Skillbox

Обучение нейронных сетей

Skillbox

Обучение нейронных сетей

Градиентный спуск

Нейронные сети

- Мы знаем, что такое нейронная сеть
- Знаем, как записать задачу оптимизации для нейронной сети

Нейронные сети

- Мы знаем, что такое нейронная сеть
- Знаем, как записать задачу оптимизации для нейронной сети
- Но не знаем как ее решить)
    - Точнее, раньше мы подбирали коэффициенты руками, а это не очень практично

Задача оптимизации

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

$\boldsymbol{x}_i$ -- признаковое описание объекта i

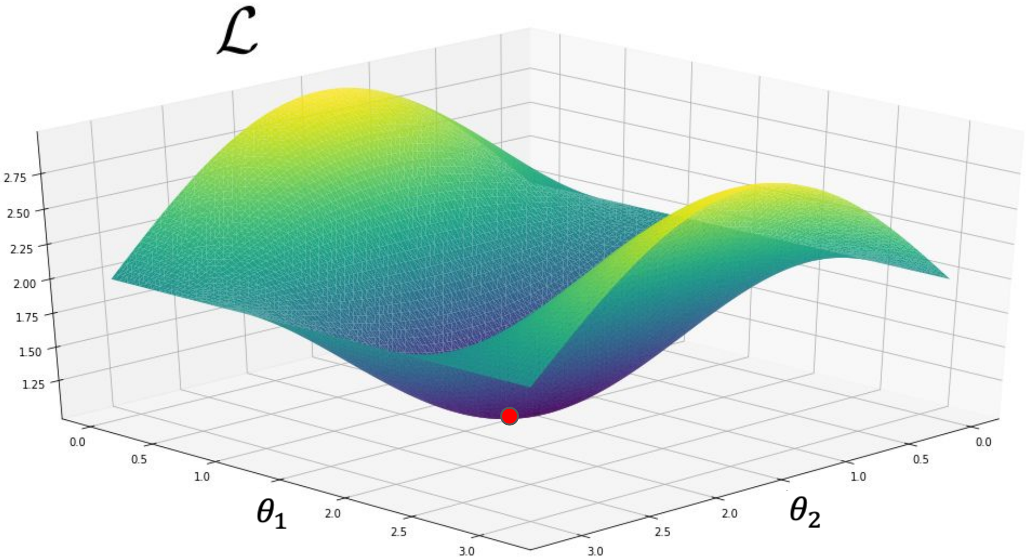
y_i -- правильный ответ для объекта i

f -- наш алгоритм (нейронная сеть)

$\boldsymbol{\theta}$ -- параметры алгоритма

Нужно найти такие параметры, чтобы значение $\mathcal{L}$ было минимальным

Задача оптимизации

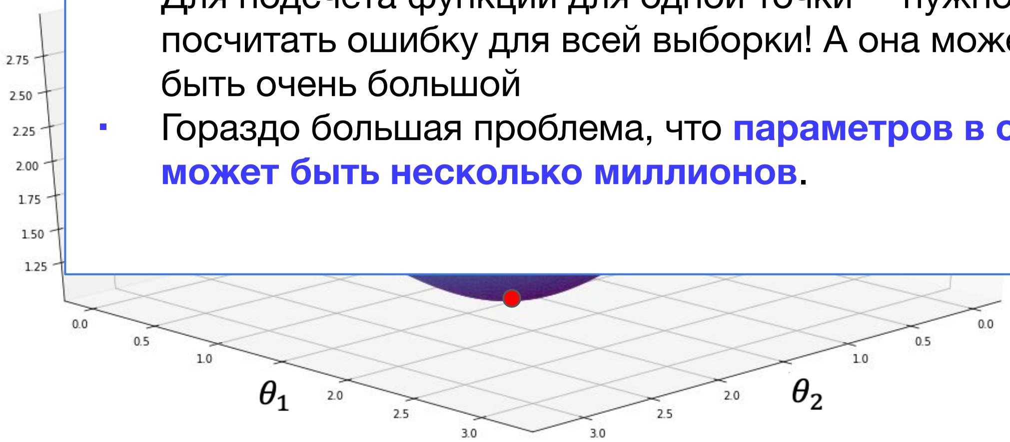


Два параметра (θ_1 , θ_2)

Посчитали значение $\mathcal{L}$ для всех комбинаций параметров
Тогда ответом для нашей задачи будет набор параметров, соответствующий красной точке

Задача оптимизации

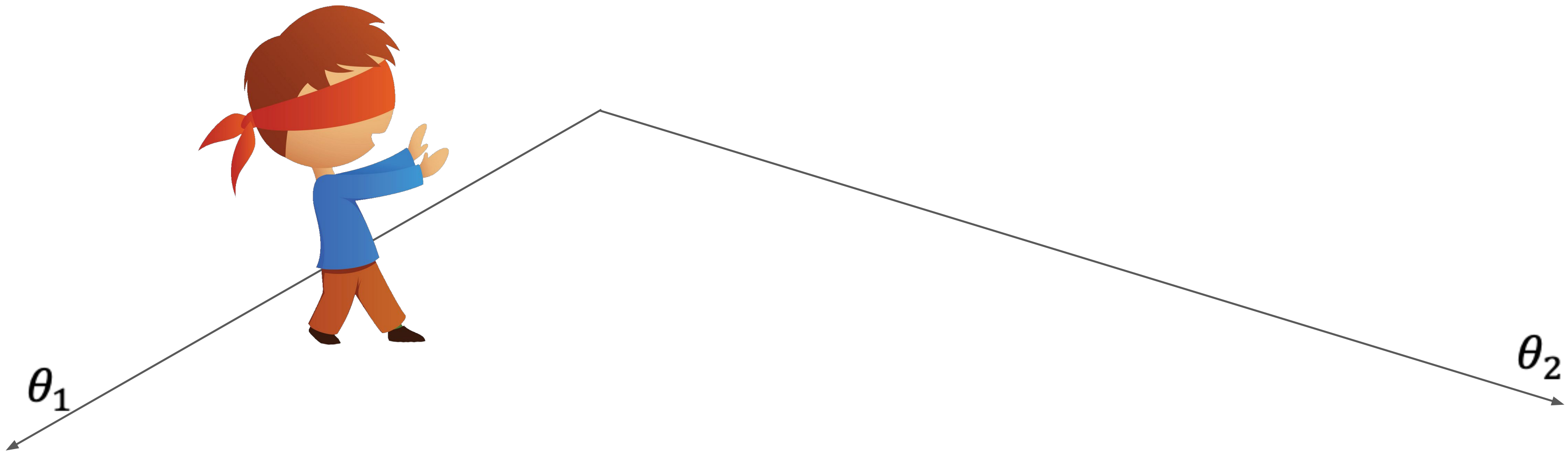
- **Звучит просто, но это не реализуемо на практике**
- Для подсчета функции для одной точки -- нужно посчитать ошибку для всей выборки! А она может быть очень большой
- Гораздо большая проблема, что **параметров в сети может быть несколько миллионов**.

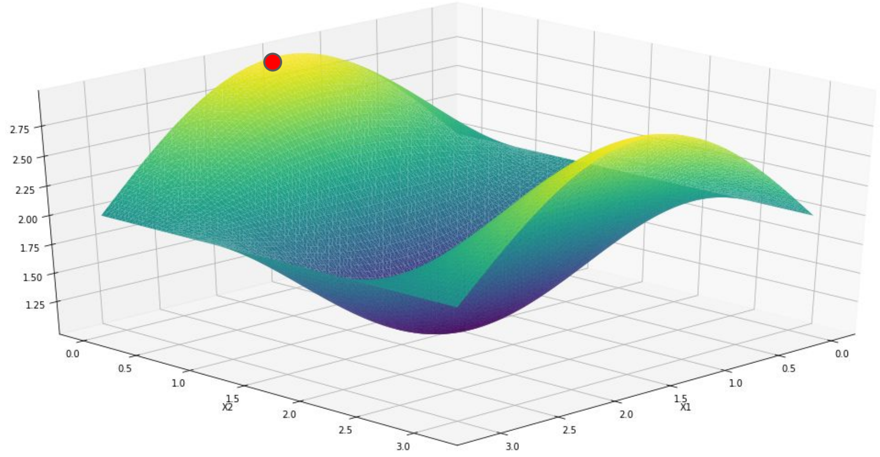


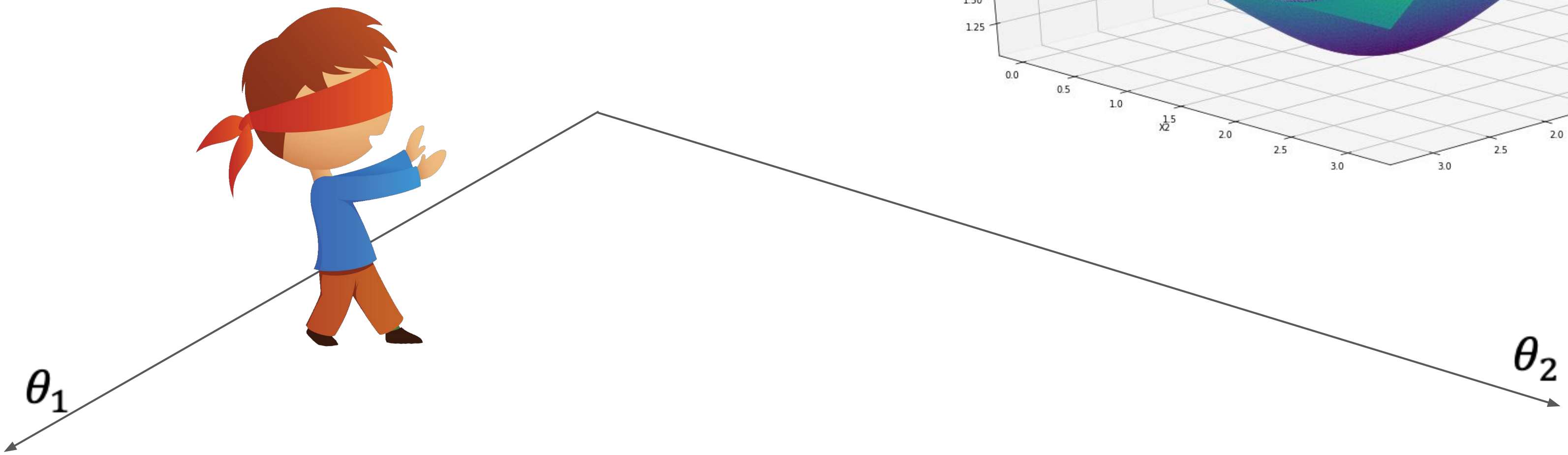
Два параметра (θ_1 , θ_2)

Посчитали значение $\mathcal{L}$ для всех комбинаций параметров
Тогда ответом для нашей задачи будет набор параметров, соответствующий красной точке

