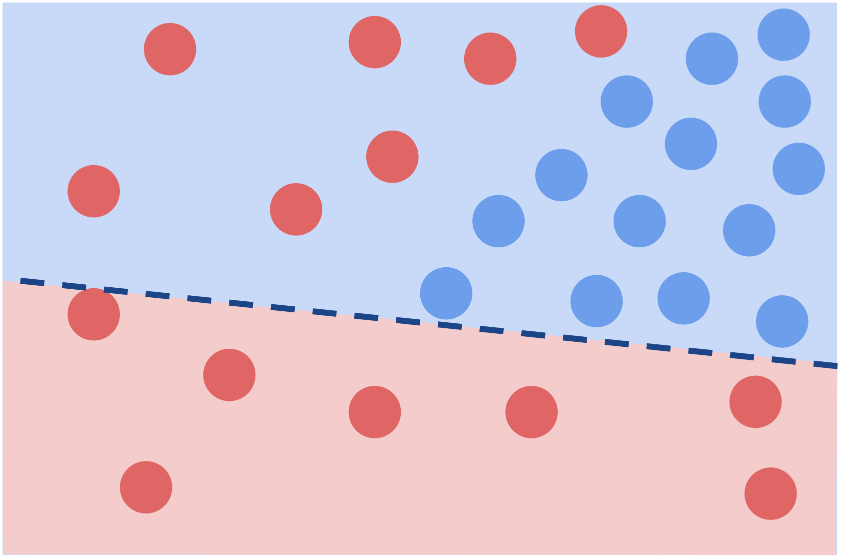
Как разделить одним нейроном?



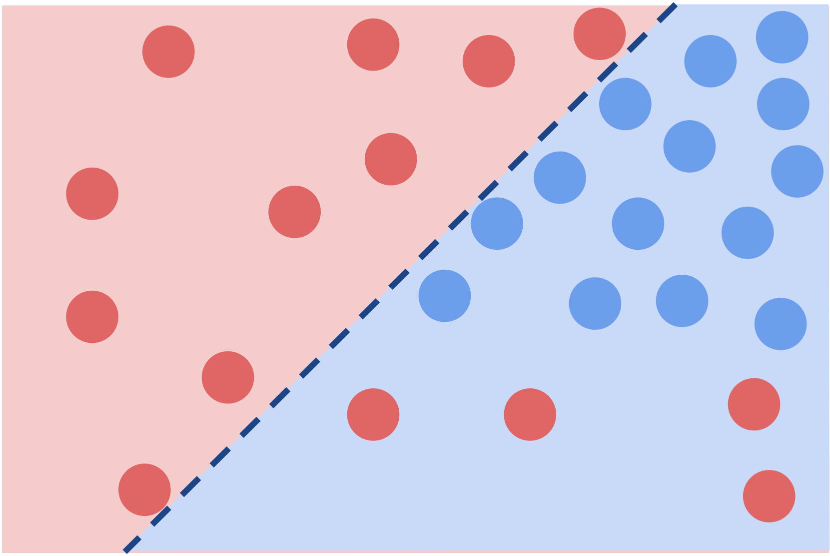
Skillbox



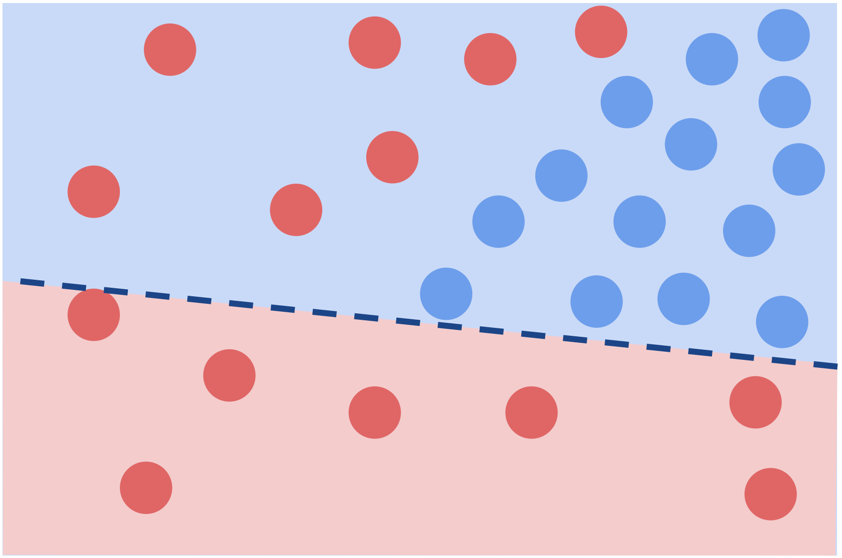




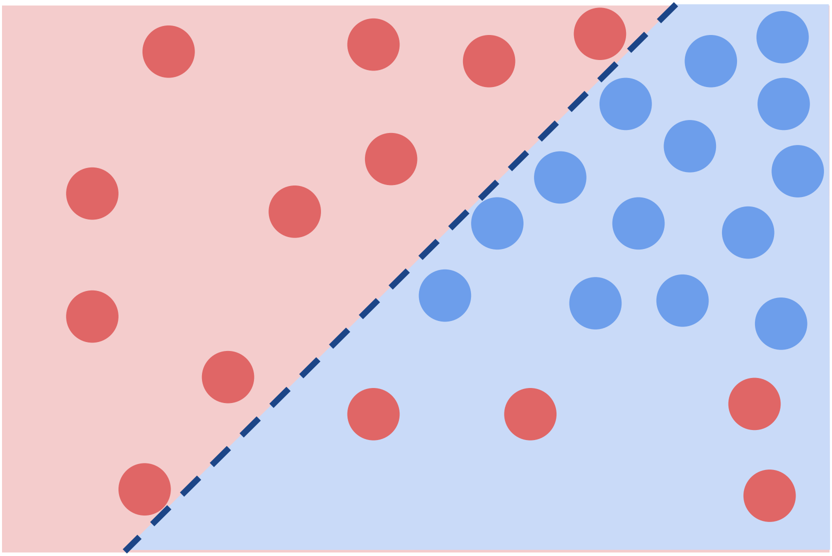
Нейрон 1



Нейрон 2

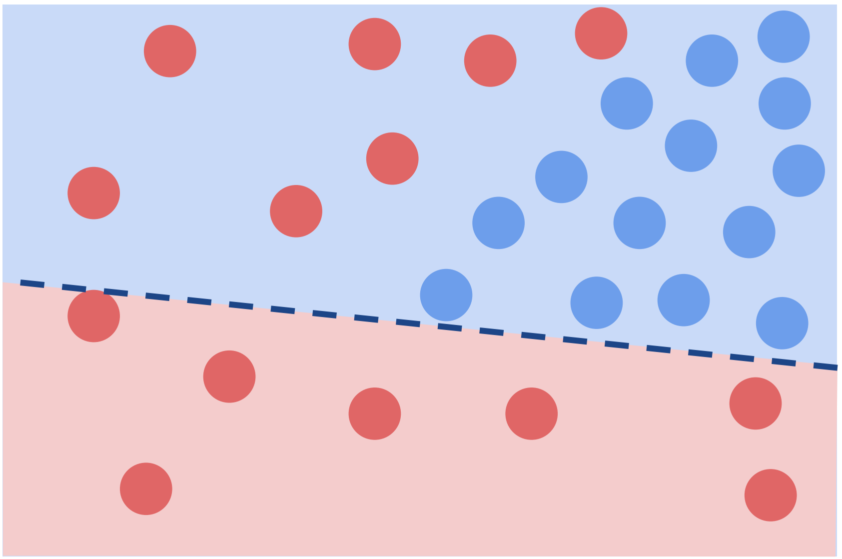


Нейрон 1

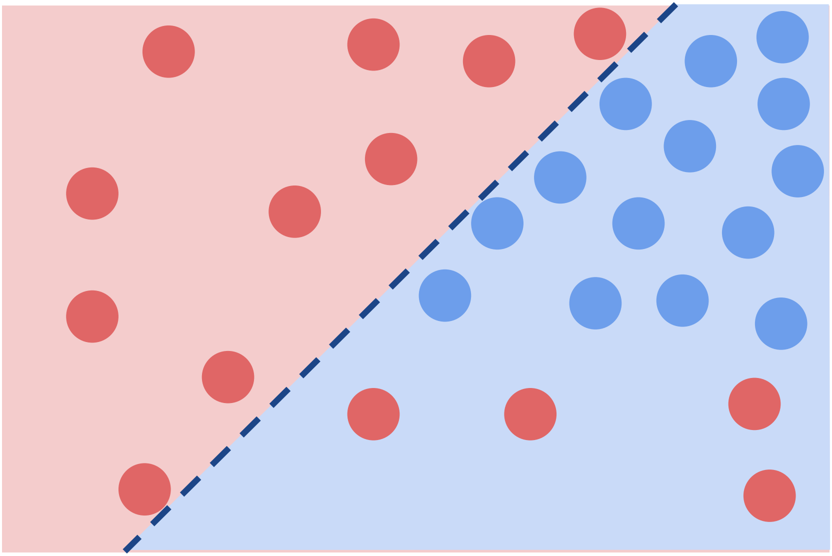


Нейрон 2

Правило:
Если первый И второй нейроны предсказали, что объект синий -- значит он синий



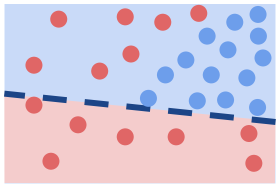
Нейрон 1



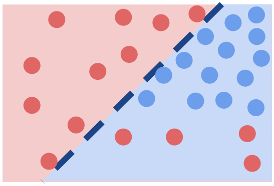
Нейрон 2

Правило:
Если первый И второй нейроны предсказали, что объект синий -- значит он синий

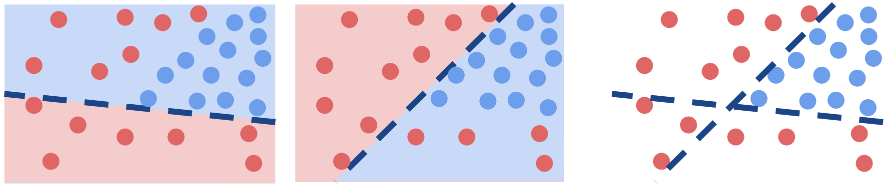
Такое правило, конечно, сработает, но как его обобщить?