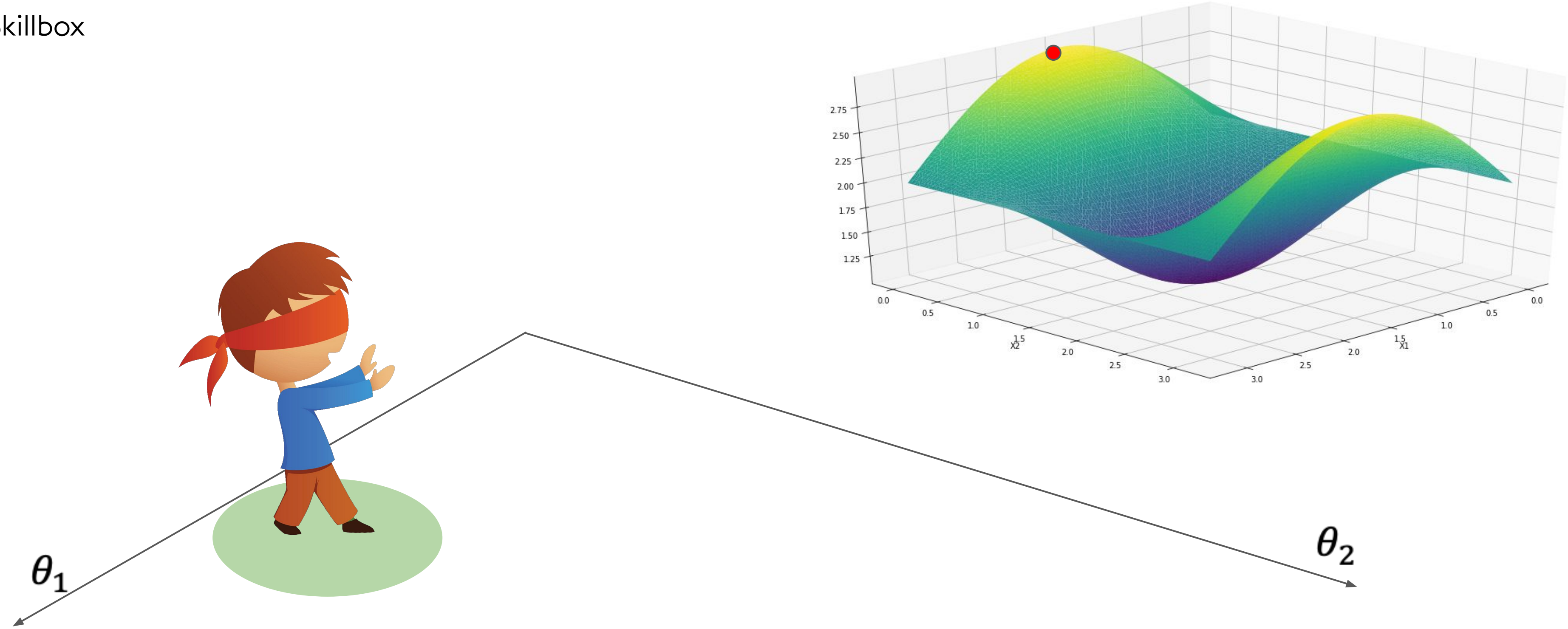
Skillbox

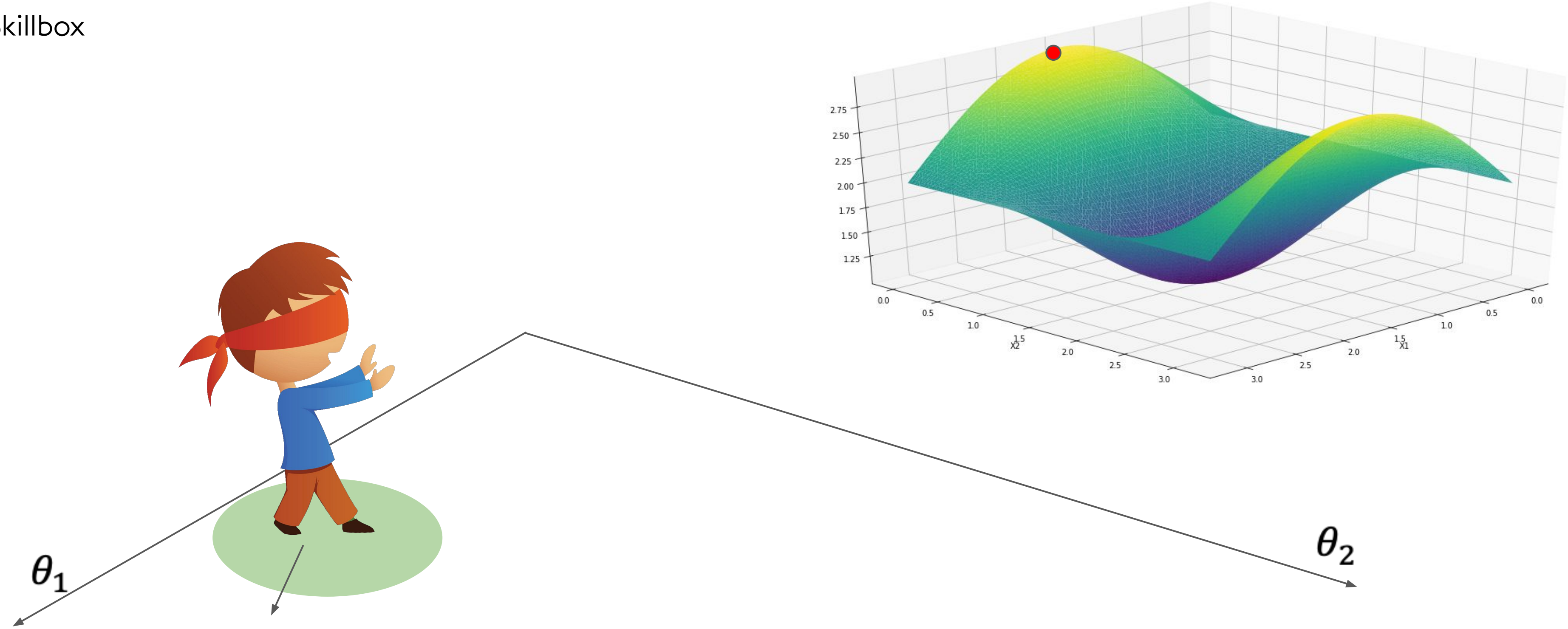


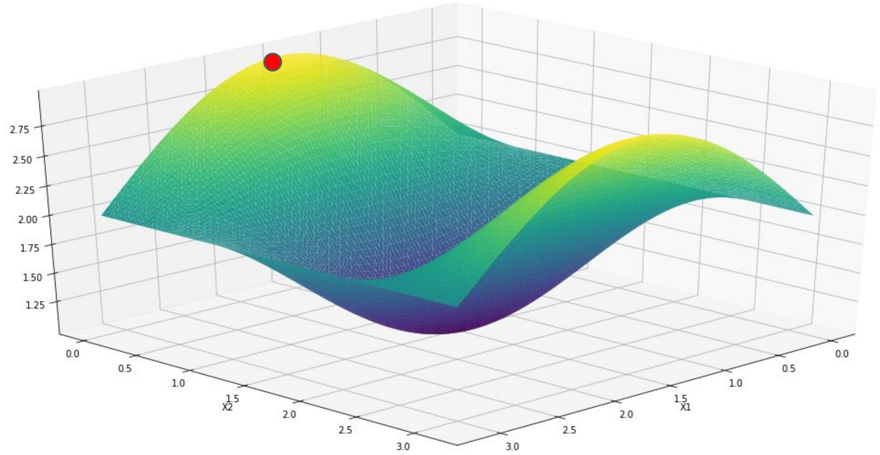

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
X2
2.0
2.5
3.0
X1

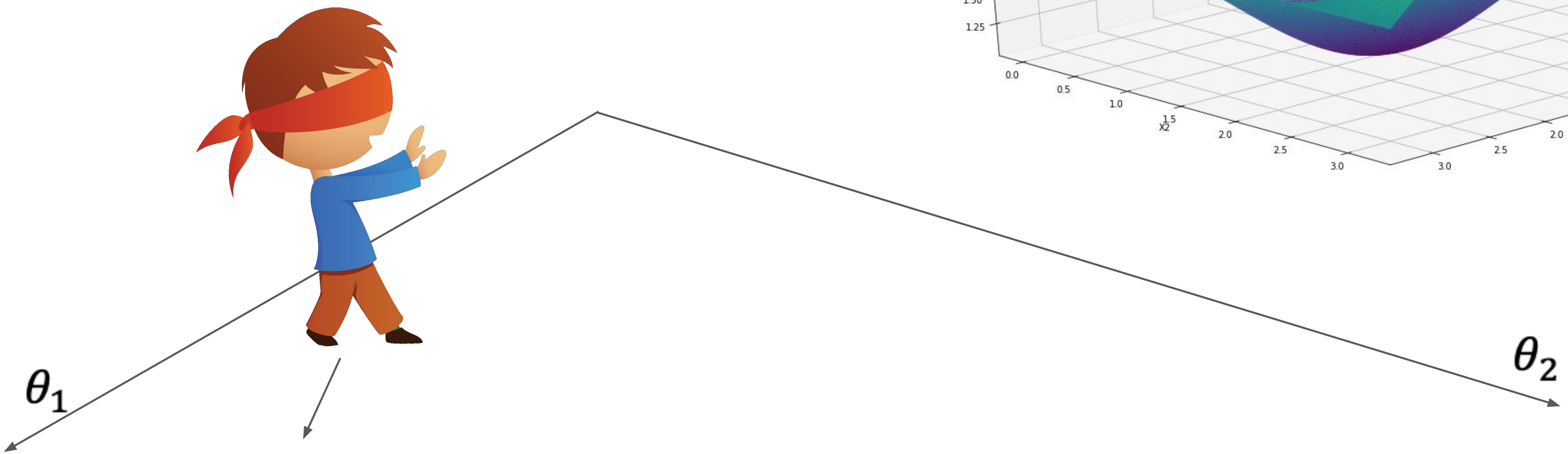

θ_1
θ_2

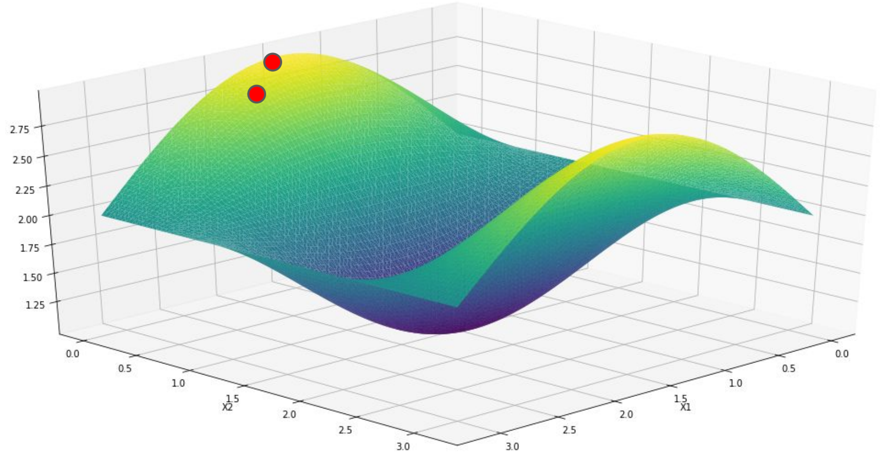
Skillbox

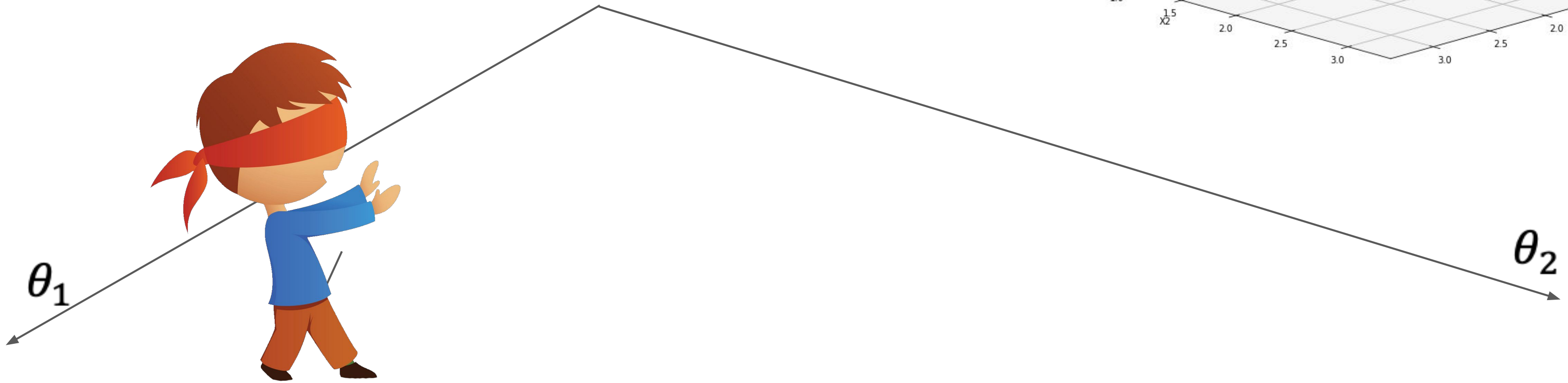



θ_1
θ_2
2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0 0.5 1.0 1.5 2.0 2.5 3.0
x2
0.0 0.5 1.0 1.5 2.0 2.5 3.0
x1

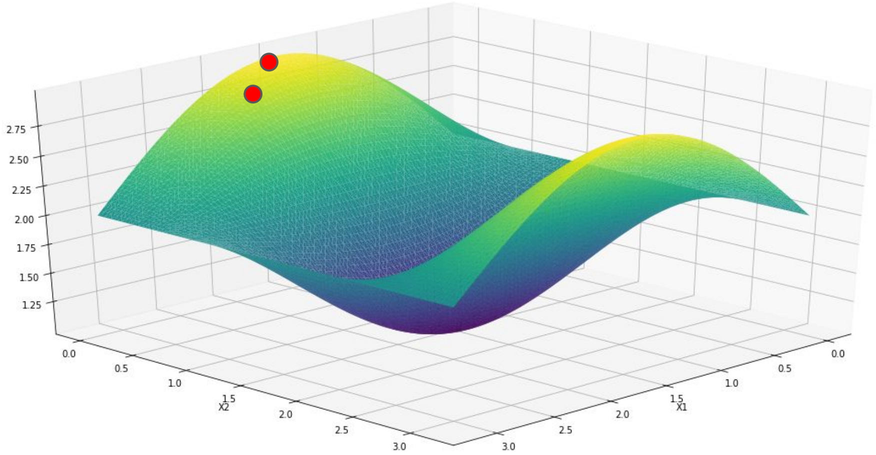

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
2.0
2.5
3.0
X2
X1

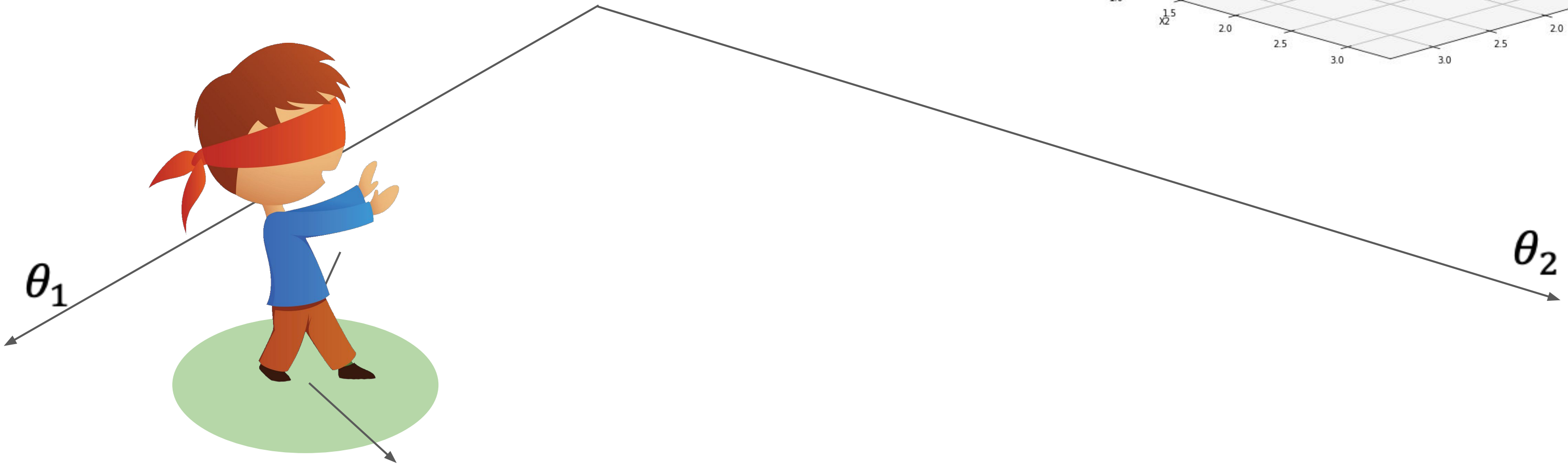

θ_1
θ_2

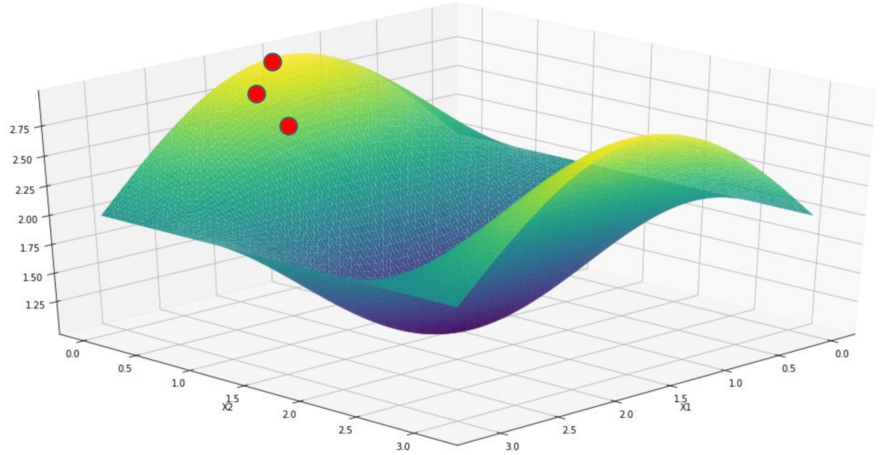

2.75
2.50
2.25
2.00
1.75
1.50
1.25
x2
x1

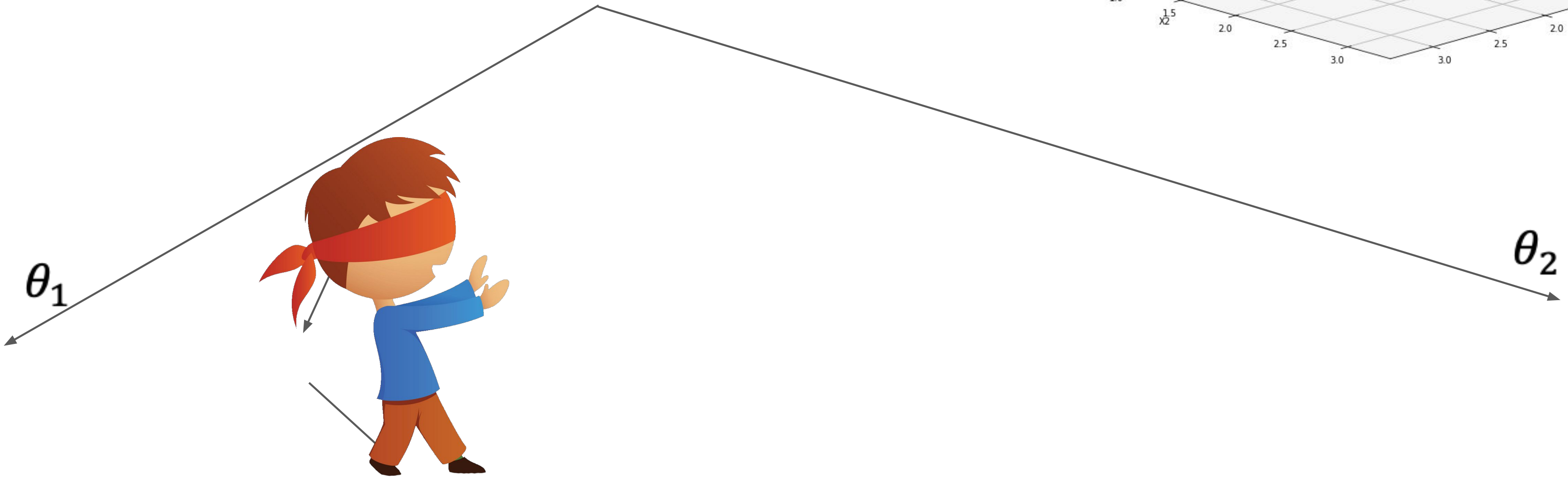

θ_1
θ_2

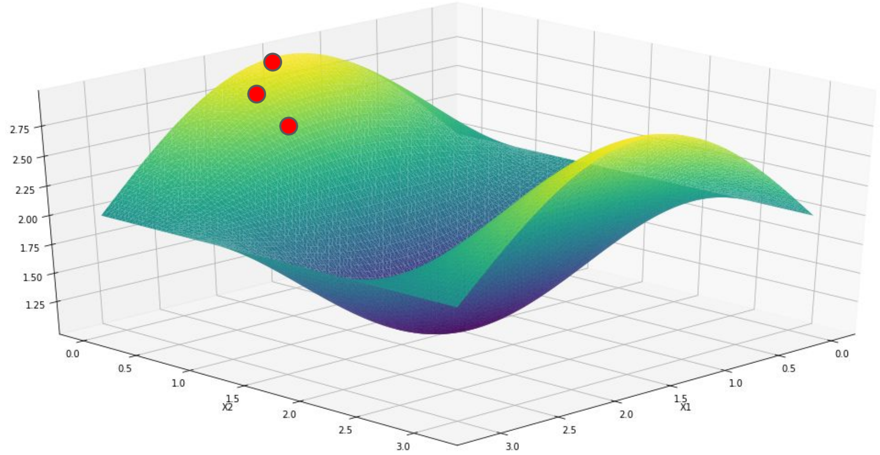
Skillbox

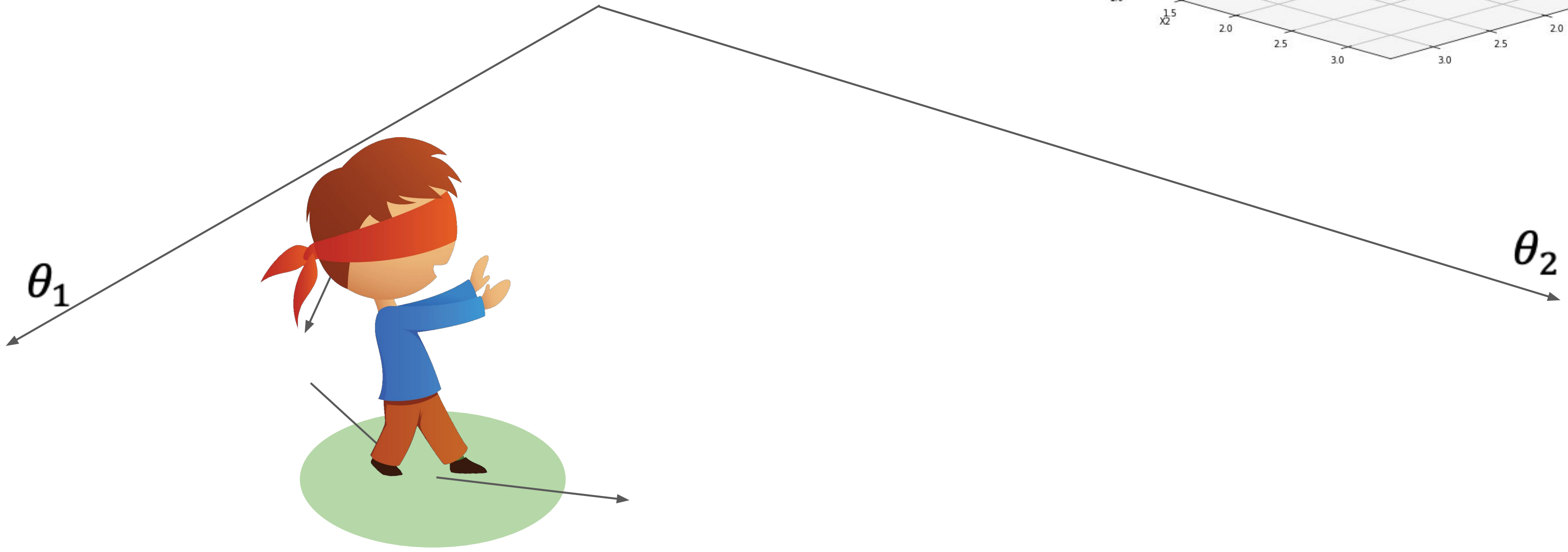

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
X2
2.0
2.5
3.0
0.0
0.5
1.0
1.5
X1
2.0
2.5
3.0