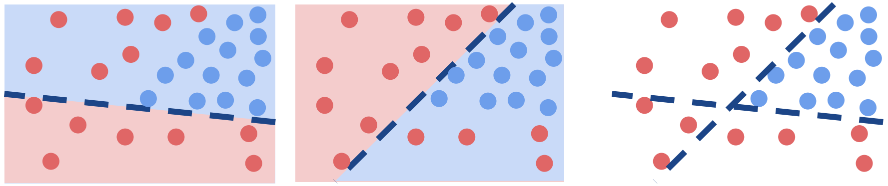
Нейрон 1



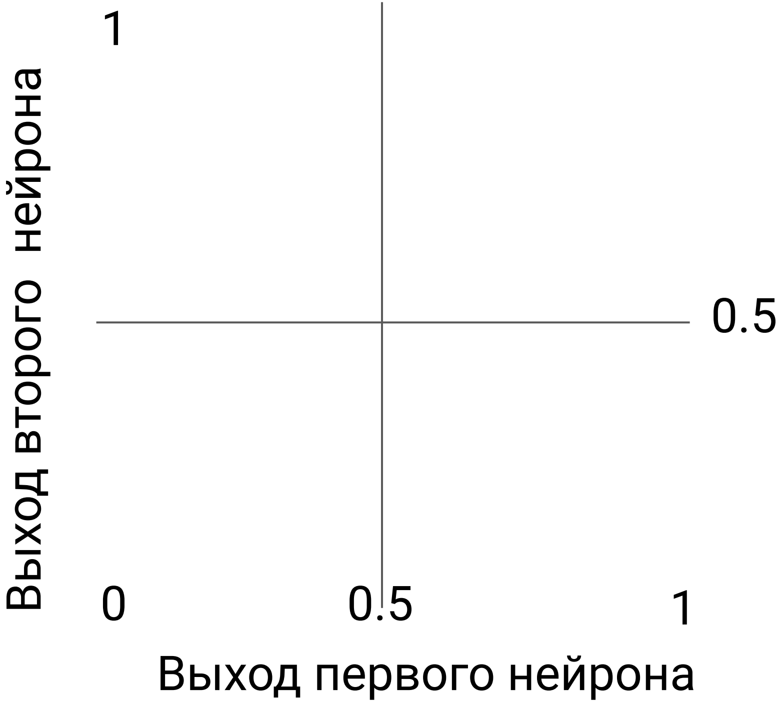
Нейрон 2

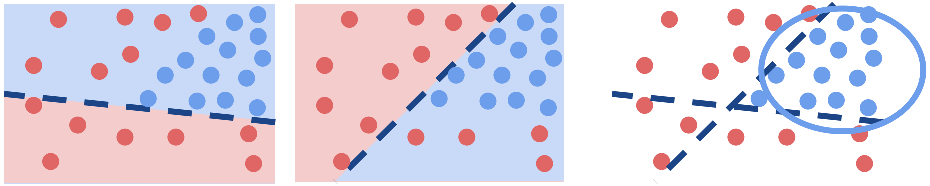


Нейрон 1 Нейрон 2

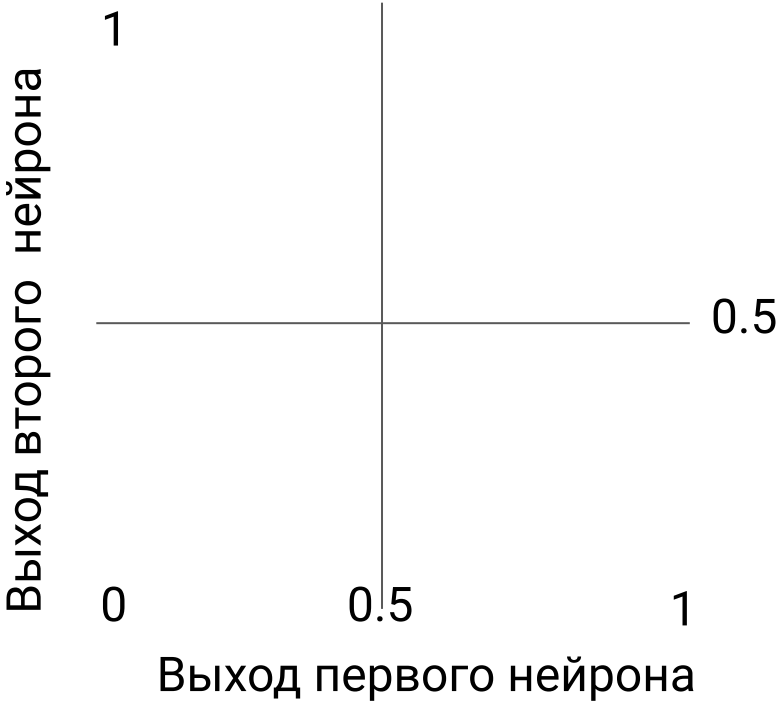


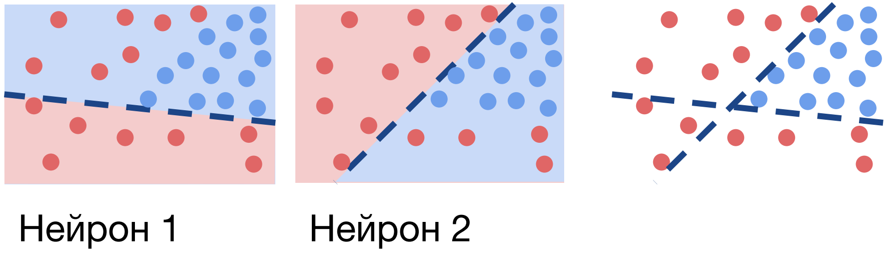
Нейрон 1 Нейрон 2

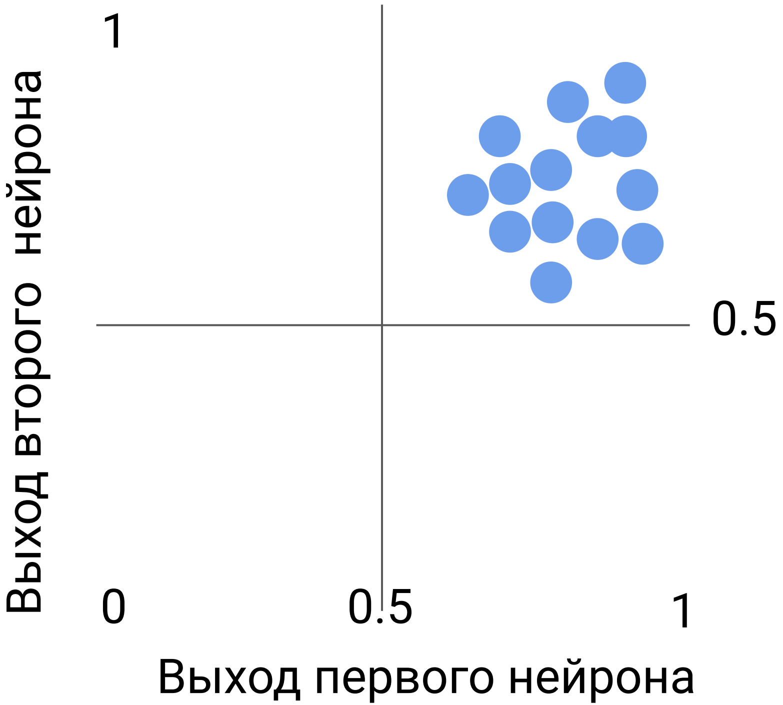


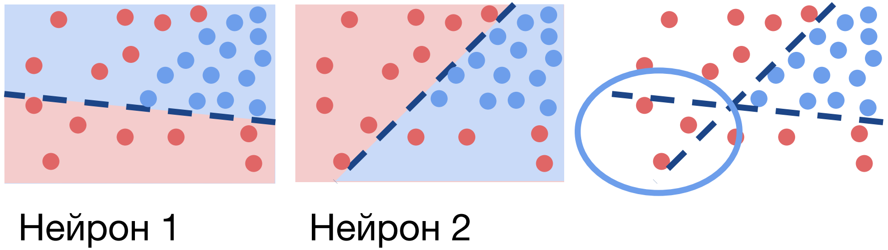


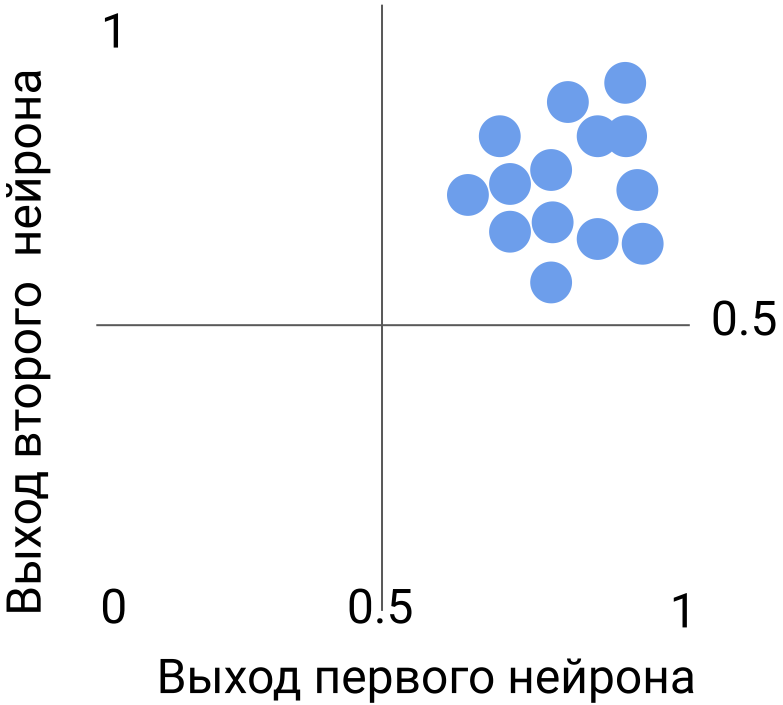
Нейрон 1 Нейрон 2

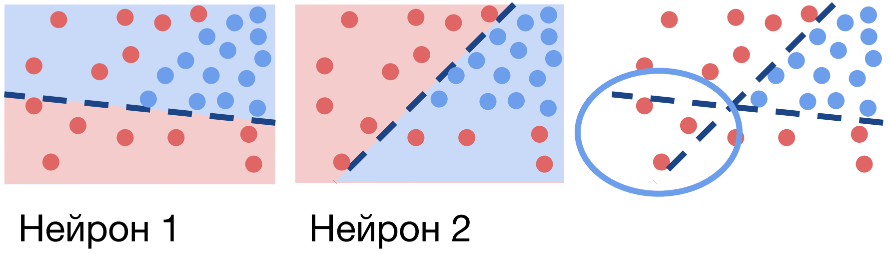


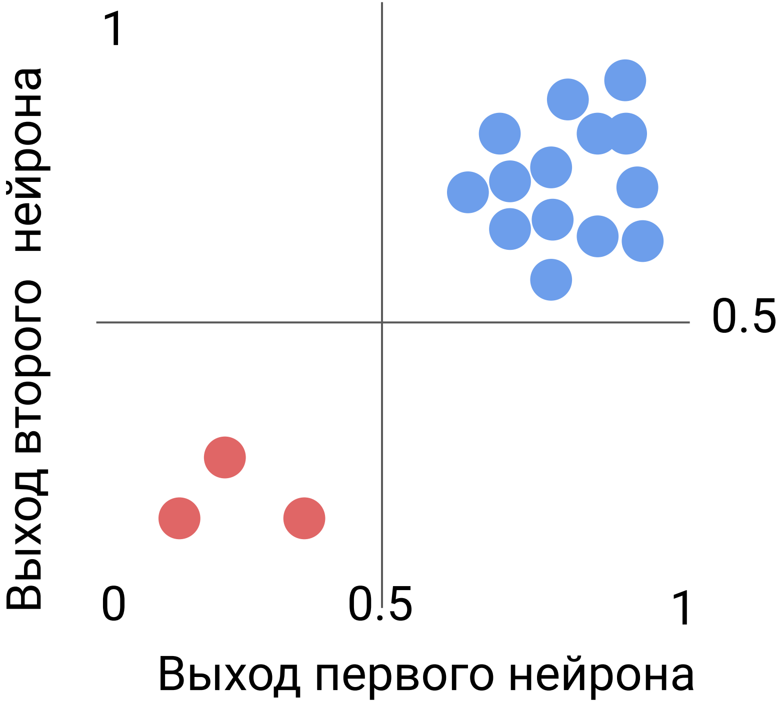

Нейрон 1
Нейрон 2

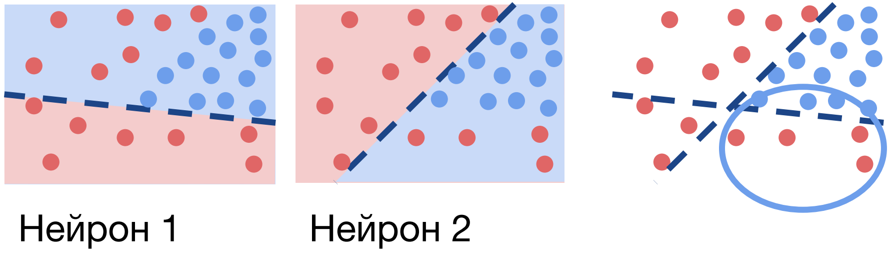

1
0.5
0
0.5
1
Выход второго нейрона
Выход первого нейрона

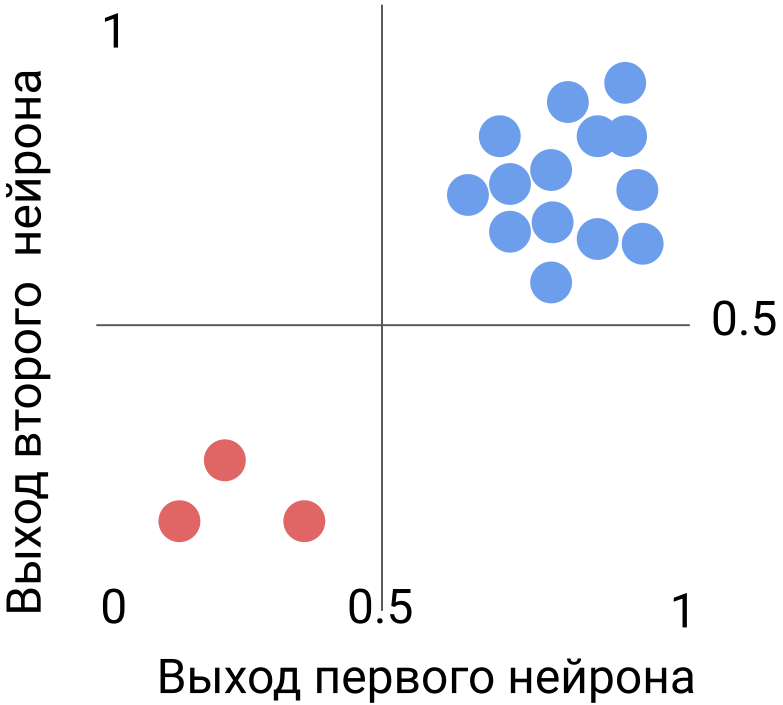

Нейрон 1
Нейрон 2

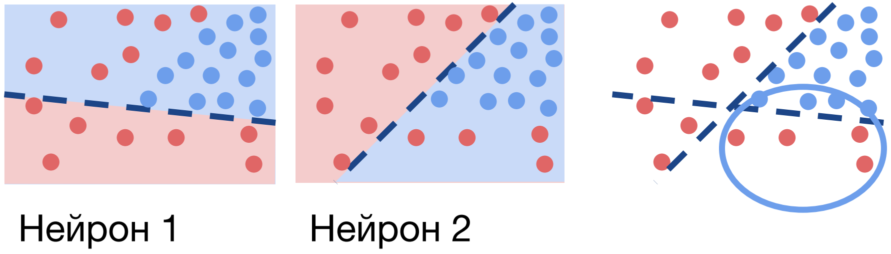

1
0.5
0
0.5
1
Выход второго нейрона
Выход первого нейрона

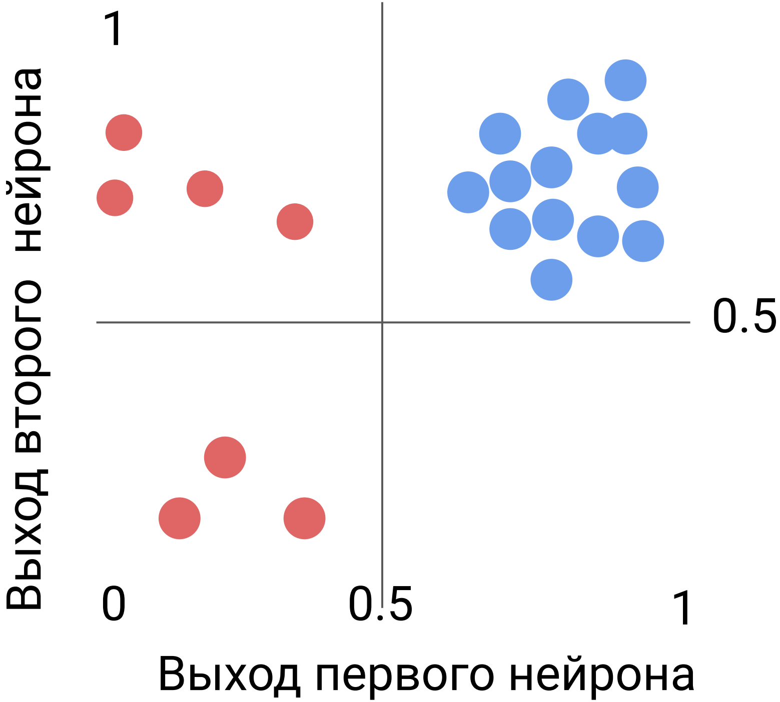

Нейрон 1
Нейрон 2

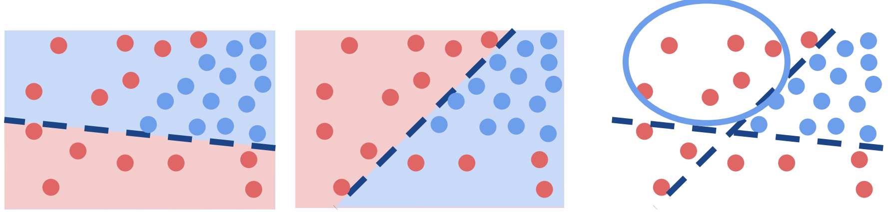

1
0.5
0
0.5
1
Выход второго нейрона
Выход первого нейрона


Нейрон 1
Нейрон 2

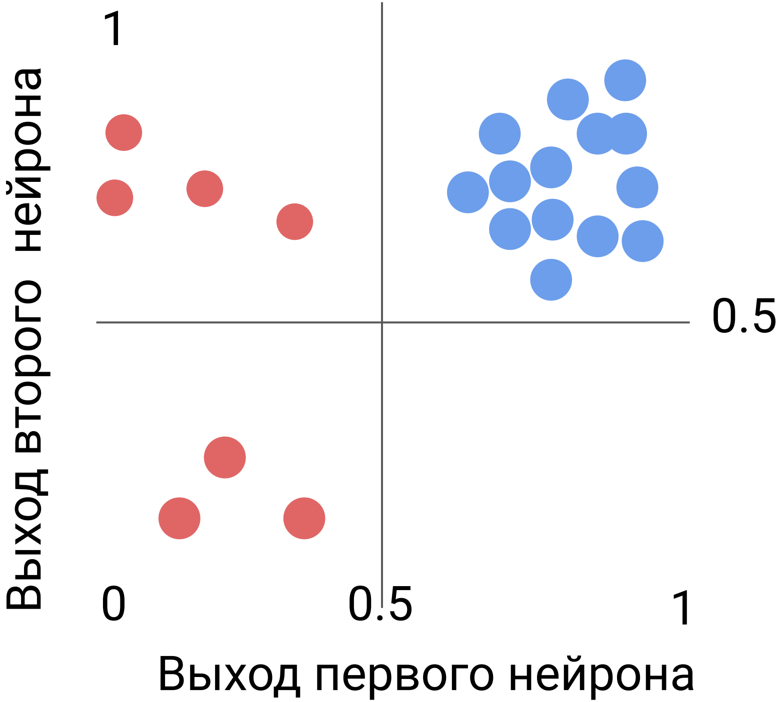

1
0.5
0
0.5
1
Выход второго нейрона
Выход первого нейрона

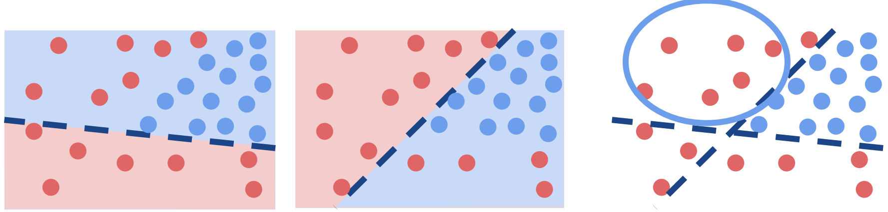

Нейрон 1
Нейрон 2


1
0.5
0
0.5
1
Выход второго нейрона
Выход первого нейрона

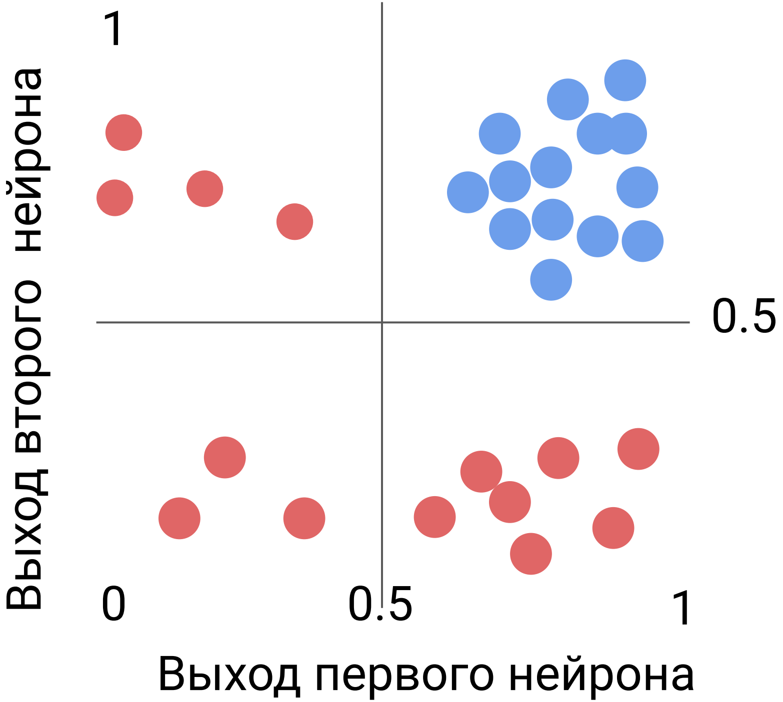


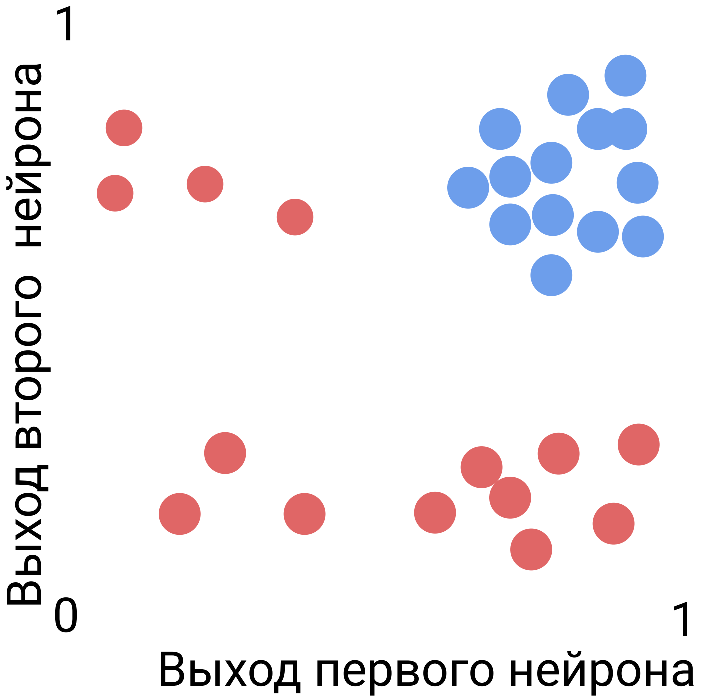
Нейрон 1 Нейрон 2



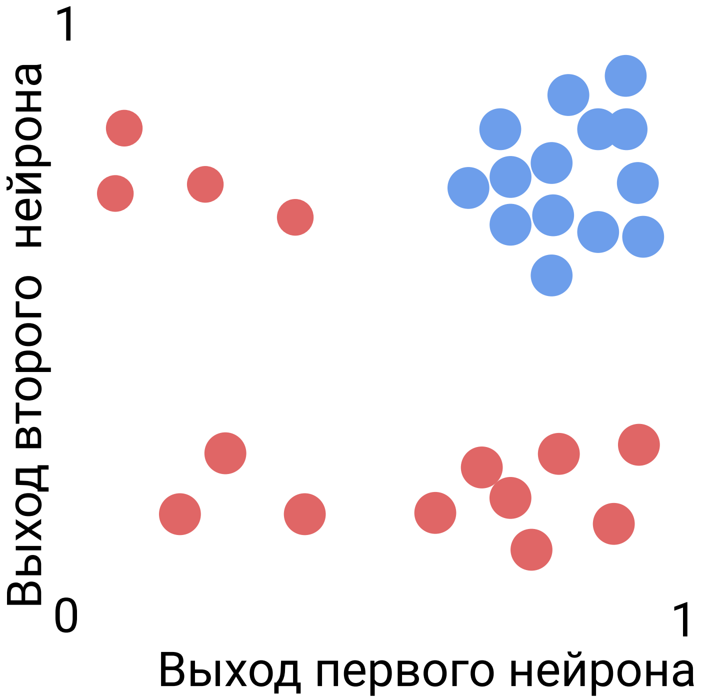


Нейрон 1 Нейрон 2




1
Выход второго нейрона
0
1
Выход первого нейрона

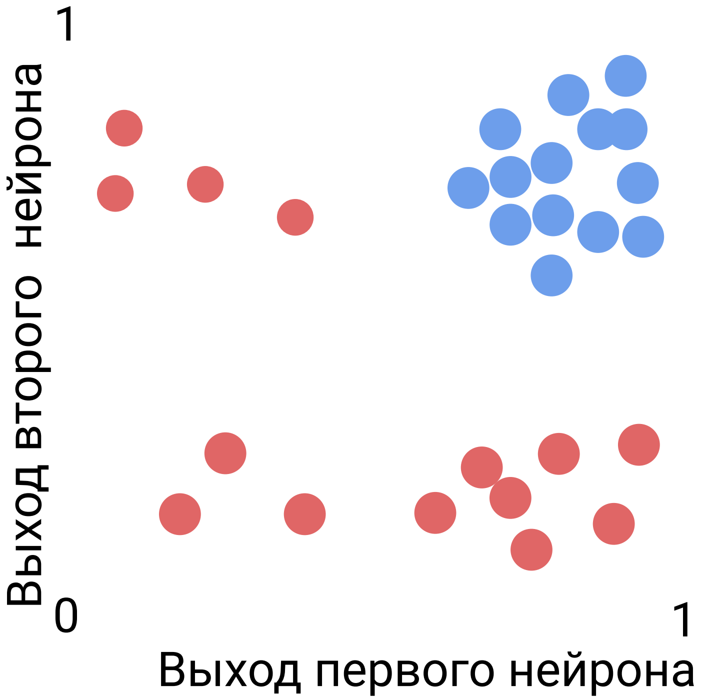
Выборка в этом пространстве -- линейно разделима

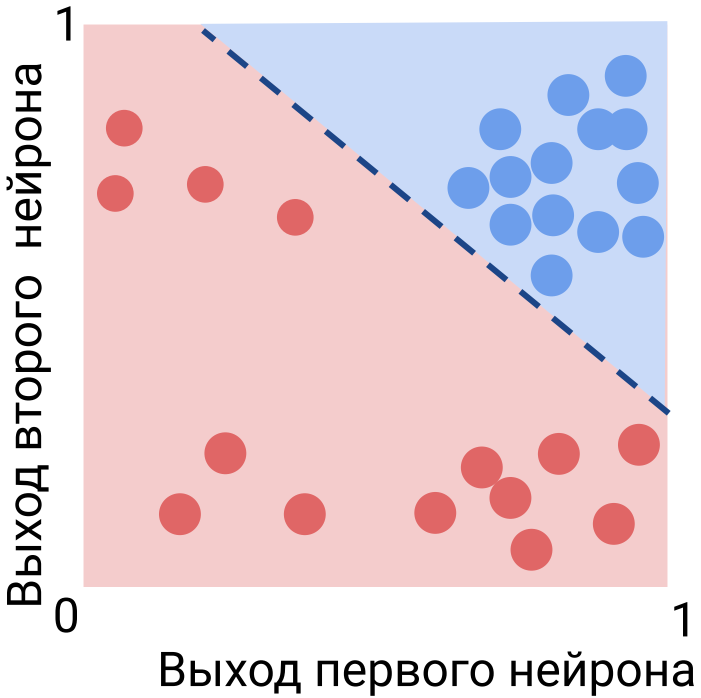


Skillbox

Выборка в этом пространстве -- линейно разделима

Можем применить к ней третий нейрон!




1
Выход второго нейрона
0
1
Выход первого нейрона