θ_1
θ_2

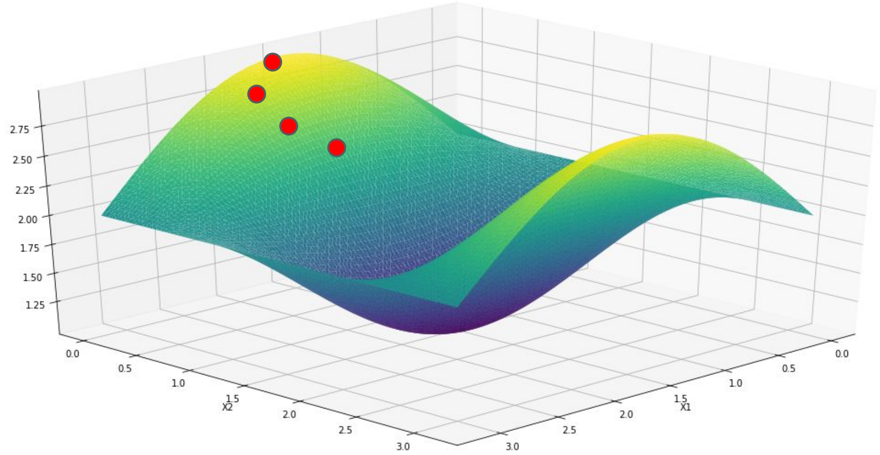

2.75
2.50
2.25
2.00
1.75
1.50
1.25
X2
X1

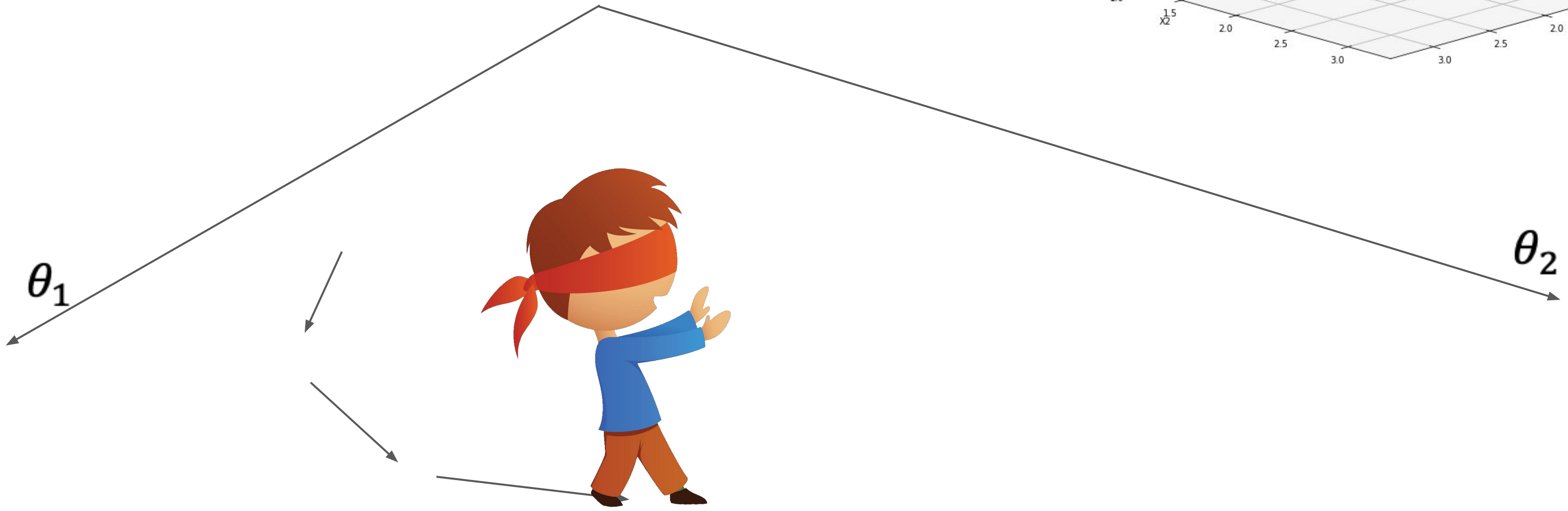

θ_1
θ_2

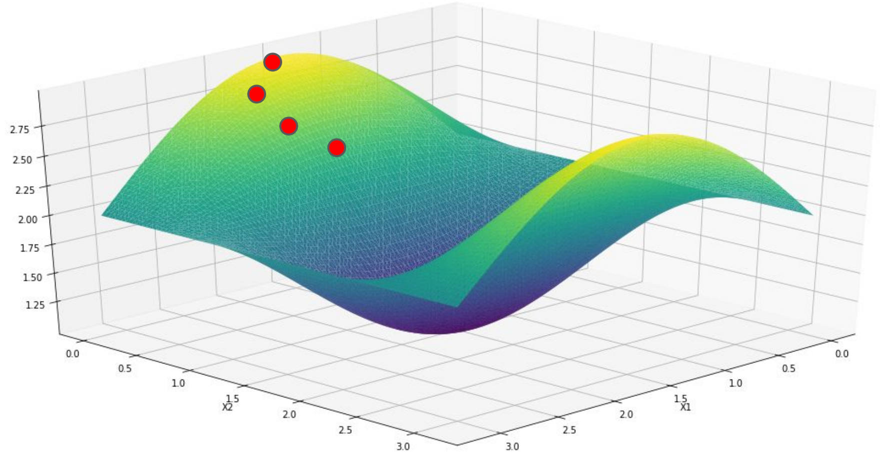

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
X2
2.0
2.5
3.0
0.0
0.5
1.0
1.5
X1
2.0
2.5
3.0

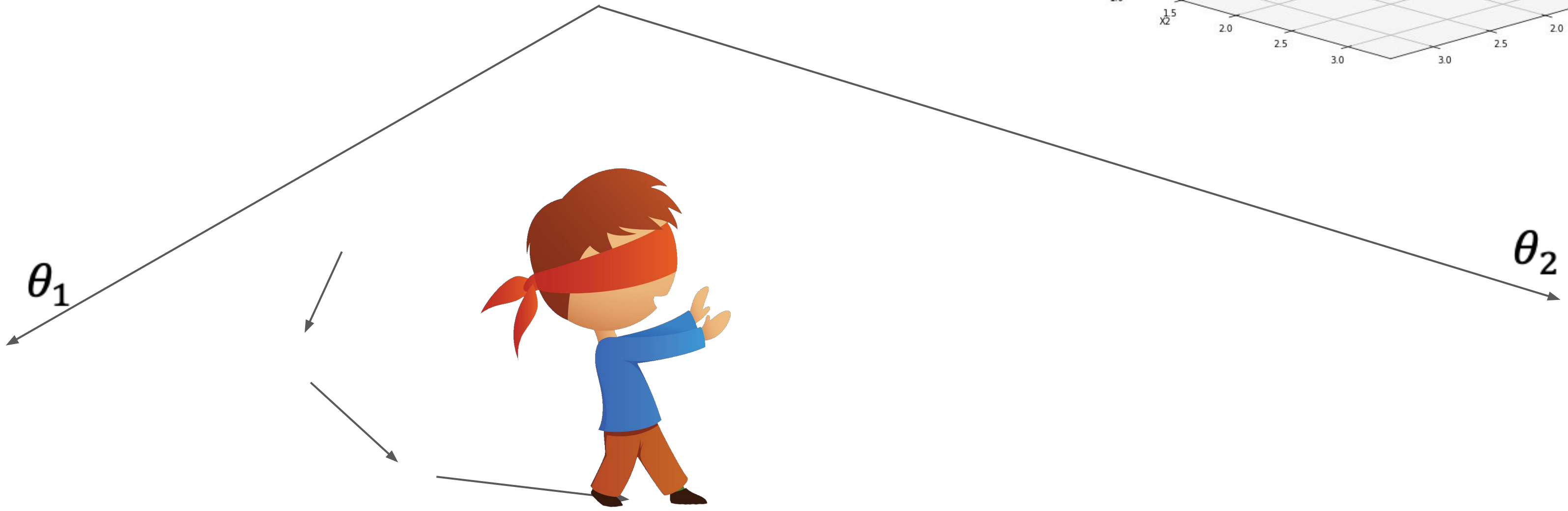

θ_1
θ_2

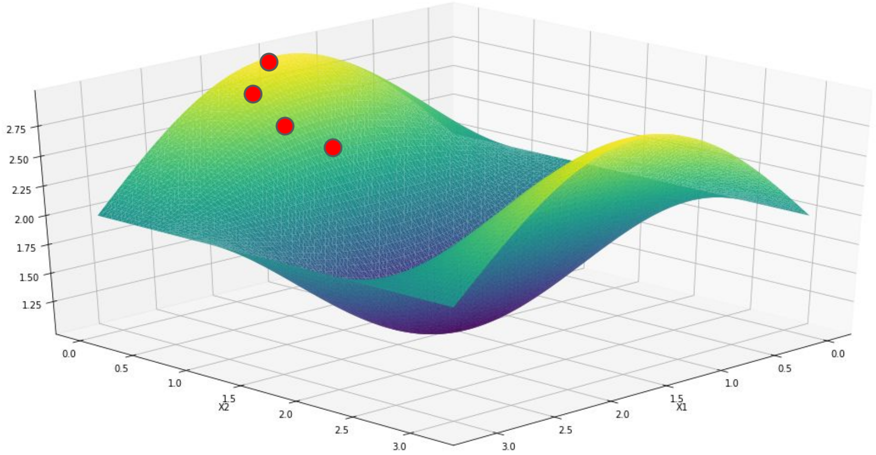
Skillbox

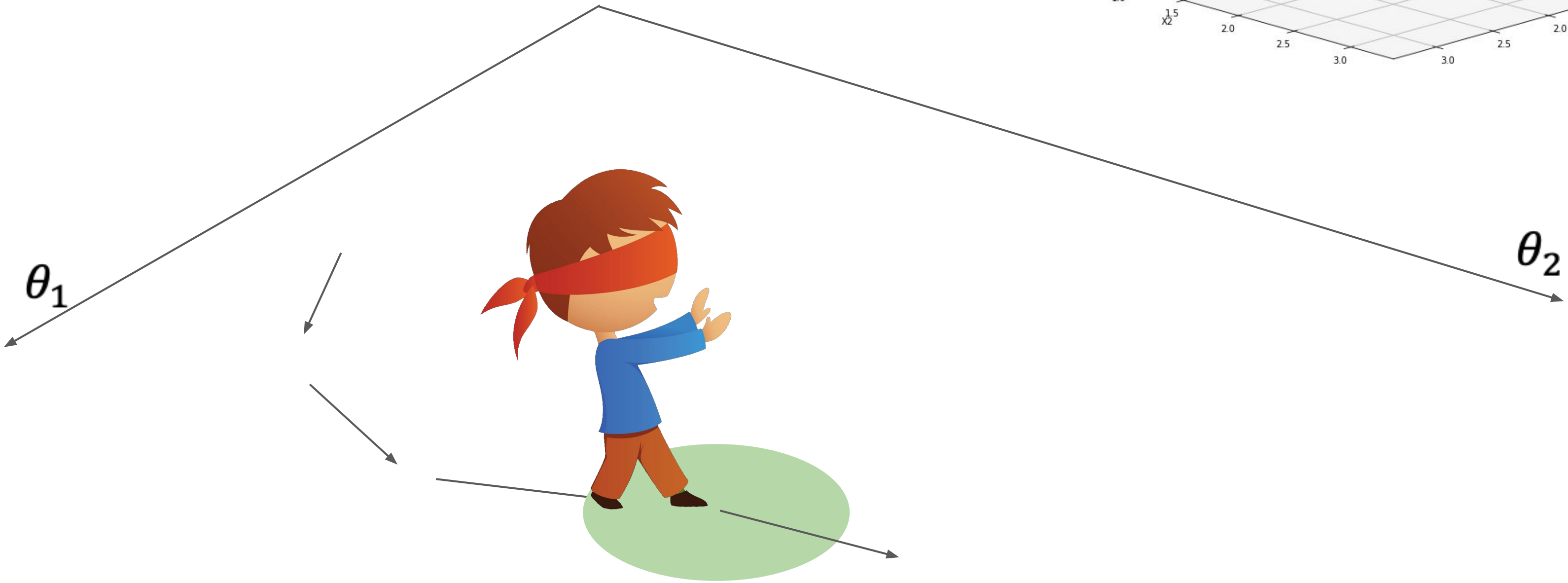

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
X2
2.0
2.5
3.0
0.0
0.5
1.0
1.5
X1
2.0
2.5
3.0

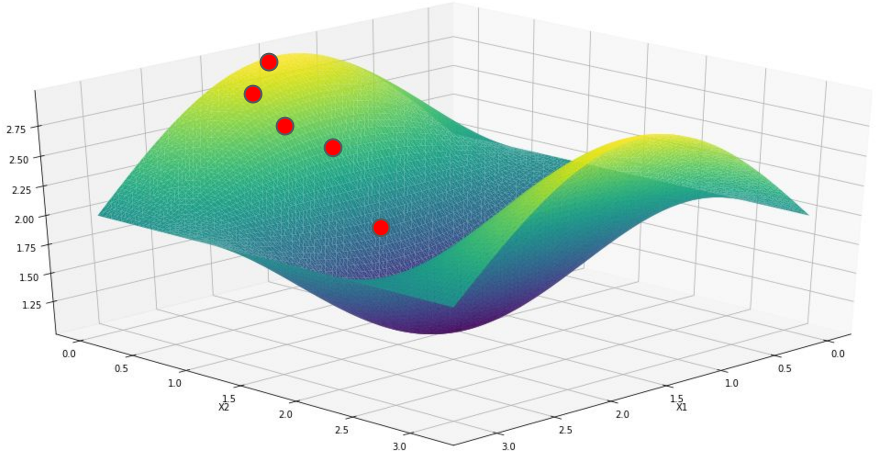

θ_1
θ_2

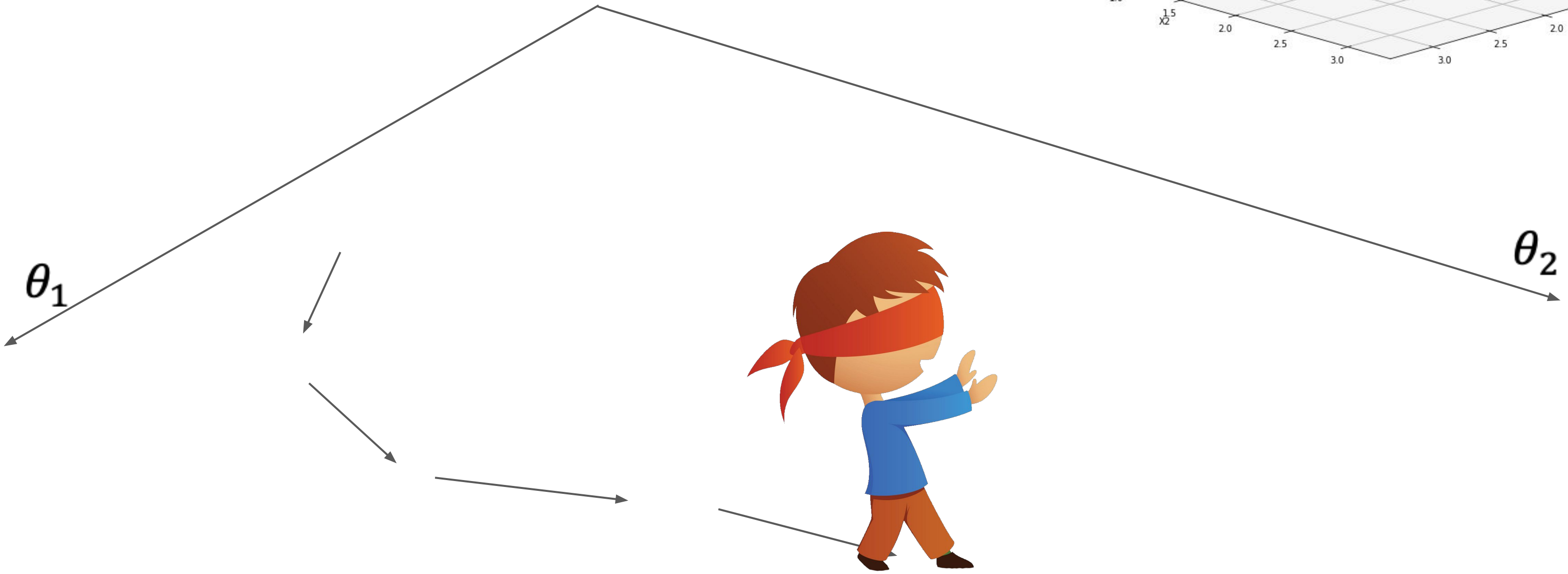

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
x2
2.0
2.5
3.0
0.0
0.5
1.0
1.5
x1
2.0
2.5
3.0


θ_1
θ_2

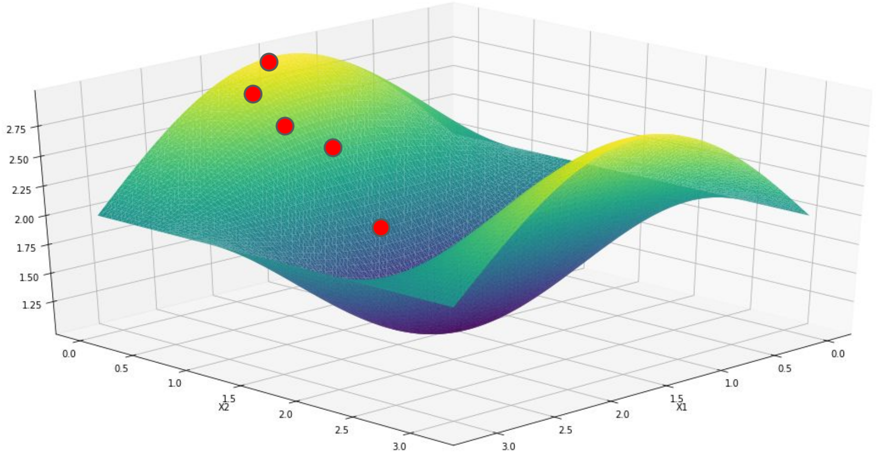

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
2.0
2.5
3.0
X2
X1

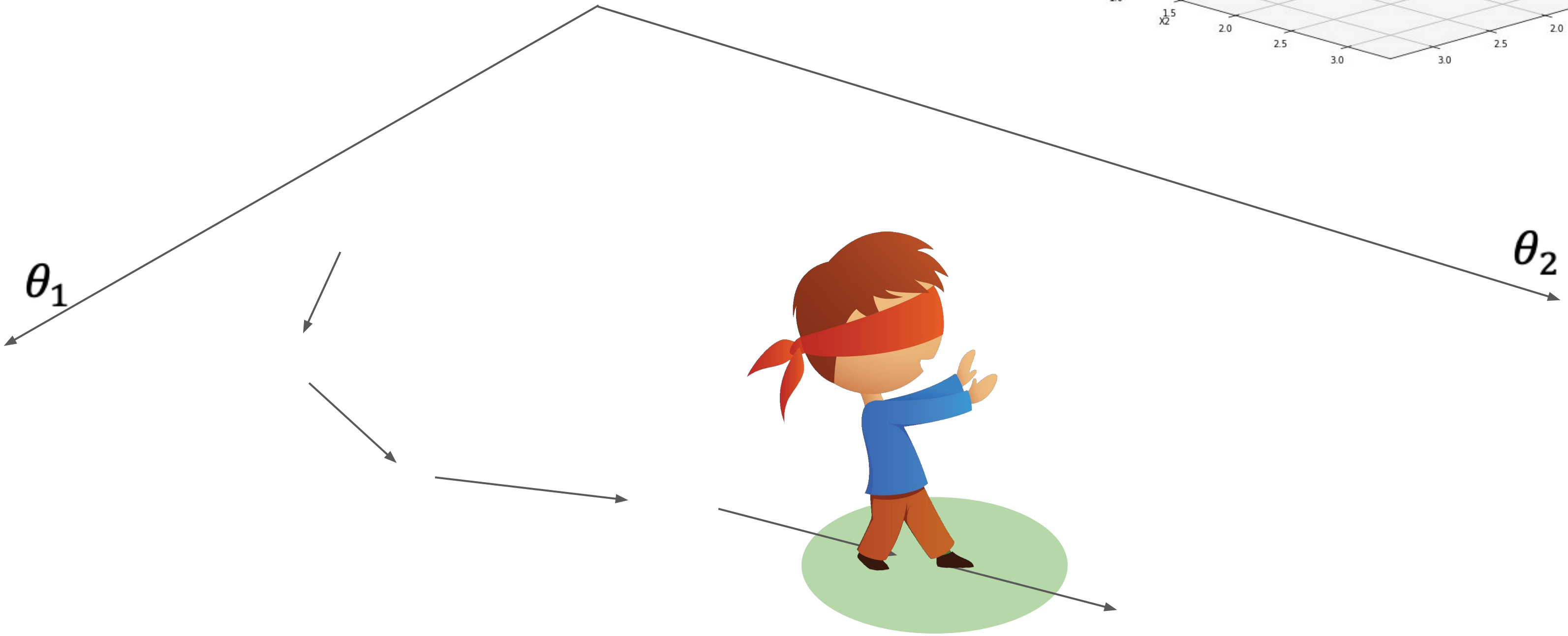

θ_1
θ_2

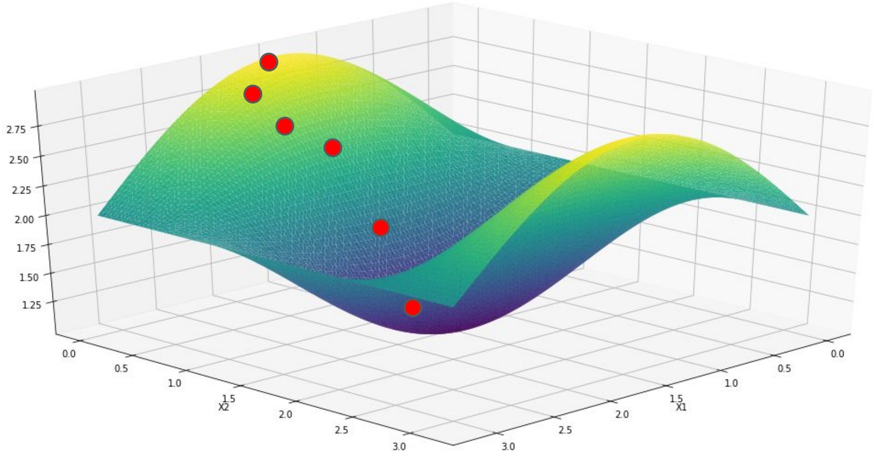

X2
X1

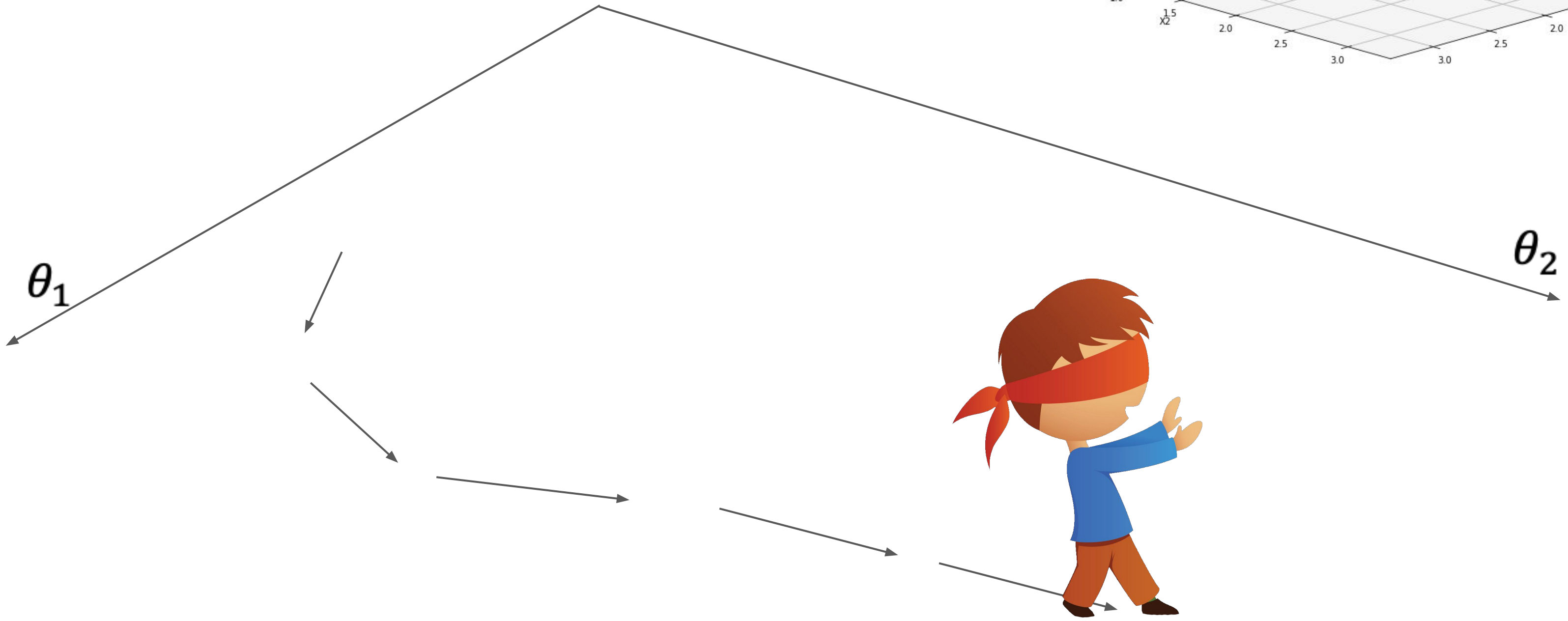

θ_1
θ_2

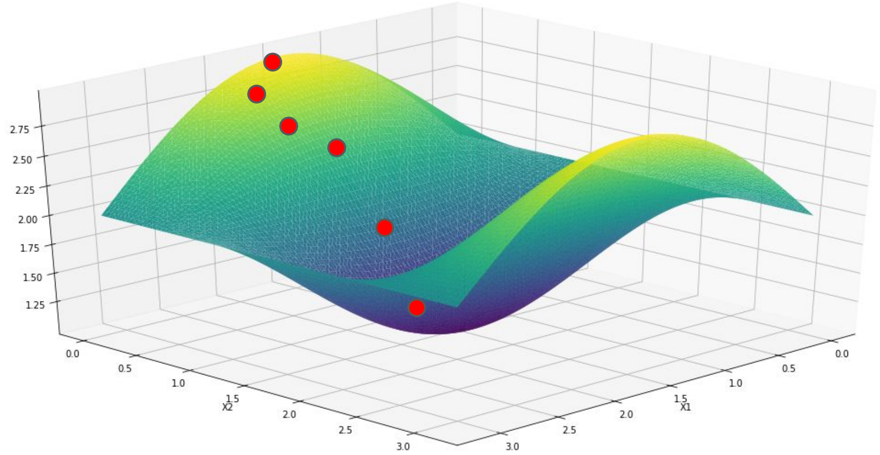
Skillbox

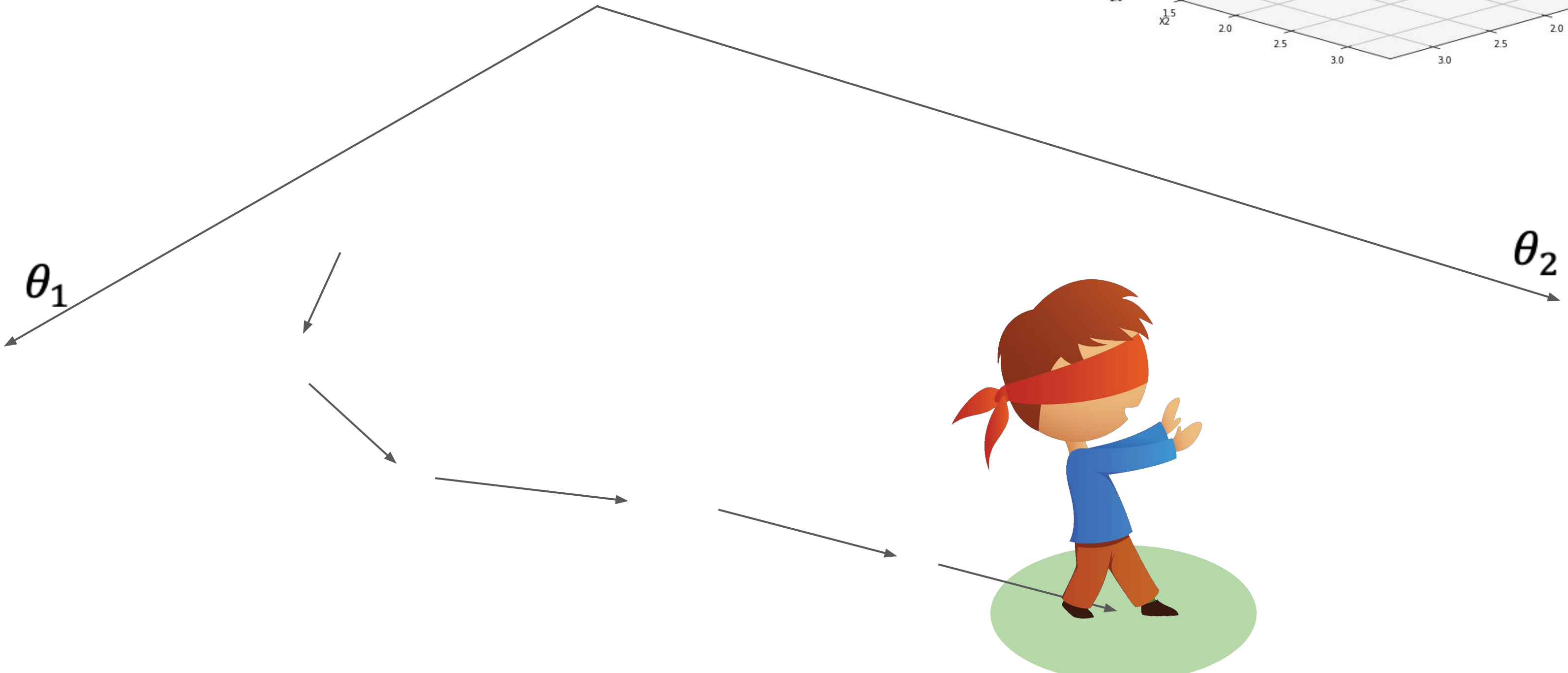

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
2.0
2.5
3.0
x2
x1


θ_1
θ_2

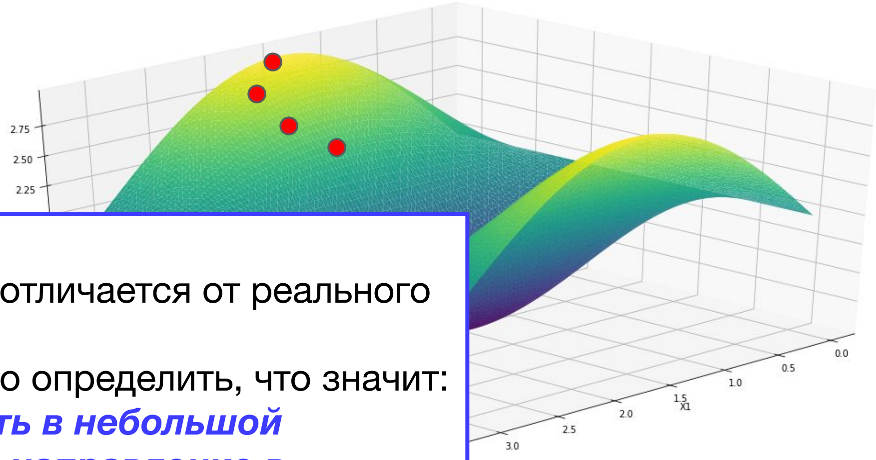

2.75
2.50
2.25
2.00
1.75
1.50
1.25
0.0
0.5
1.0
1.5
X2
2.0
2.5
3.0
0.0
0.5
1.0
1.5
X1
2.0
2.5
3.0


θ_1
θ_2

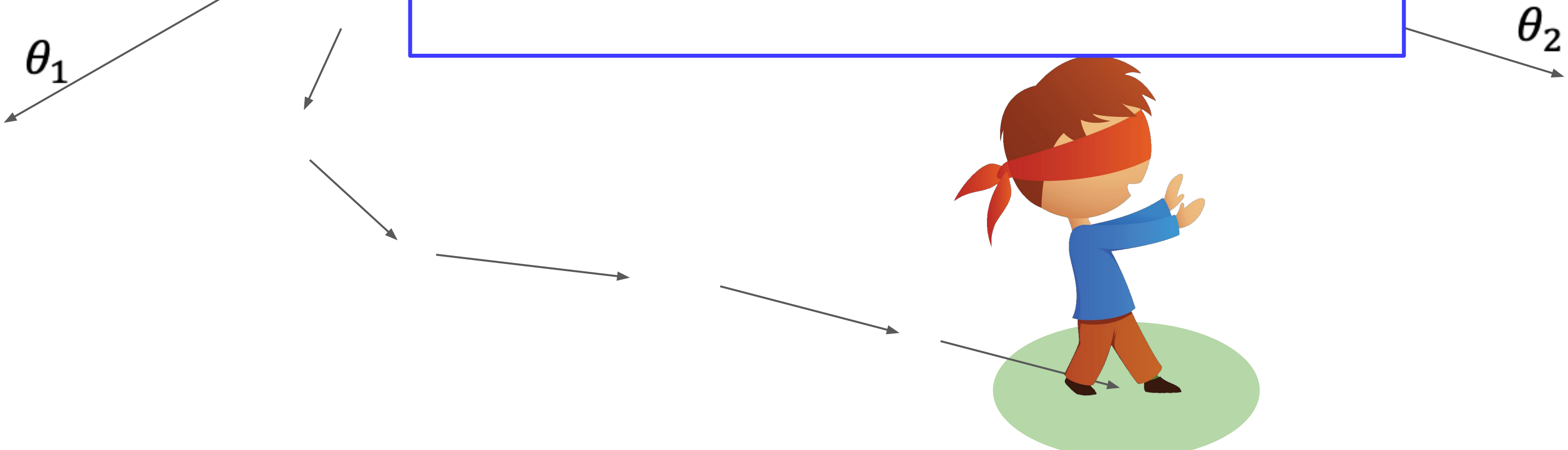

X2
X1


θ_1
θ_2

Skillbox



- Это описание не сильно отличается от реального алгоритма оптимизации!
- Нужно только формально определить, что значит: ***“прощупать поверхность в небольшой окрестности и выбрать направление в котором наблюдается наибольшее уменьшение высоты”***