Что произошло?

Неформально:

- Нейроны “связаны” своими выходами
- Вместе они смогли “придумать” что каждый из них может сделать, чтобы в итоге получить правильный ответ

Что произошло?

Неформально:

- Нейроны “связаны” своими выходами
- Вместе они смогли “придумать” что каждый из них может сделать, чтобы в итоге получить правильный ответ
    - На самом деле это мы за них придумали. Но это пока :)

Что произошло?

Неформально:

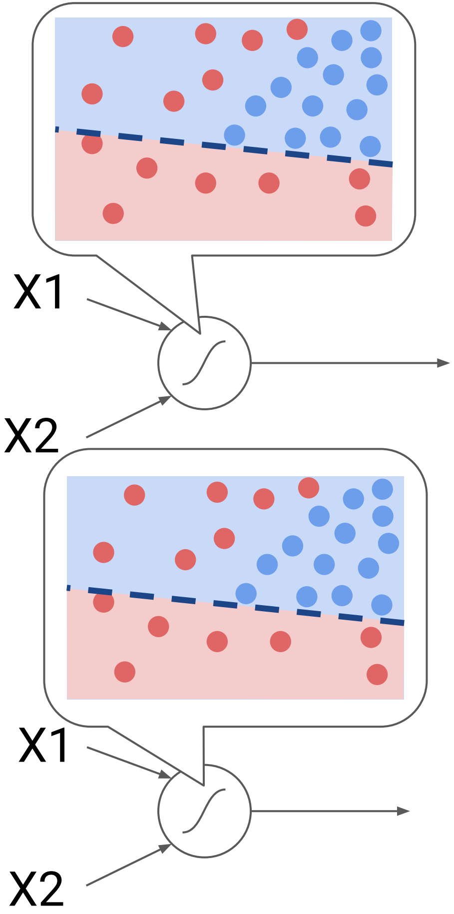
- Нейроны “связаны” своими выходами
- Вместе они смогли “придумать” что каждый из них может сделать, чтобы в итоге получить правильный ответ
  - На самом деле это мы за них придумали. Но это пока :)
- Т.к. **нейроны могут разделять пространство линейно**, то первые два сделали так, чтобы третьему данные достались в удобной -- линейно разделимой форме

Что произошло?