$$\theta_0$$

$$\Delta\theta$$

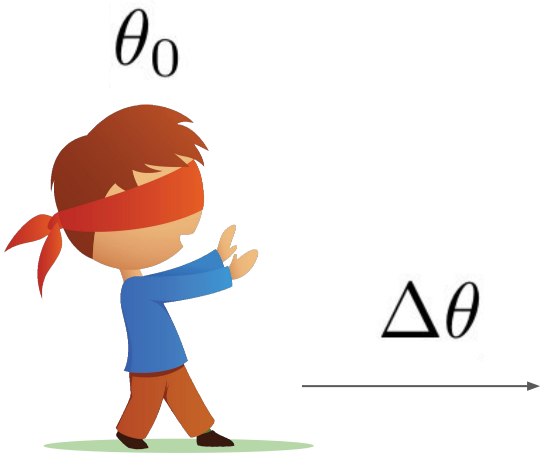
$$\theta_0$$

$$\Delta\theta$$

$$\frac{f(\theta_0 + \Delta\theta) - f(\theta_0)}{\Delta\theta}$$

- > 0 : если пойти прямо, то значение функции увеличится
- < 0 : значение уменьшится
- **Нам нужно идти в сторону уменьшения**

$$\frac{f(\theta_0 + \Delta\theta) - f(\theta_0)}{\Delta\theta}$$



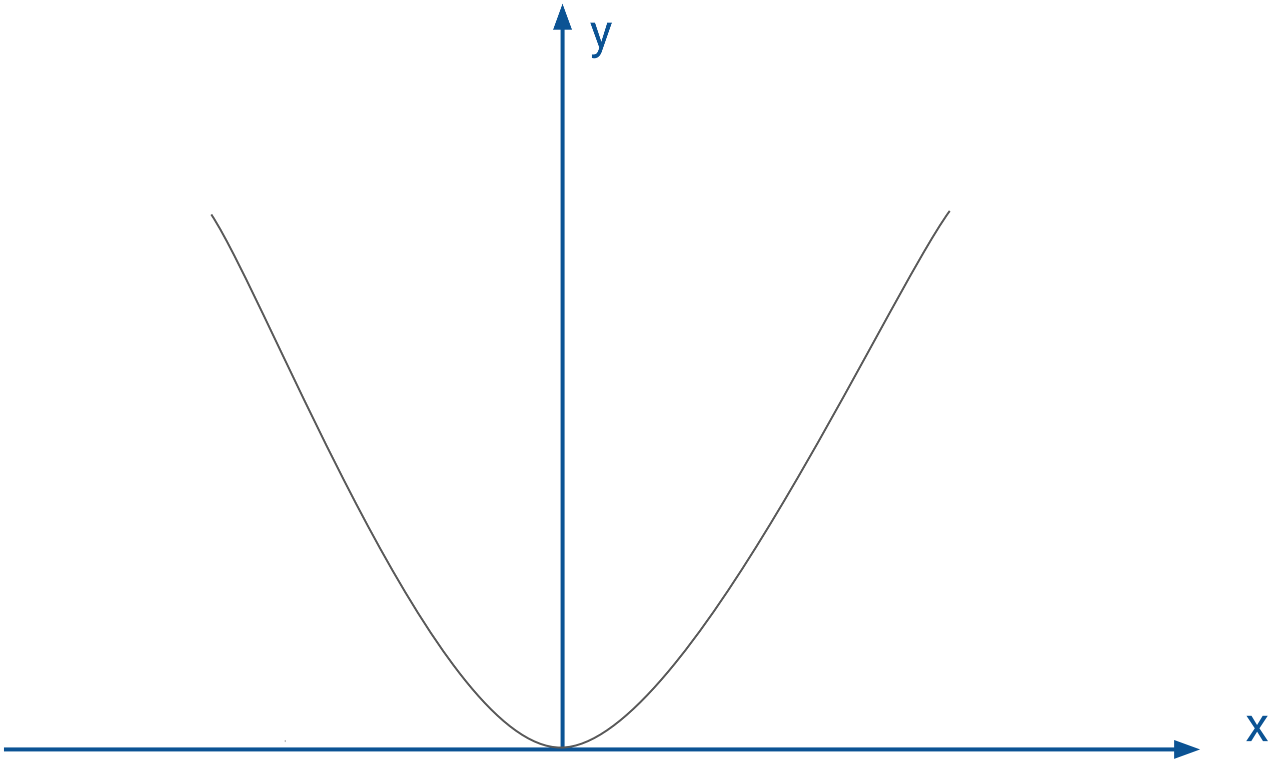
$$f'(\theta_0) = \lim_{\Delta\theta \to 0} \frac{f(\theta_0 + \Delta\theta) - f(\theta_0)}{\Delta\theta}$$

Skillbox

Как найти минимум?

$$y = x^2$$

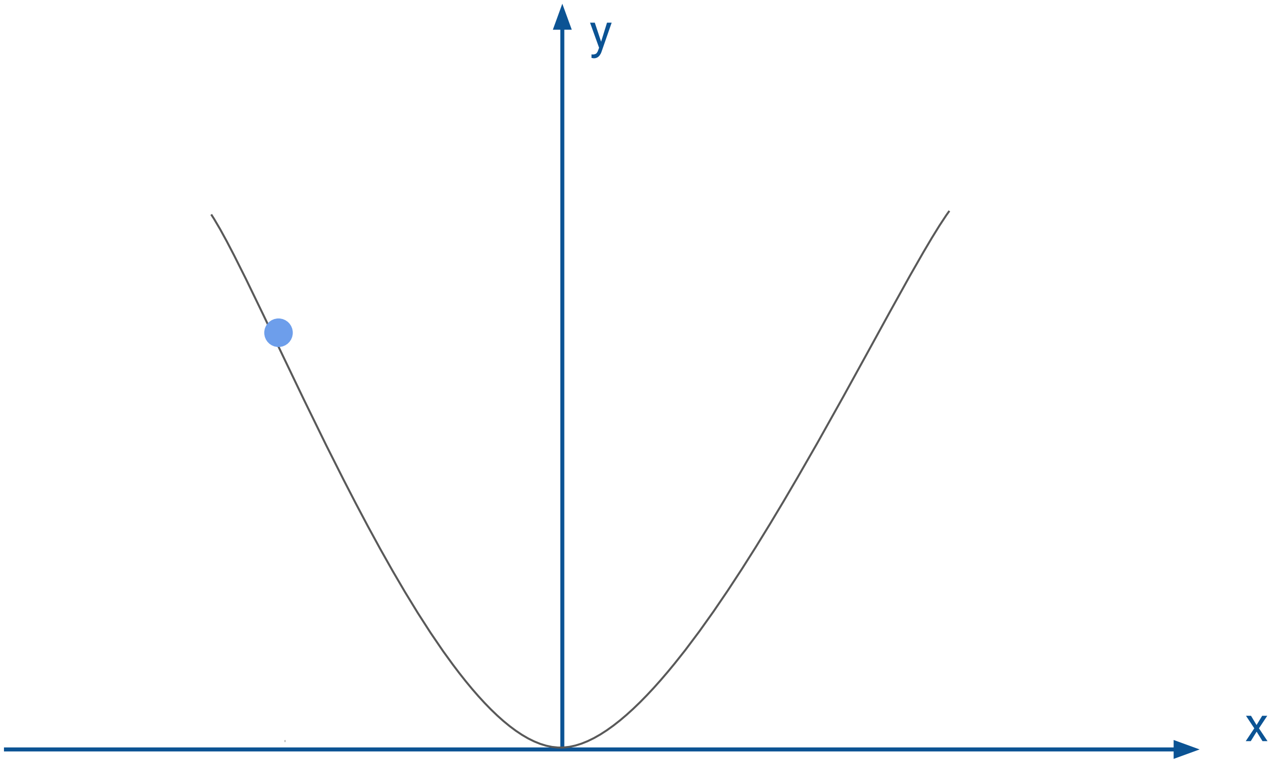
$$y' = 2x$$



Как найти минимум?

$$y = x^2$$

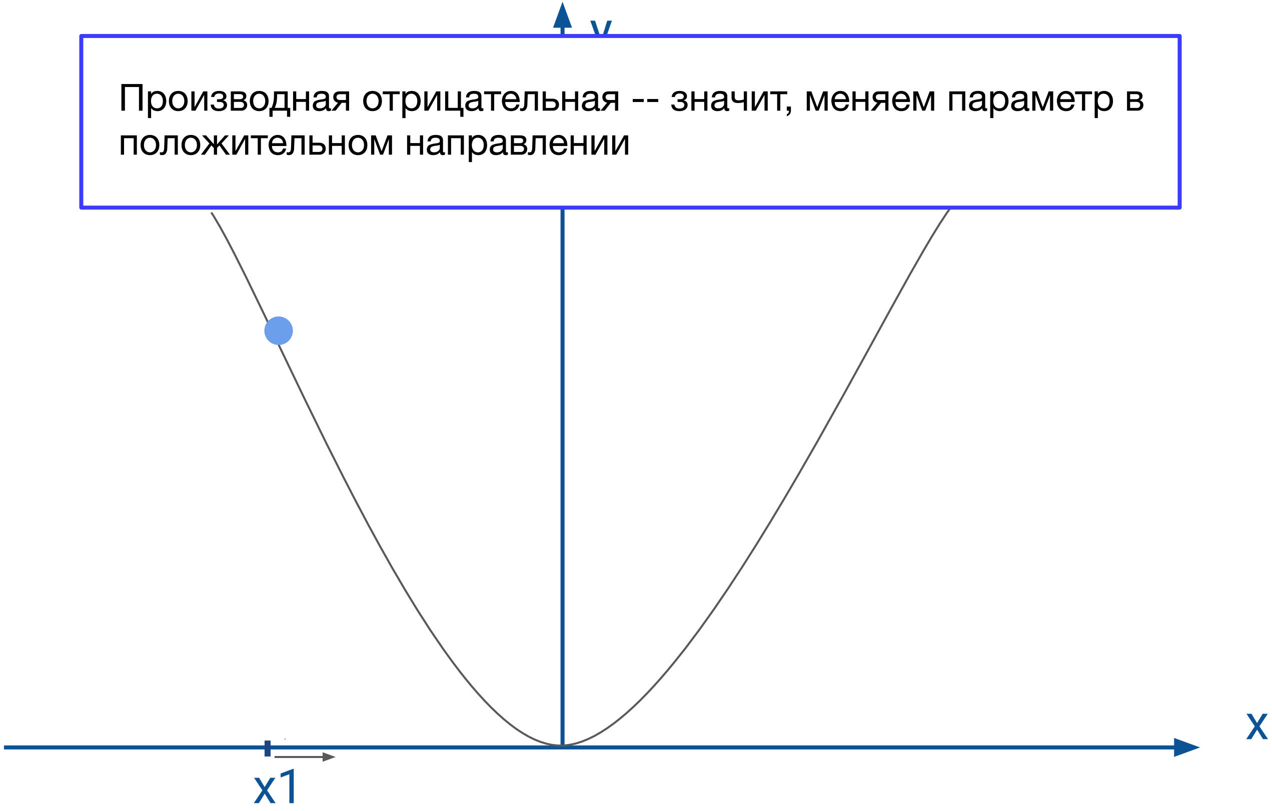
$$y' = 2x$$



Как найти минимум?

$$y = x^2$$

$$y' = 2x$$

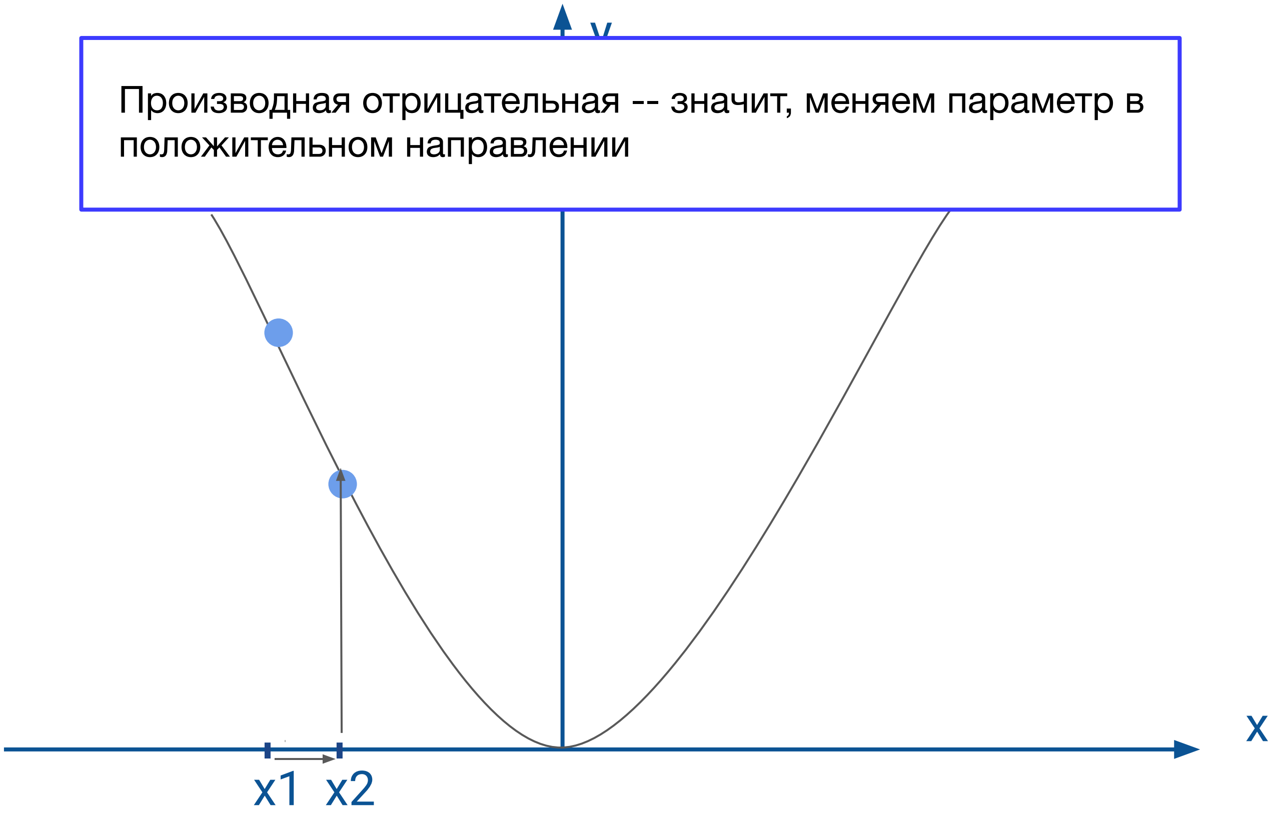


Производная отрицательная -- значит, меняем параметр в положительном направлении

Как найти минимум?