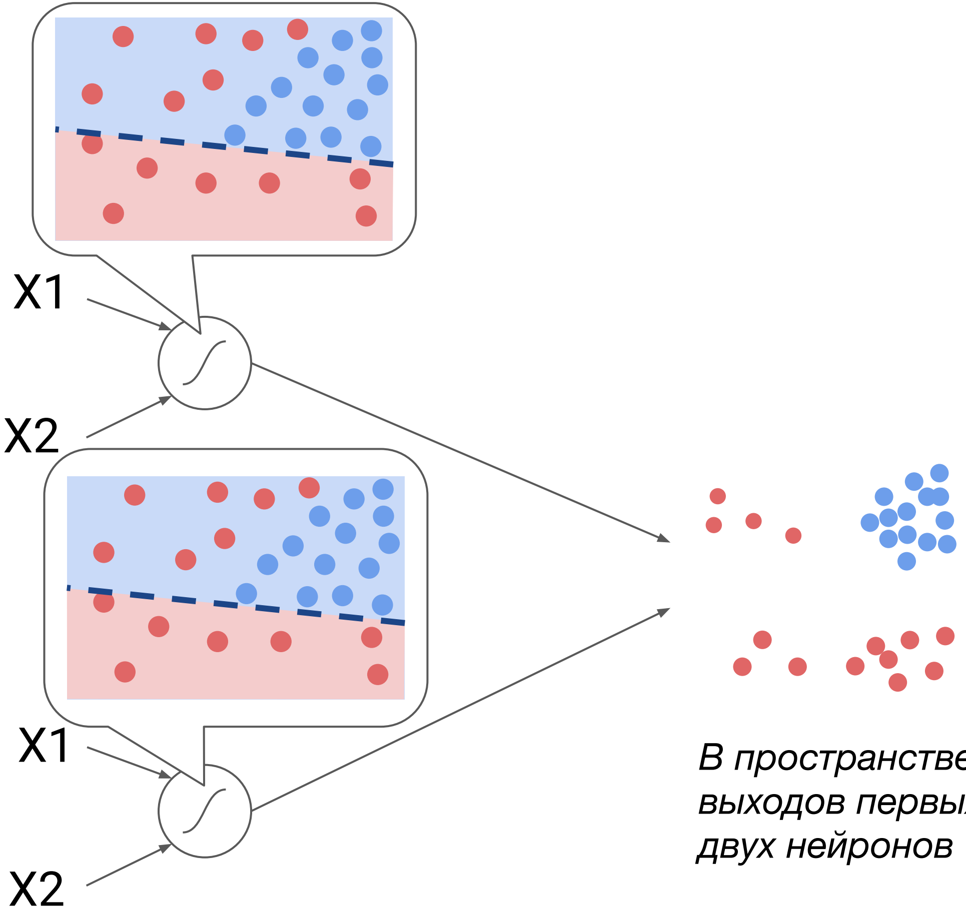
Неформально:

- Нейроны “связаны” своими выходами
- Вместе они смогли “придумать” что каждый из них может сделать, чтобы в итоге получить правильный ответ
  - На самом деле это мы за них придумали. Но это пока :)
- Т.к. **нейроны могут разделять пространство линейно**, то первые два сделали так, чтобы третьему данные достались в удобной -- линейно разделимой форме

Кажется, что где-то это уже слышали? -- Kernel trick в SVM.
[видео пример]

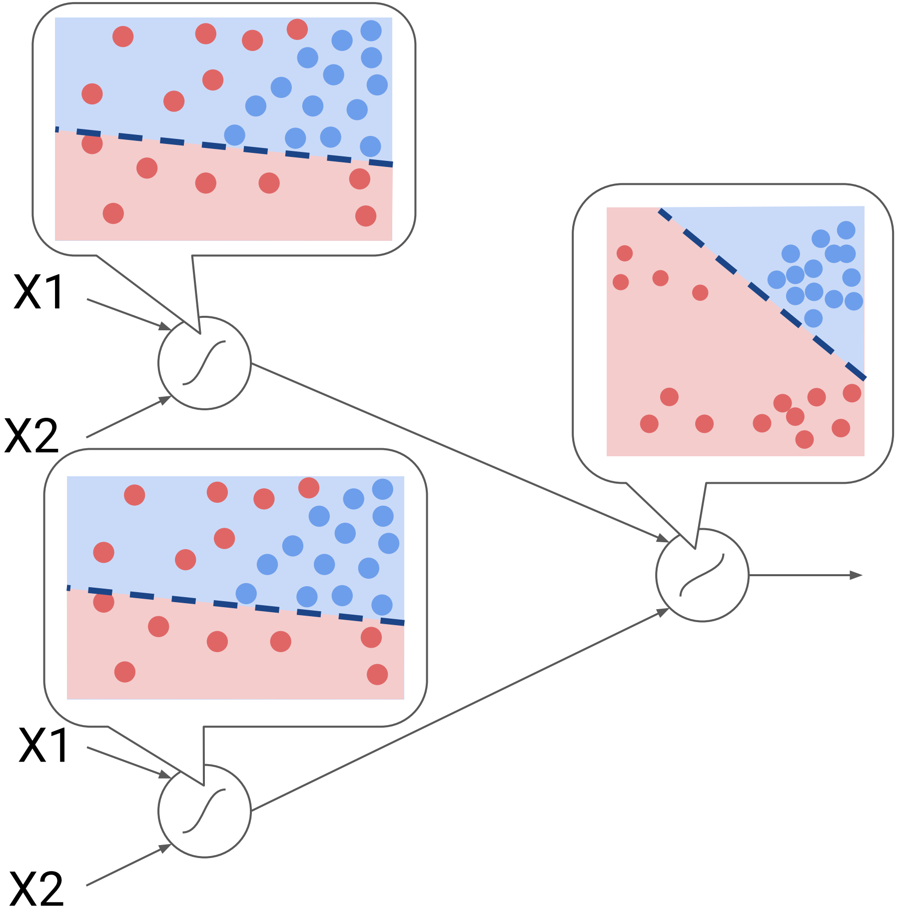

X1
X2
X1
X2

Skillbox

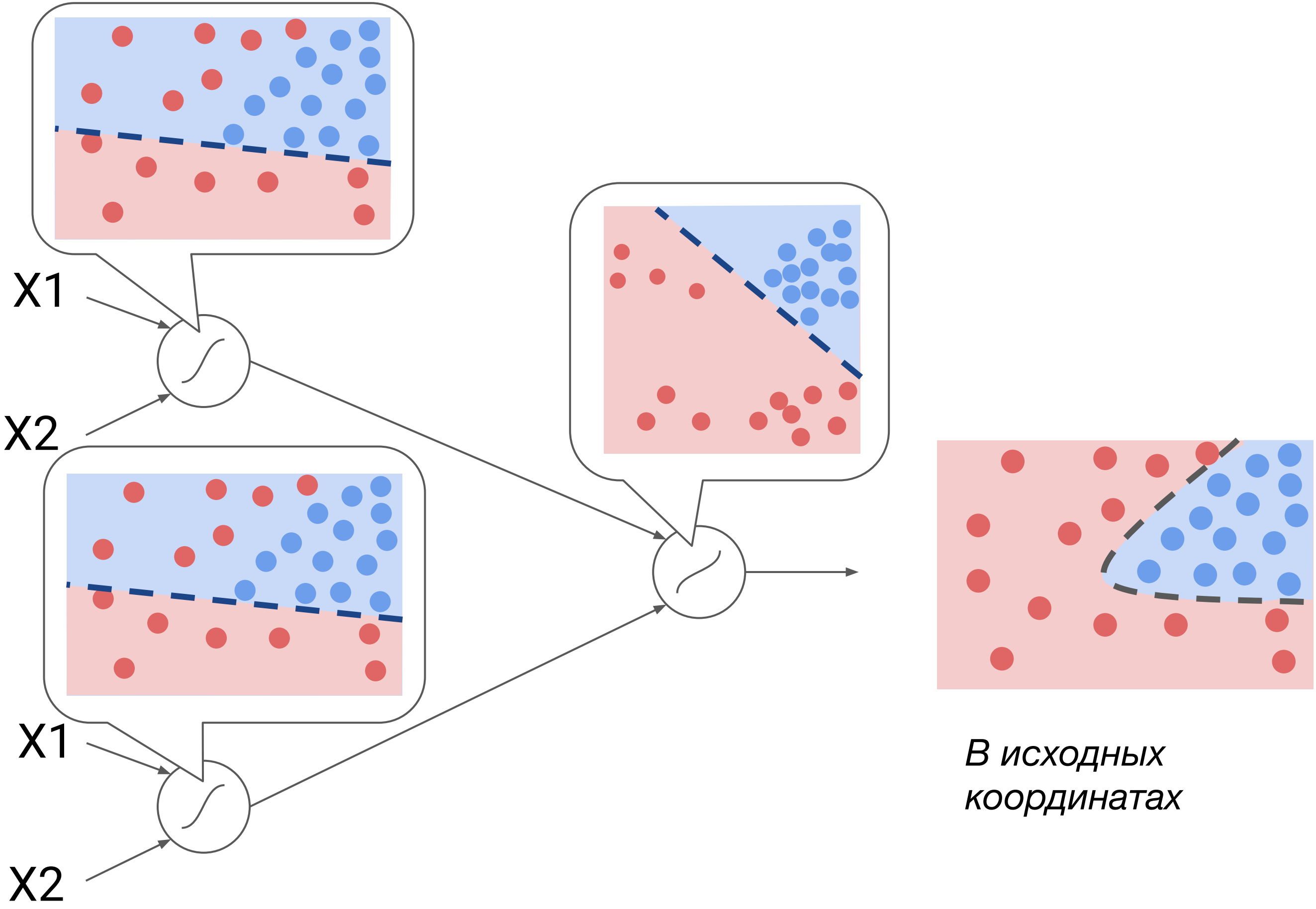


В пространстве выходов первых двух нейронов

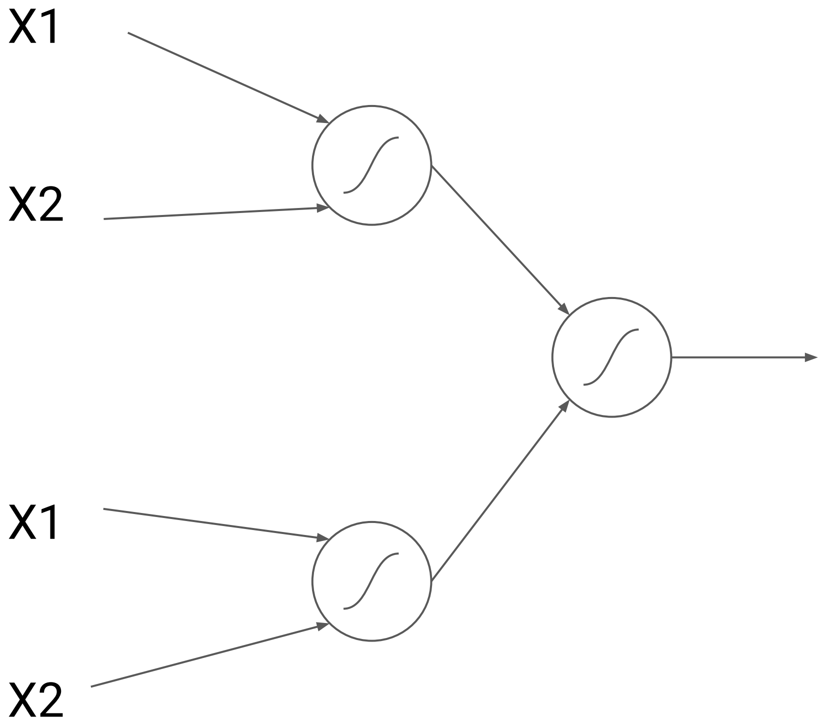
Skillbox

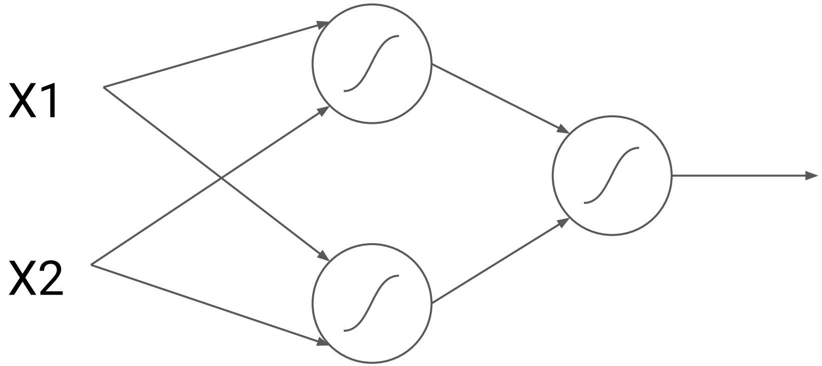


Skillbox



В исходных координатах

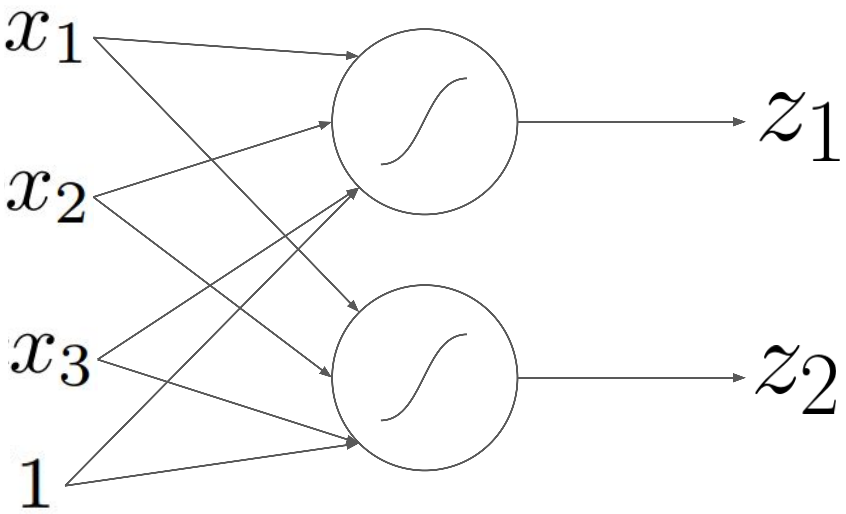

X1
X2
X1
X2



Нейронная сеть!