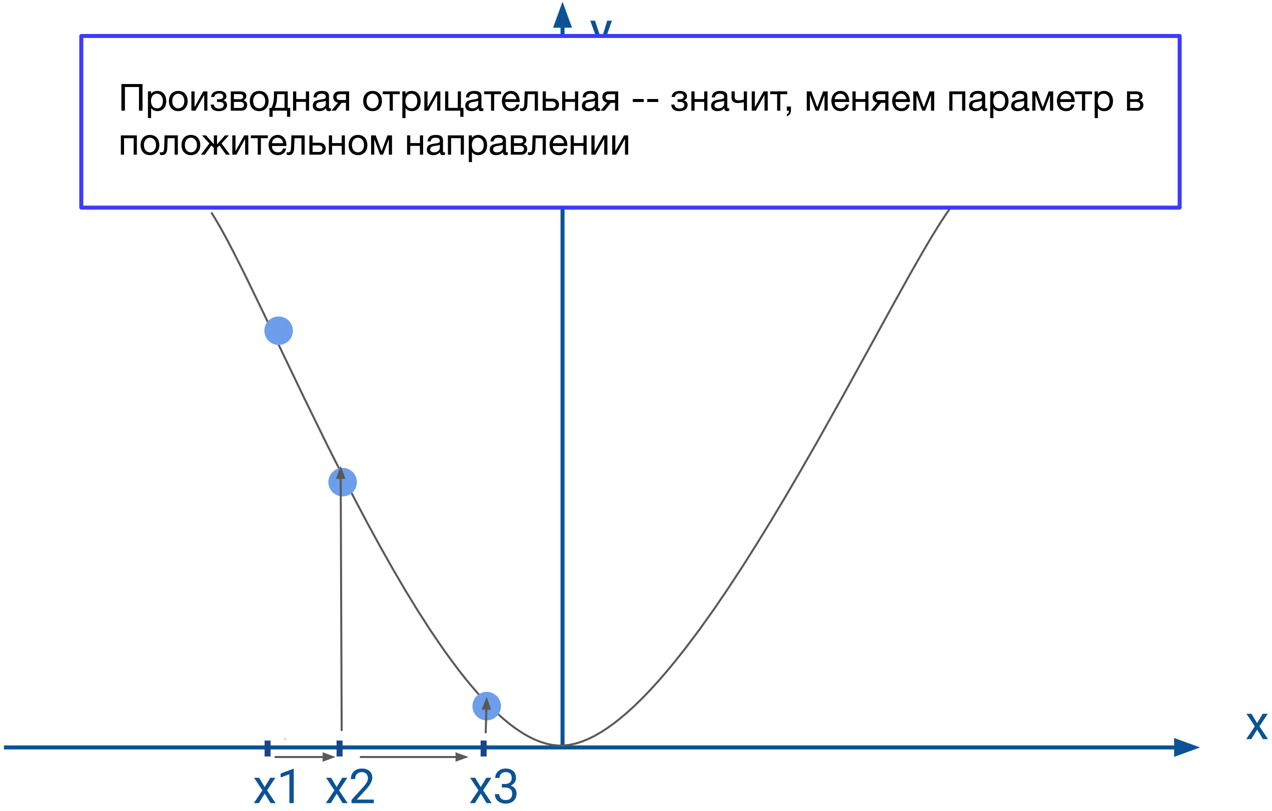
$$y = x^2$$
$$y' = 2x$$

Производная отрицательная -- значит, меняем параметр в положительном направлении



Как найти минимум?

$$y = x^2$$
$$y' = 2x$$



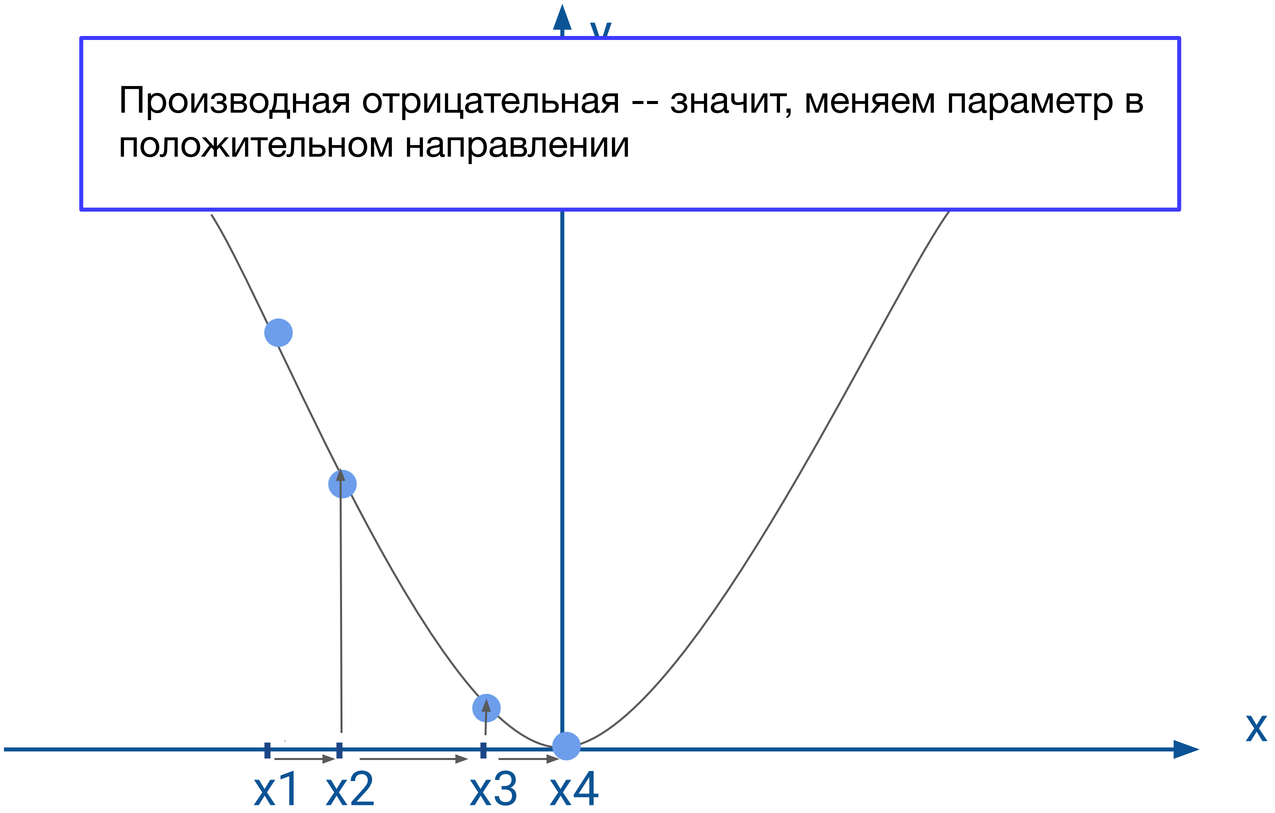
Производная отрицательная -- значит, меняем параметр в положительном направлении

Как найти минимум?

$$y = x^2$$

$$y' = 2x$$

Производная отрицательная -- значит, меняем параметр в положительном направлении

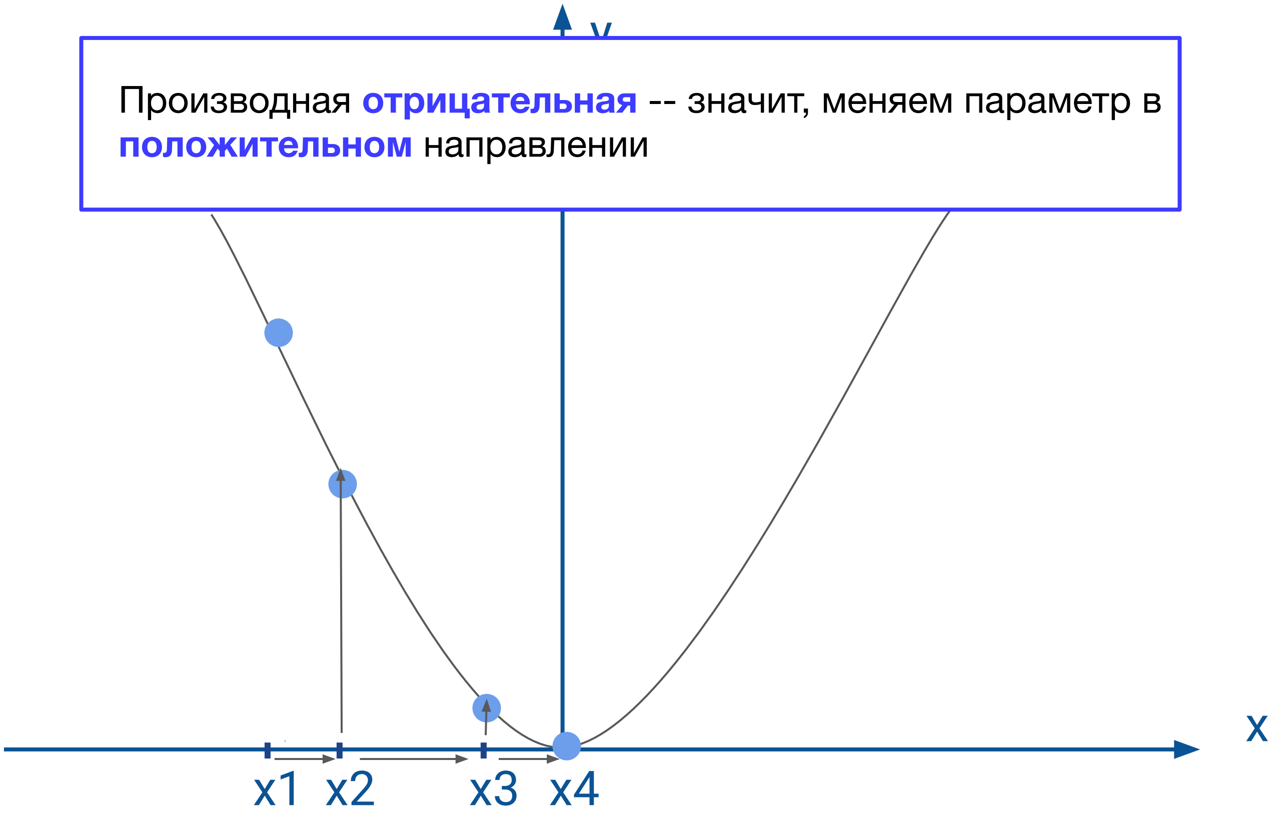


Как найти минимум?

$$y = x^2$$

$$y' = 2x$$

Производная **отрицательная** -- значит, меняем параметр в **положительном** направлении



Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:
$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$
   a. **После этого шага параметры изменились!**
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:

$$\theta_i = \theta_{i-1} - \boxed{\lambda} \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$

Скорость обучения
(learning rate)

   a. **После этого шага параметры изменились!**
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:
$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$
   a. **После этого шага параметры изменились!**
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:

$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$

   a. **После этого шага параметры изменились!**

5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

** градиент -- вектор из производных по каждому из параметров (для многомерного случая)*

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:

$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$

   a. **После этого шага параметры изменились!**

5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

“Обучение” алгоритма

** градиент -- вектор из производных по каждому из параметров (для многомерного случая)*

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:

$$\boxed{\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})}$$

   a. **После этого шага параметры изменились!**

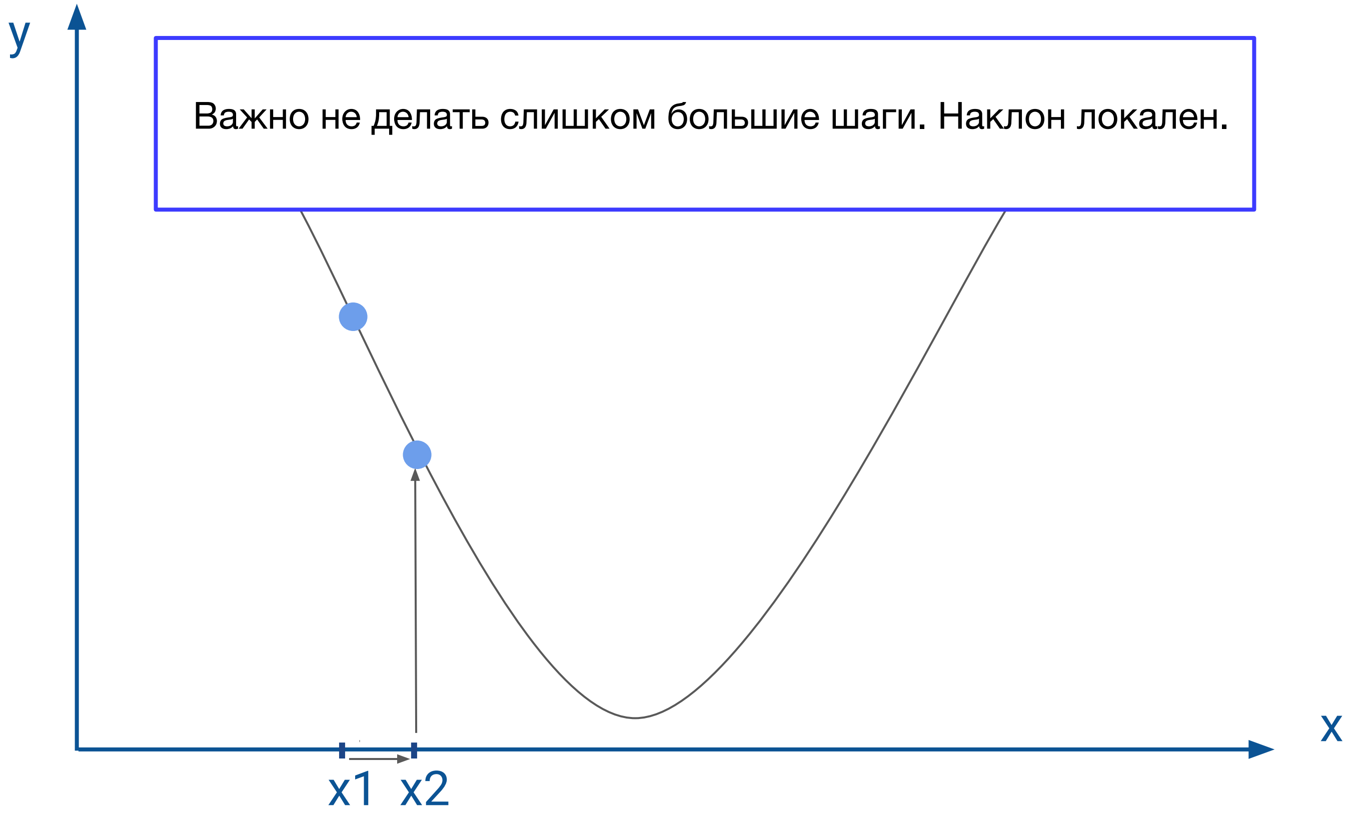
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

"Обучение" алгоритма
~
Настройка параметров

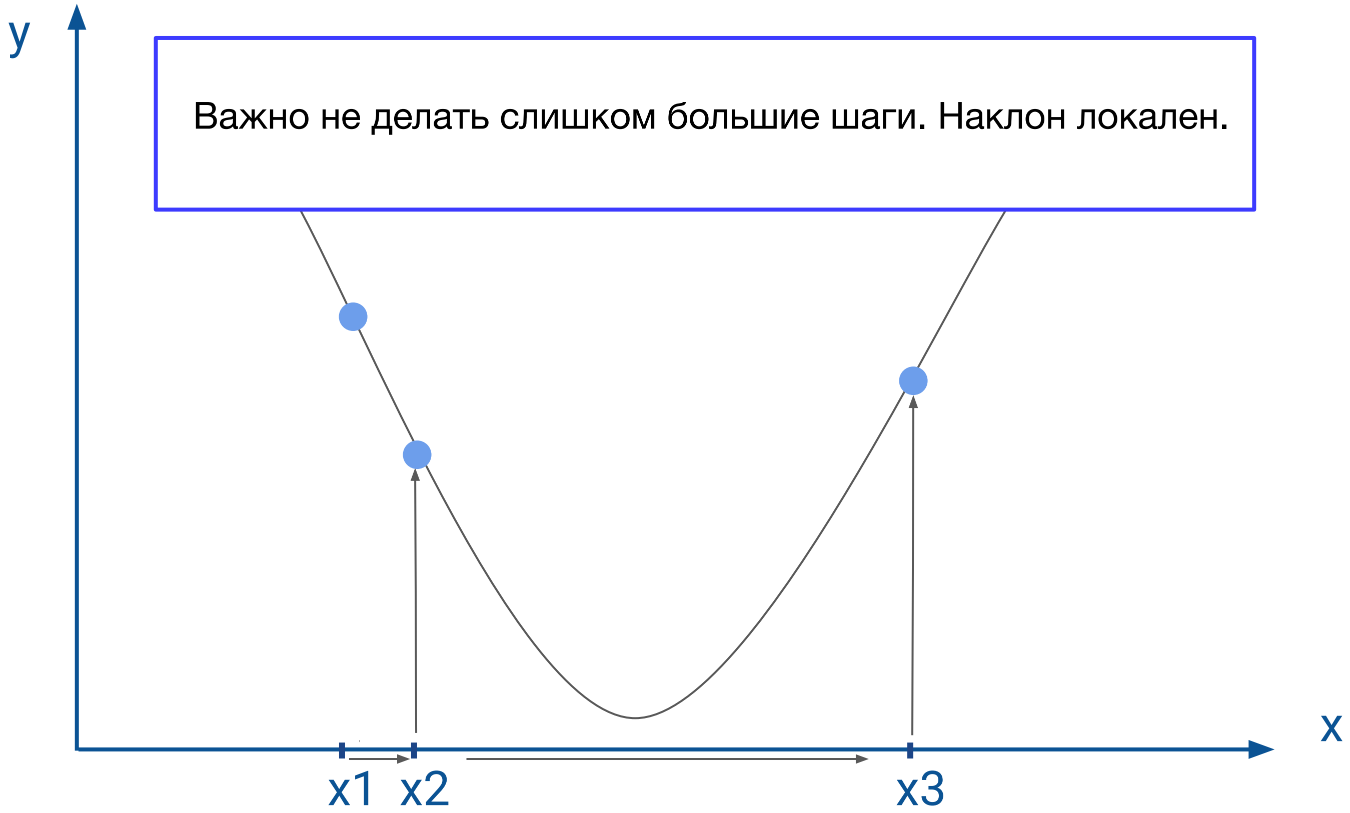
** градиент -- вектор из производных по каждому из параметров (для многомерного случая)*

Skillbox

Скорость обучения



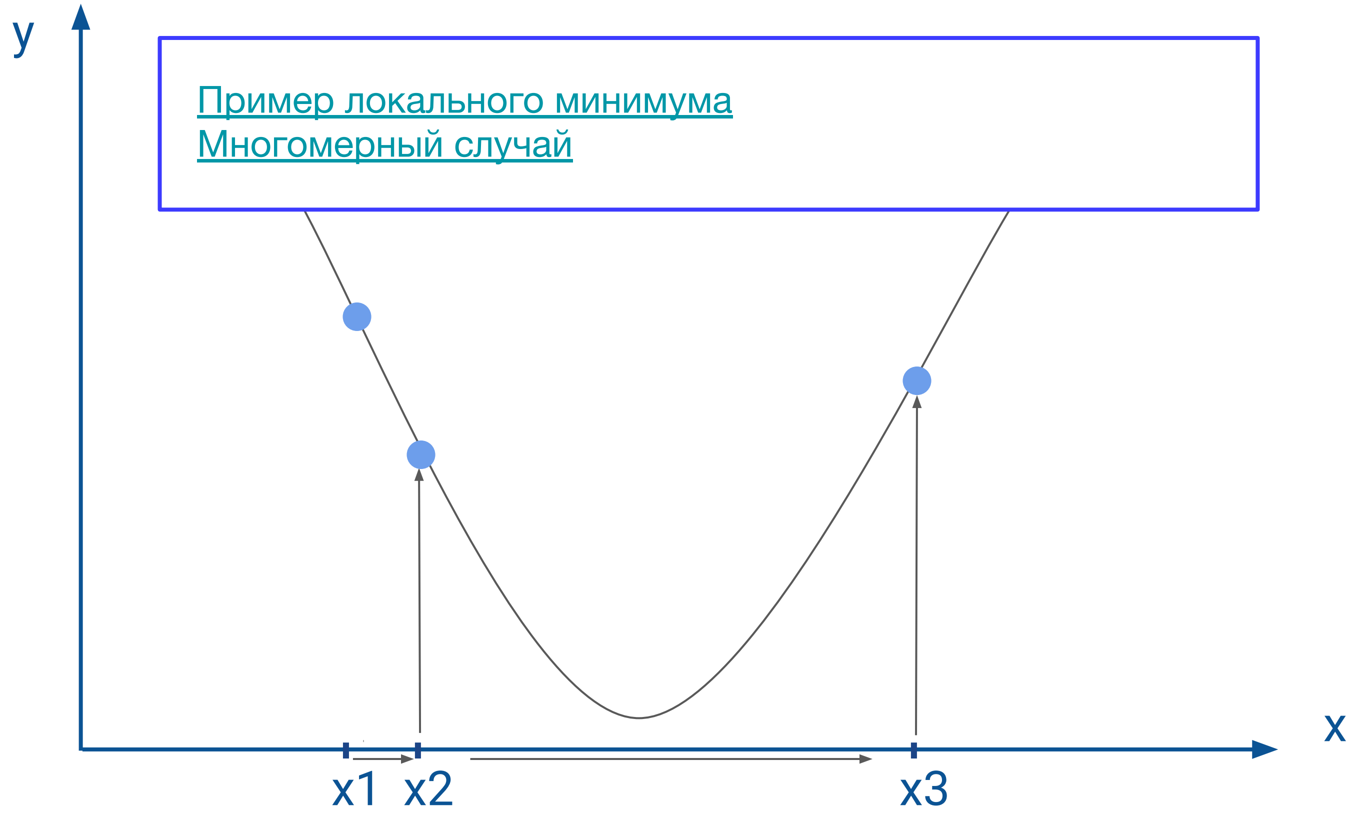
Скорость обучения



Skillbox

Визуализация

Пример локального минимума
Многомерный случай



Градиентный спуск

- Метод прост в реализации (вы напишите его сами в практическом задании)
- Именно он (с некоторыми модификациями) используется сегодня для оптимизации нейронных сетей

Градиентный спуск

- Метод прост в реализации (вы напишите его сами в практическом задании)
- Именно он (с некоторыми модификациями) используется сегодня для оптимизации нейронных сетей

Важные ограничения метода:

- Мы на всех этапах пользовались требованием дифференцируемости функции (без этого мы бы не могли посчитать производную)

Градиентный спуск

- Метод прост в реализации (вы напишите его сами в практическом задании)
- Именно он (с некоторыми модификациями) используется сегодня для оптимизации нейронных сетей

Важные ограничения метода:

- Мы на всех этапах пользовались требованием дифференцируемости функции (без этого мы бы не могли посчитать производную).
- Нахождение глобального минимума не гарантировано.
- Одной их самых серьезных проблем данного подхода является застревание в локальных минимумах
- Метод очень чувствителен к инициализации

Skillbox

Обучение нейронных сетей

Метод обратного распространения ошибки

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:
$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$
    a. **После этого шага параметры изменились!**
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L}$
3. $\boxed{\text{Вычислить } \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1}) \text{ при текущем значении параметра}}$
4. Сделать шаг оптимизации:
$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$
   a. **После этого шага параметры изменились!**
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

Вычисление производных

- Нейронная сеть: 10 -> 512 -> 256 -> 1
- Содержит около миллиона параметров
- Наивный подход:
  - Посчитать производные для каждого параметра в общем виде
  - Подставлять текущие значения параметров для вычисления производной
- Для одного шага градиентного спуска мы вычислим 1 000 000 раз значения для функций сравнимых по сложности с исходной!
- Квадратичная сложность шага

Вычисление производных

- Нейронная сеть: 10 -> 512 -> 256 -> 1
- Содержит около м
- Наивный подход:
    - Посчитать п
      виде
    - Подставлять
      вычисления
- Для одного шага г
  раз значения для функции сравнимых по сложности с исходной!
- Квадратичная сложность шага

Back propagation может это делать линейно!
С помощью Chain Rule

**(Back propagation -- подход к вычислению производных, обучение происходит методом градиентного спуска)*

Chain Rule

$$f(x, y, z) = (2x + y)z$$

Найти: $\frac{\partial f}{\partial x} \ \frac{\partial f}{\partial y} \ \frac{\partial f}{\partial z}$

В точке: $x = 10, y = -2, z = -4$

Chain Rule

$$f(x, y, z) = (2x + y)z$$

Найти: $\frac{\partial f}{\partial x}$ $\frac{\partial f}{\partial y}$ $\frac{\partial f}{\partial z}$

В точке: $x = 10, y = -2, z = -4$

$$\frac{\partial f}{\partial x} = 2z$$

$$\frac{\partial f}{\partial y} = z$$

$$\frac{\partial f}{\partial z} = 2x + y$$

Chain Rule

$$f(x, y, z) = (2x + y)z$$

Найти: $\frac{\partial f}{\partial x}$ $\frac{\partial f}{\partial y}$ $\frac{\partial f}{\partial z}$

В точке: $x = 10, y = -2, z = -4$

$$\frac{\partial f}{\partial x} = 2z = -8$$

$$\frac{\partial f}{\partial y} = z = -4$$

$$\frac{\partial f}{\partial z} = 2x + y = 18$$

Chain Rule

$$f(x, y, z) = (2x + y)z$$

Найти: $\dfrac{\partial f}{\partial x}\ \dfrac{\partial f}{\partial y}\ \dfrac{\partial f}{\partial z}$

В точке: $x = 10, y = -2, z = -4$

$$\frac{\partial f}{\partial x} = \boxed{2}z = -8$$

$$\frac{\partial f}{\partial y} = z = -4$$

$$\frac{\partial f}{\partial z} = 2x + y = 18$$

Chain Rule

$$f(x, y, z) = (2x + y)z$$

Найти: $\frac{\partial f}{\partial x} \; \frac{\partial f}{\partial y} \; \frac{\partial f}{\partial z}$

В точке: $x = 10, y = -2, z = -4$

$$\frac{\partial f}{\partial x} = \boxed{2}z = -8$$

$$\frac{\partial f}{\partial y} = z = -4$$

$$\frac{\partial f}{\partial z} = 2x + y = 18$$

$$\boxed{\frac{\partial f(g(x))}{\partial x} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial x}}$$

Chain Rule

$$f(x, y, z) = (2x + y)z$$

Найти: $\dfrac{\partial f}{\partial x}\ \dfrac{\partial f}{\partial y}\ \dfrac{\partial f}{\partial z}$

В точке: $x = 10, y = -2, z = -4$

$$\frac{\partial f}{\partial x} = \boxed{2}z = -8$$

$$\frac{\partial f}{\partial y} = z = -4$$

$$\frac{\partial f}{\partial z} = 2x + y = 18$$

z

2

$$\boxed{\frac{\partial f(g(x))}{\partial x} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial x}}$$