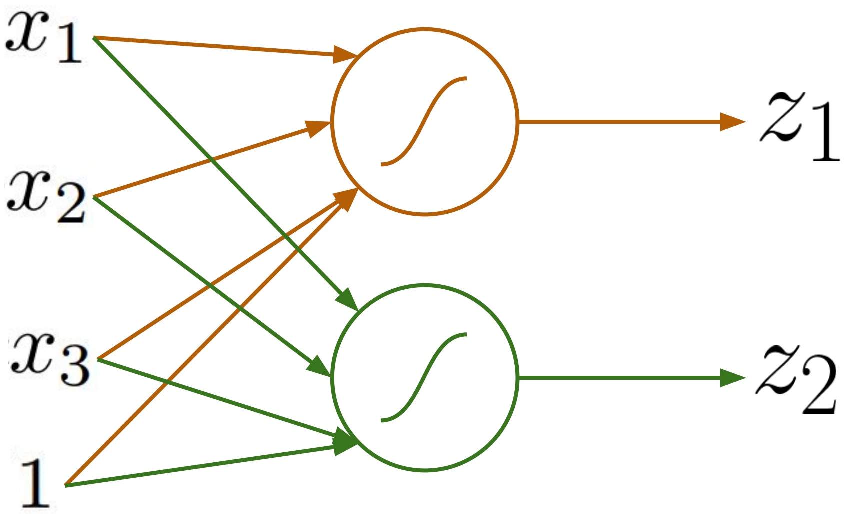
Skillbox

Нейронная сеть

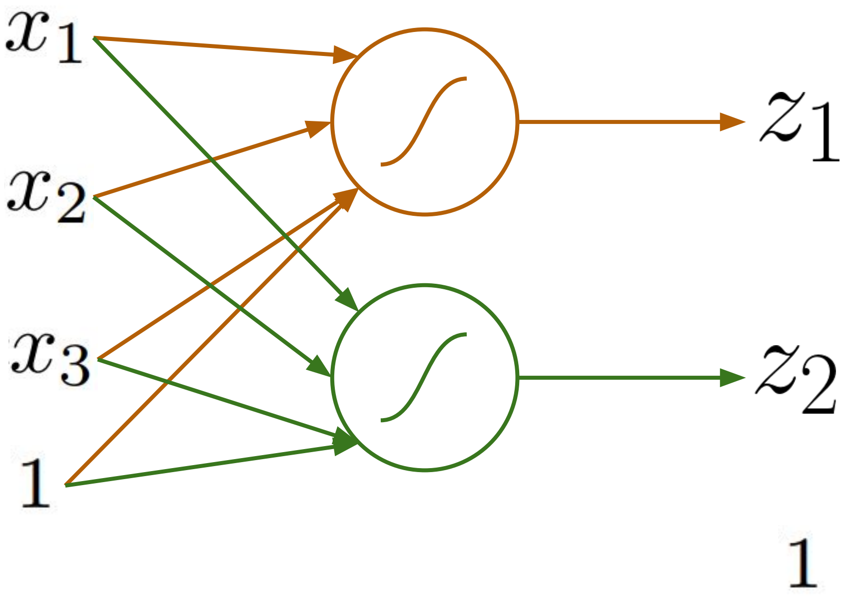


Skillbox

Нейронная сеть

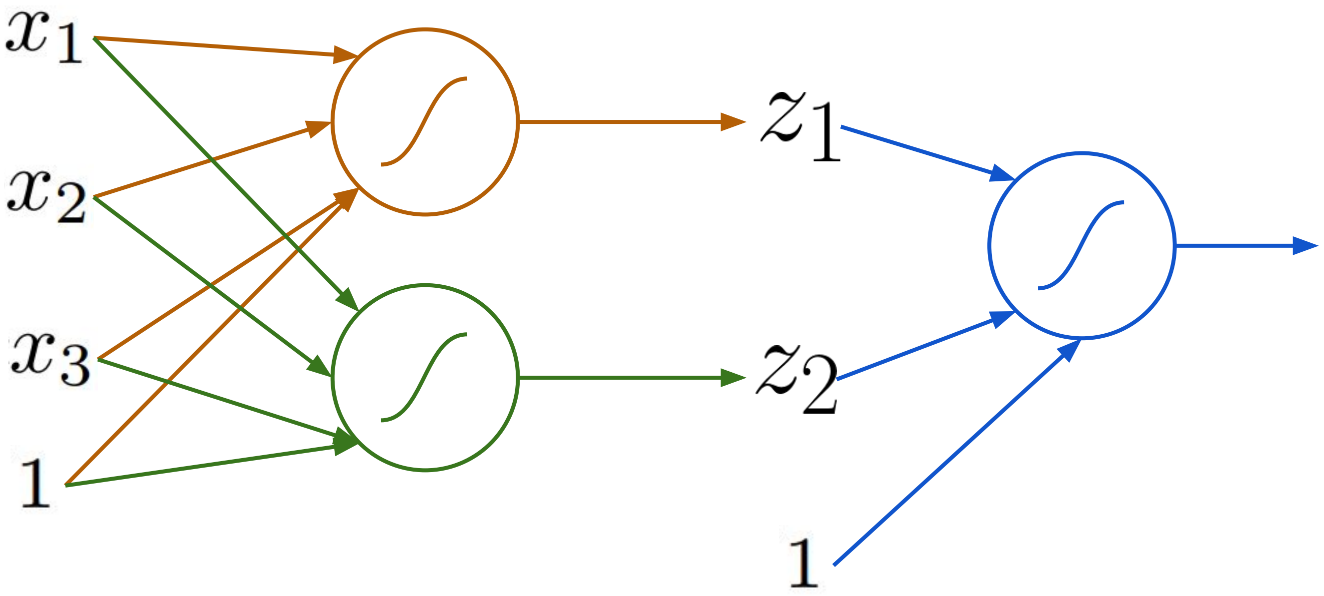


Нейронная сеть



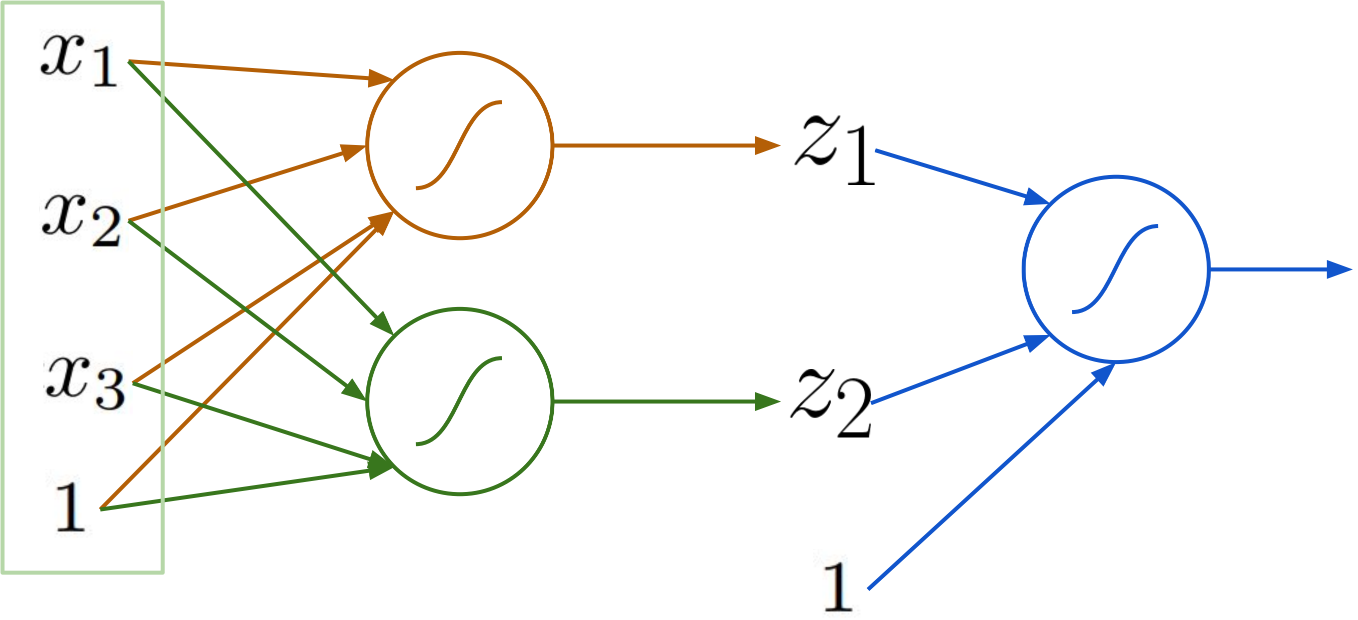
Skillbox

Нейронная сеть



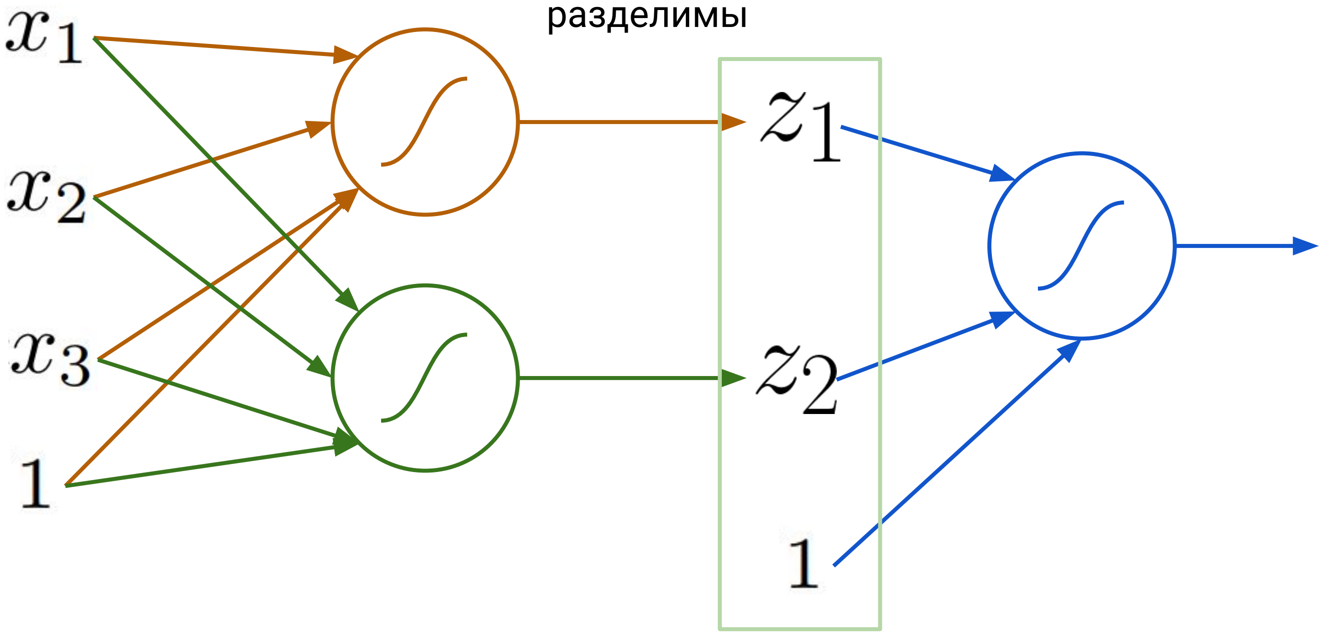
Нейронная сеть

Исходные признаки



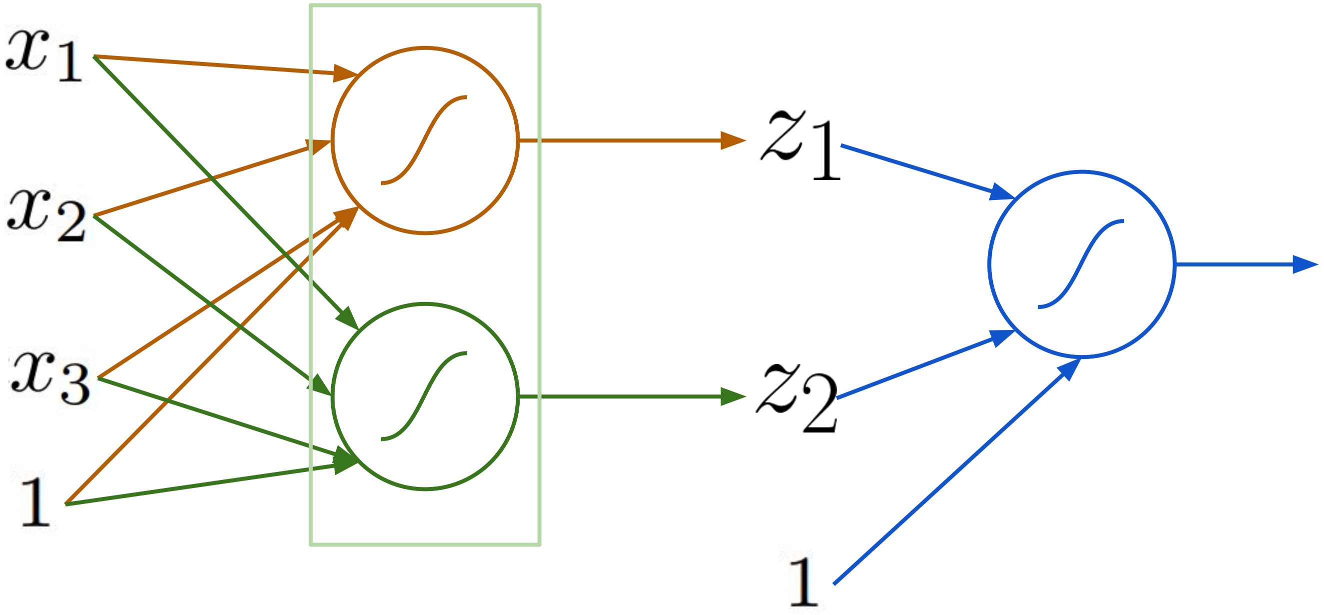
Нейронная сеть

Признаки, которые “придумала” сеть, чтобы объекты были линейно разделимы

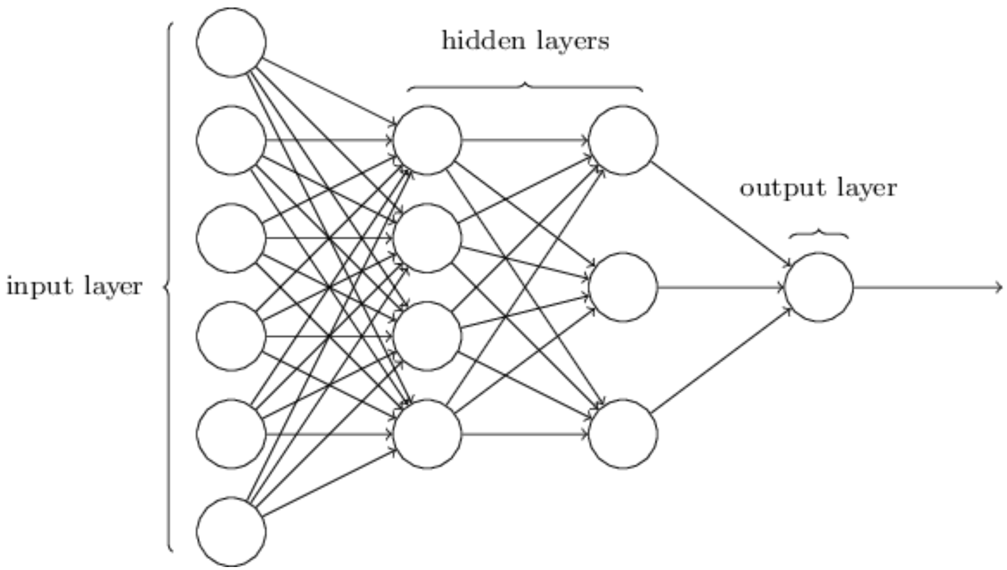


Нейронная сеть

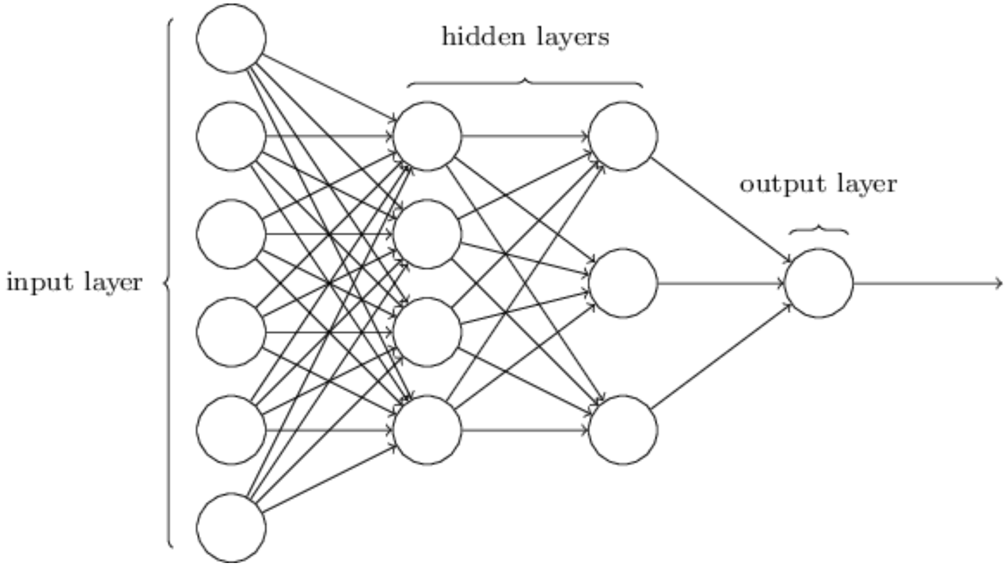
Скрытый слой (hidden layer)



Нейронная сеть



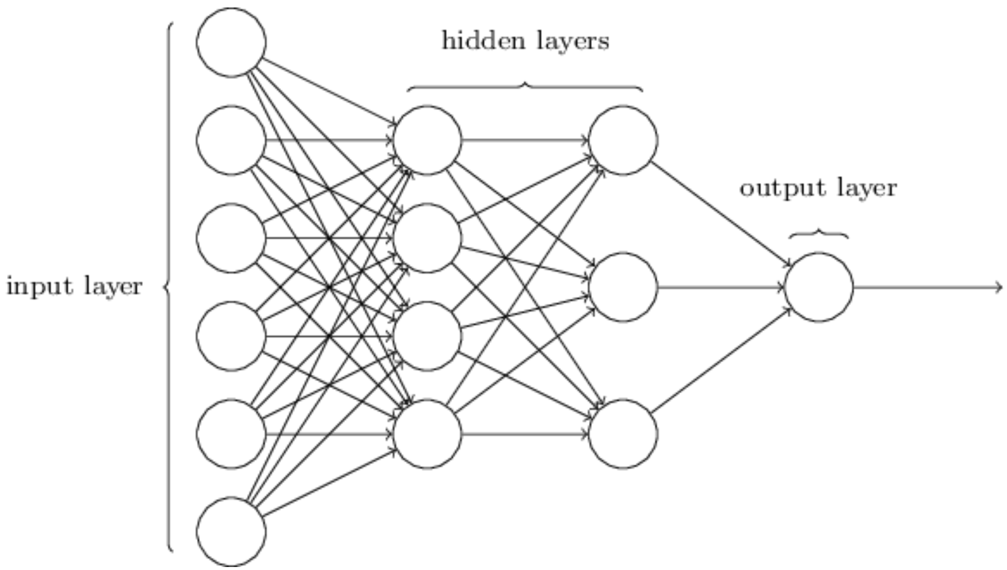
Нейронная сеть



Как “читать”?

- Двухслойная нейронная сеть (количество скрытых слоев)

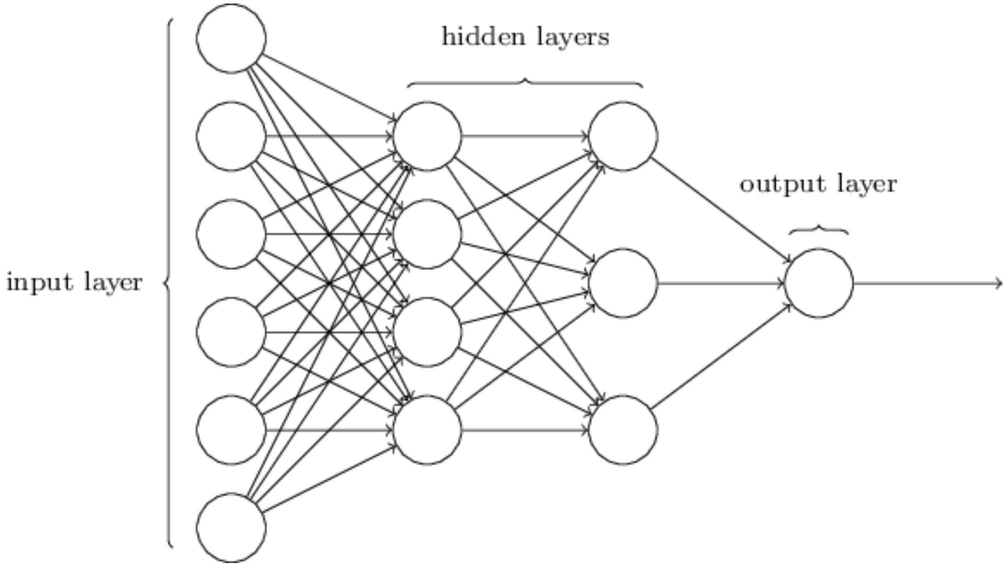
Нейронная сеть



Как “читать”?

- Двухслойная нейронная сеть (количество скрытых слоев)
- Количество входных признаков -- 6
- Количество выходов -- 1

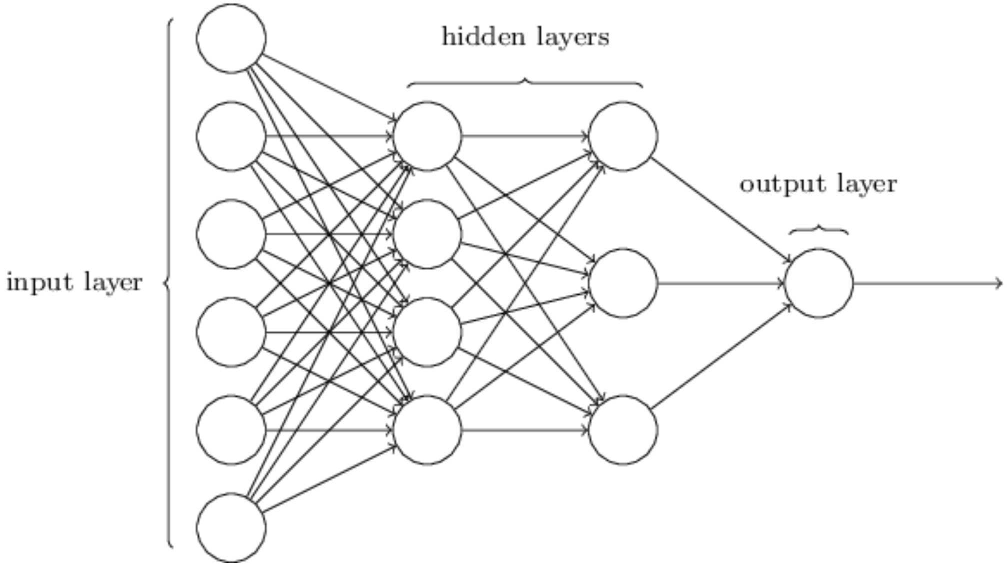
Нейронная сеть



Как “читать”?

- Двухслойная нейронная сеть (количество скрытых слоев)
- Количество входных признаков -- 6
- Количество выходов -- 1
- Первый скрытый слой состоит из 4 нейронов, а второй из 3.

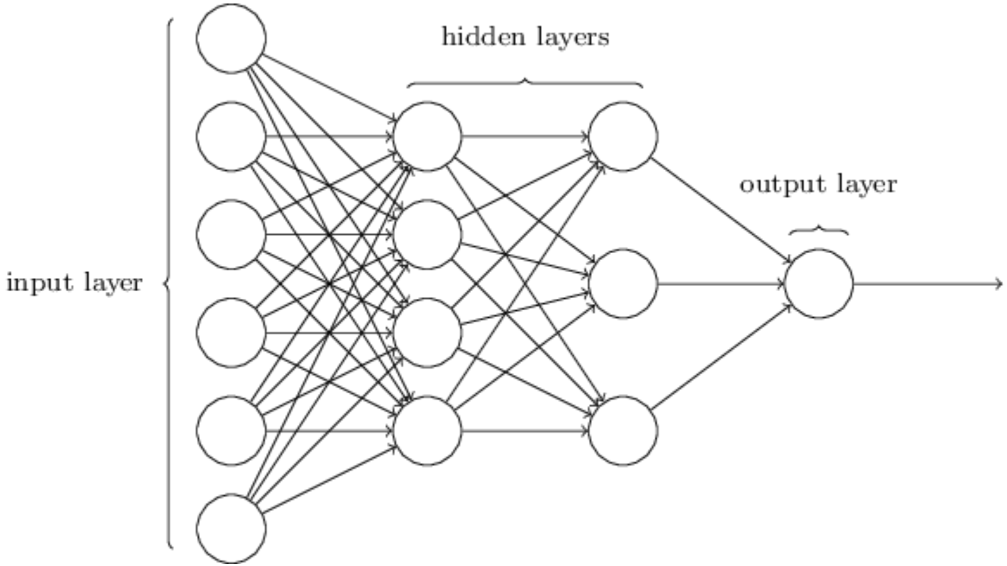
Нейронная сеть



Как “читать”?

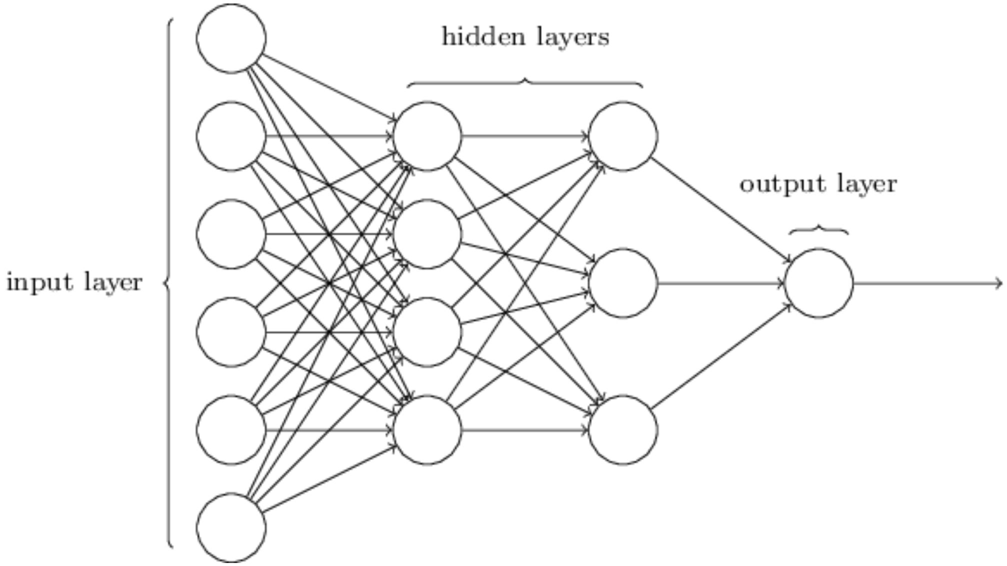
- Двухслойная нейронная сеть (количество скрытых слоев)
- Количество входных признаков -- 6
- Количество выходов -- 1
- Первый скрытый слой состоит из 4 нейронов, а второй из 3.
- *Скрытые слои называют еще:*
  - *Полносвязный*
  - *Fully connected, Dense*

Нейронная сеть



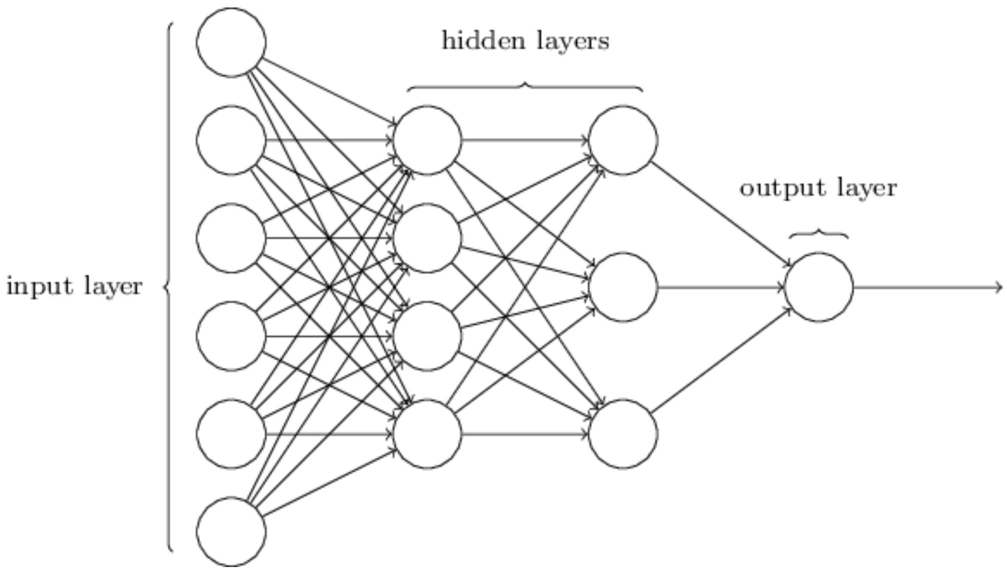
- Чем больше скрытых слоев и нейронов, тем более сложную функцию может аппроксимировать нейронная сеть

Нейронная сеть



- Чем больше скрытых слоев и нейронов, тем более сложную функцию может аппроксимировать нейронная сеть
- Доказано, что нейронная сеть с одним скрытым слоем, нелинейной функцией активации и с конечным числом нейронов может приблизить любую функцию* (неформально)
- *Universal approximation theorem

Нейронная сеть



- Чем больше скрытых слоев и нейронов, тем более сложную функцию может аппроксимировать нейронная сеть
- Доказано, что нейронная сеть с одним скрытым слоем, нелинейной функцией активации и с конечным числом нейронов может приблизить любую функцию* (неформально)
- *Universal approximation theorem
- Но не стоит забывать о переобучении)

Итог

- Мы своими глазами увидели с какой задачей не может справится один нейрон
- Поняли, что, используя несколько нейронов можно разделять более сложные данные
- Такие группы связанных нейронов называются нейронной сетью
- Познакомились с терминологией, связанной с нейронными сетями