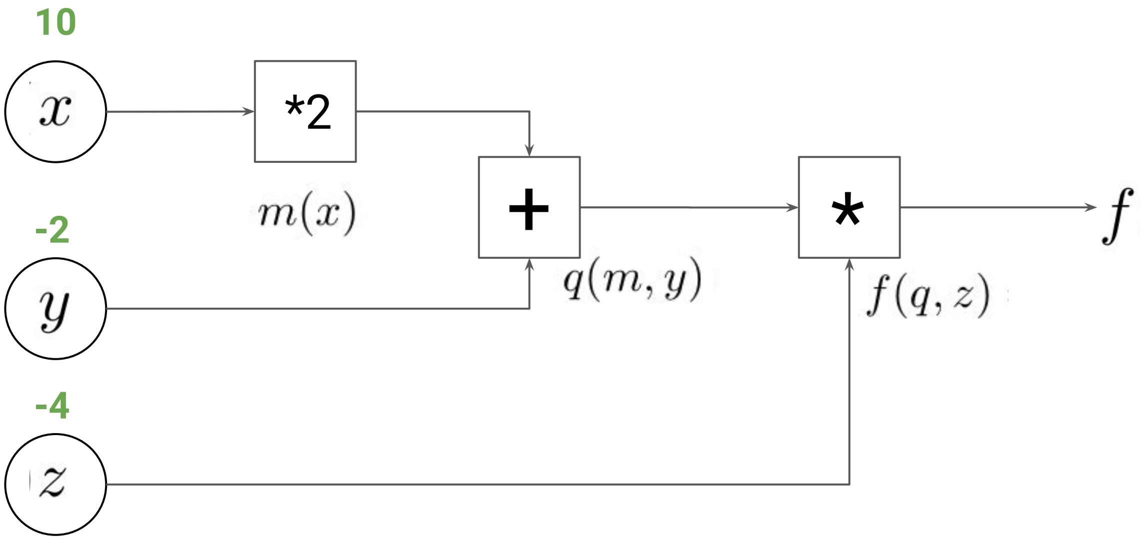
$$f(g, z) = gz$$

$$g(x, y) = 2x + y$$

Вычислительный граф

$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$



$$m(x) = 2x$$

$$q(m, y) = m + y$$

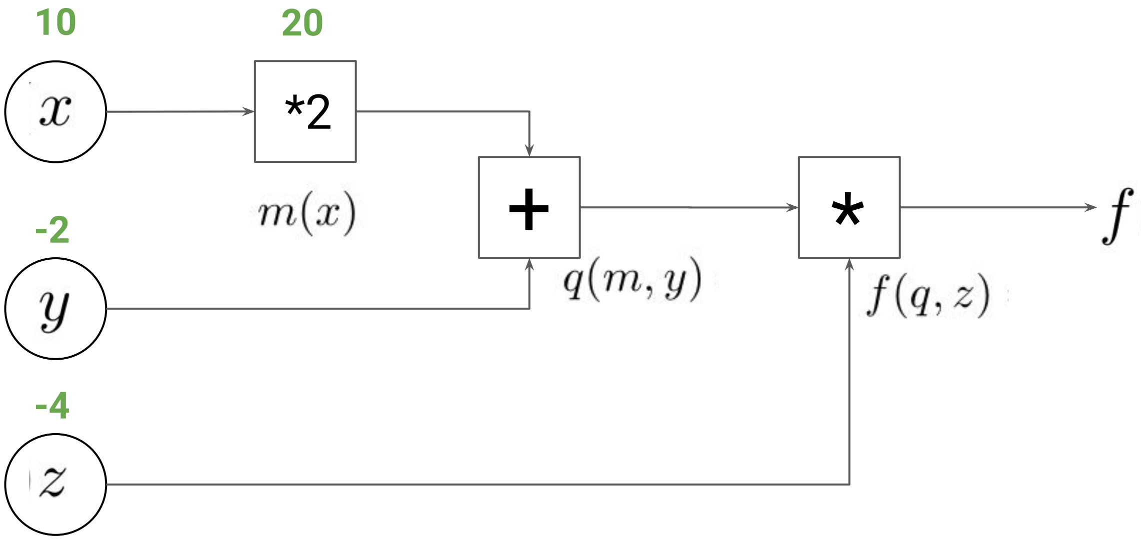
$$f(q, z) = qz$$

Вычислительный граф

Forward pass

$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$



$$m(x) = 2x$$

$$q(m, y) = m + y$$

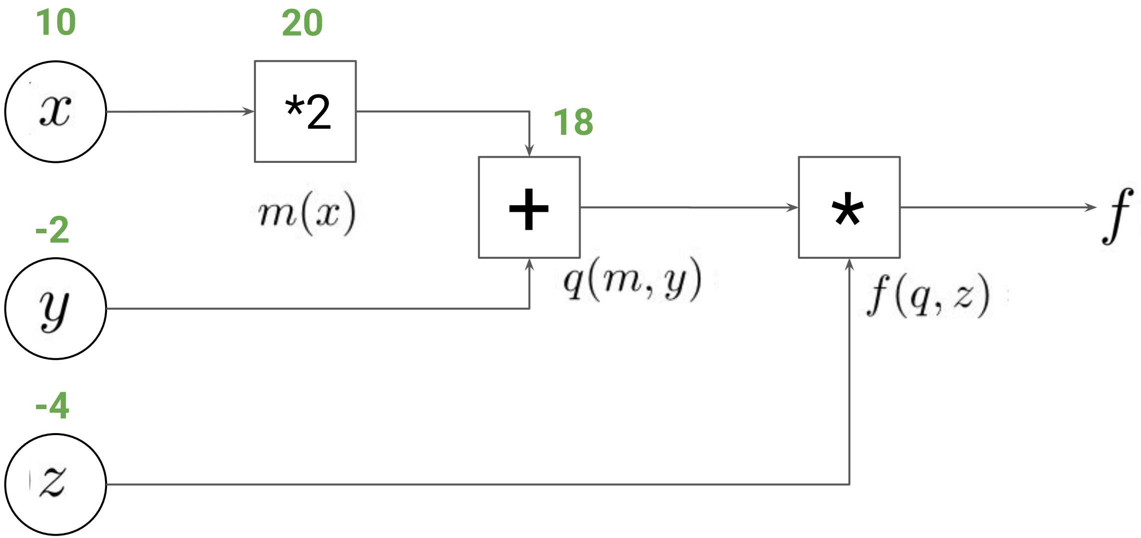
$$f(q, z) = qz$$

Вычислительный граф

Forward pass

$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$



$$m(x) = 2x$$

$$q(m, y) = m + y$$

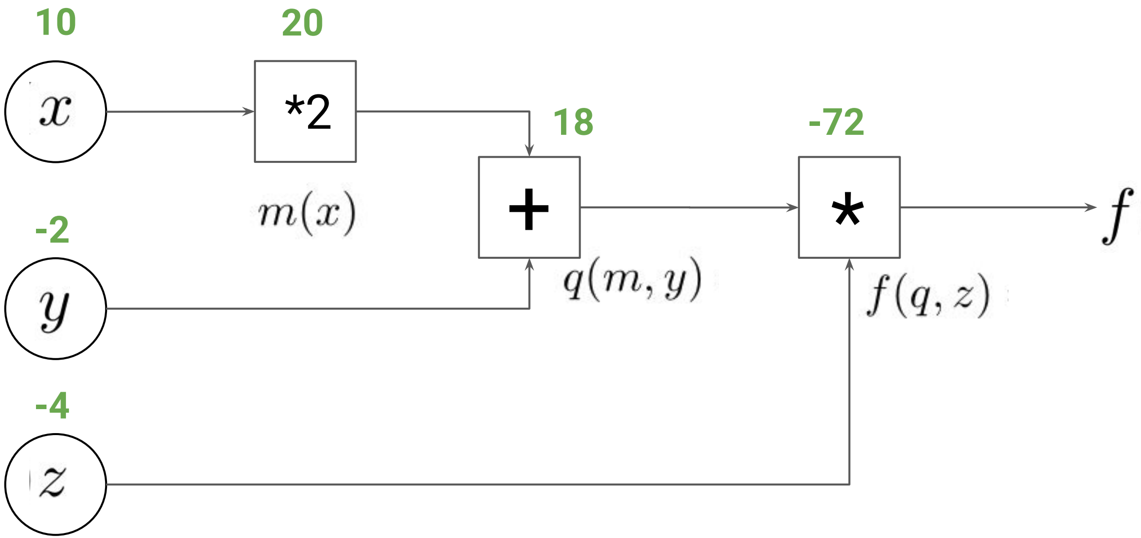
$$f(q, z) = qz$$

Вычислительный граф

Forward pass

$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$



$$m(x) = 2x$$

$$q(m, y) = m + y$$

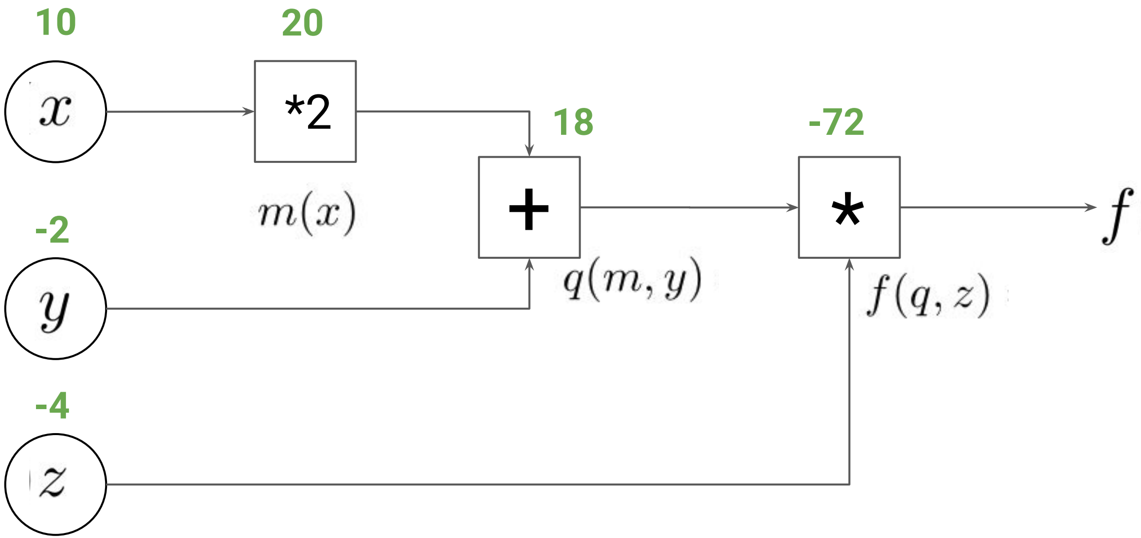
$$f(q, z) = qz$$

Вычислительный граф

Backward pass

$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$



$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$

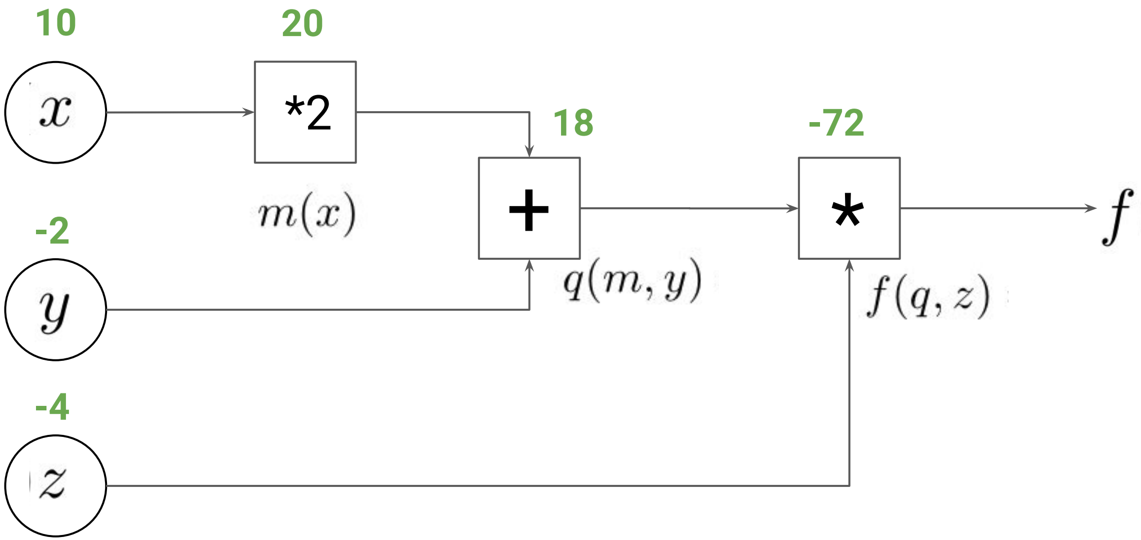
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



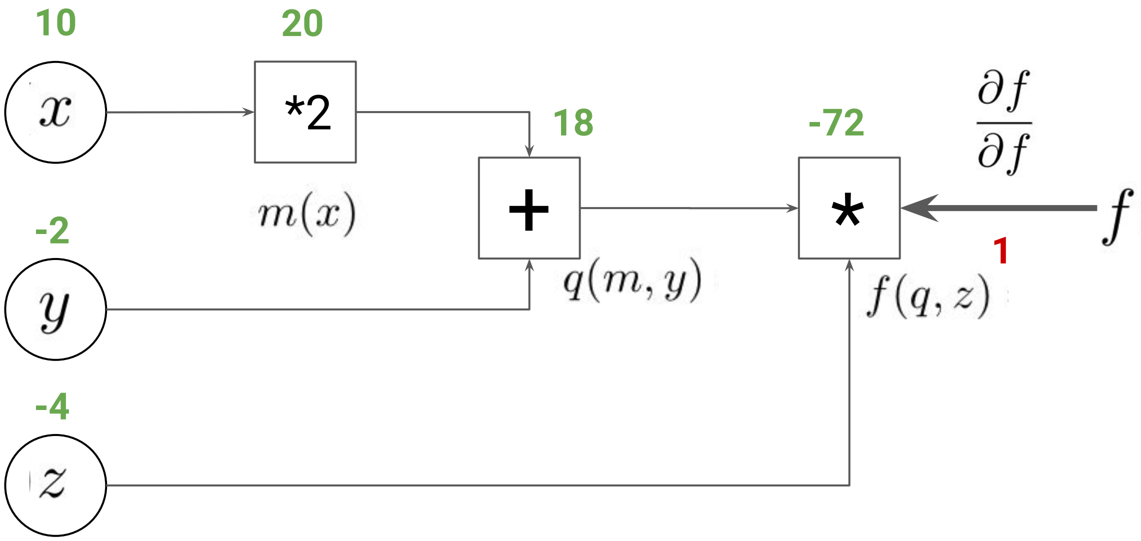
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



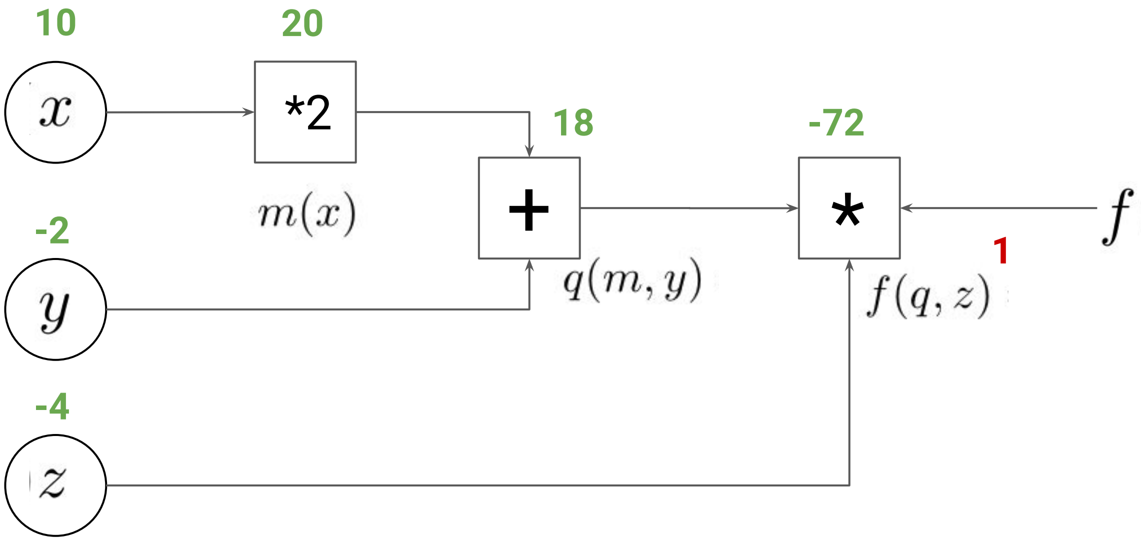
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



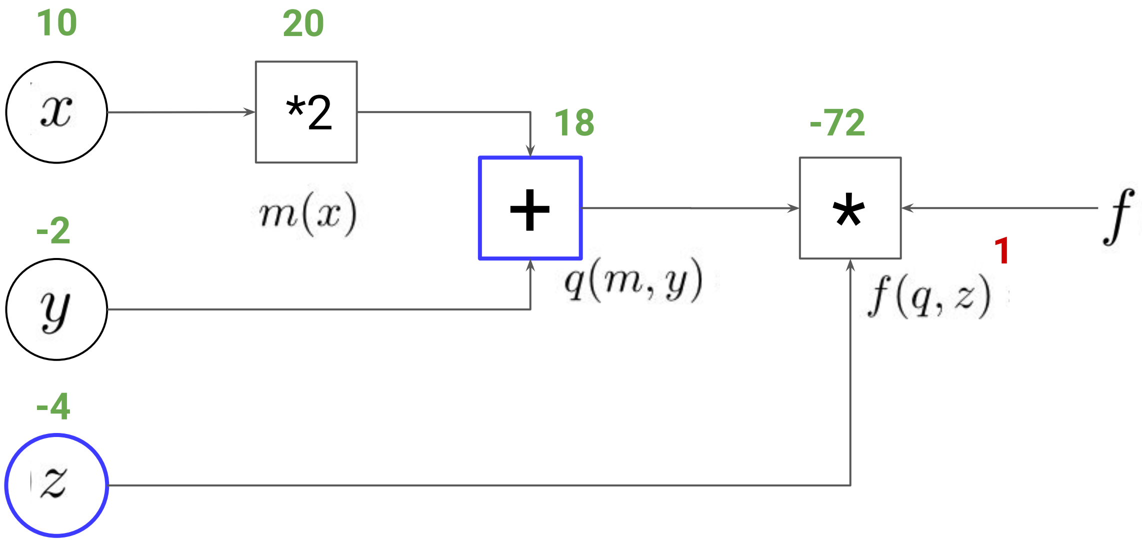
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



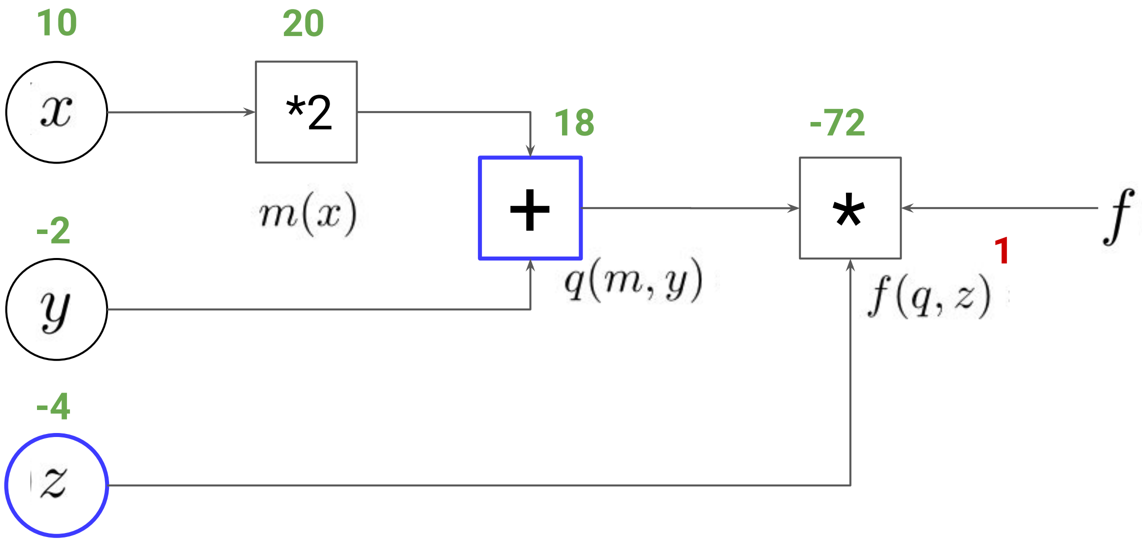
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial z} = q$$

$$\frac{\partial f}{\partial q} = z$$

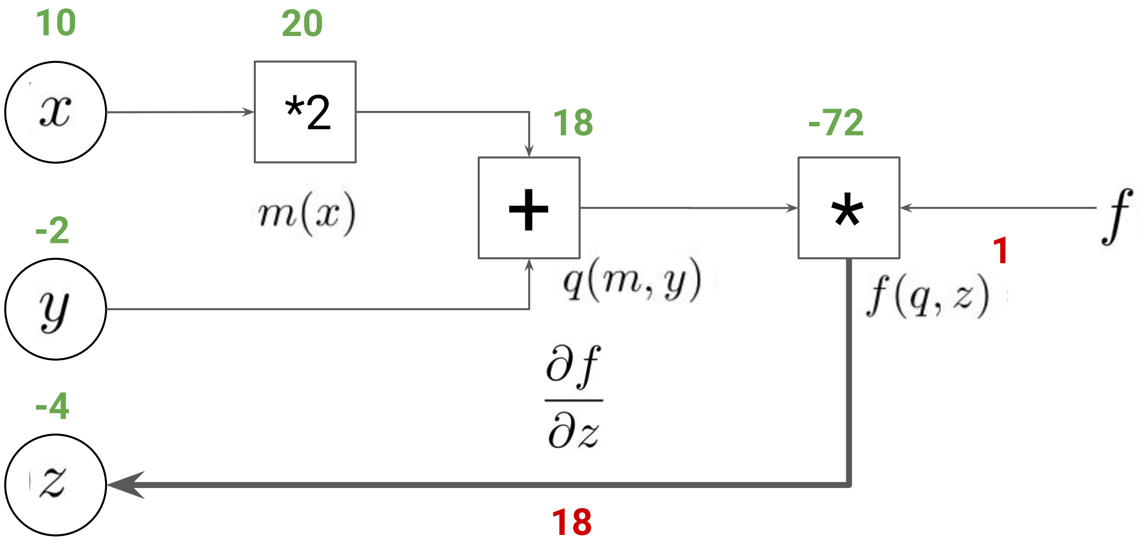
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial z} = q$$

$$\frac{\partial f}{\partial q} = z$$

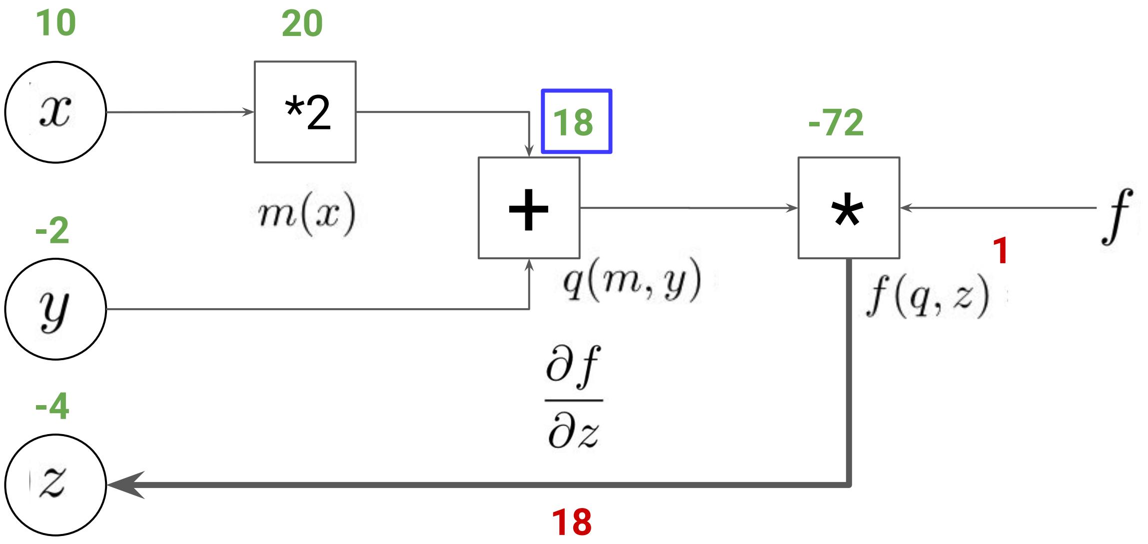
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial z} = q$$

$$\frac{\partial f}{\partial q} = z$$

Мы не считали это значение!
Оно нам известно!

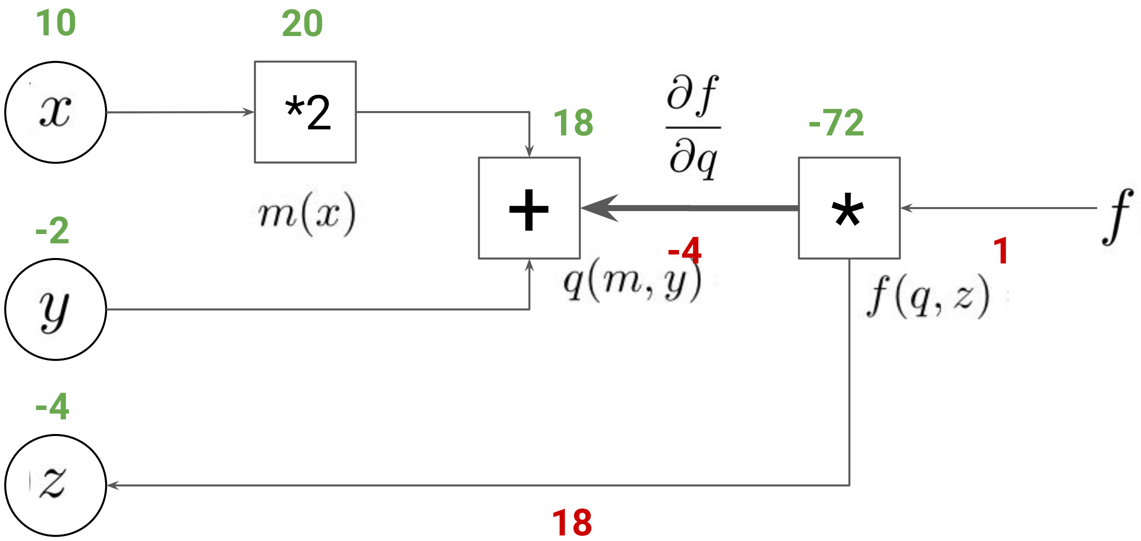
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial z} = q$$

$$\frac{\partial f}{\partial q} = z$$

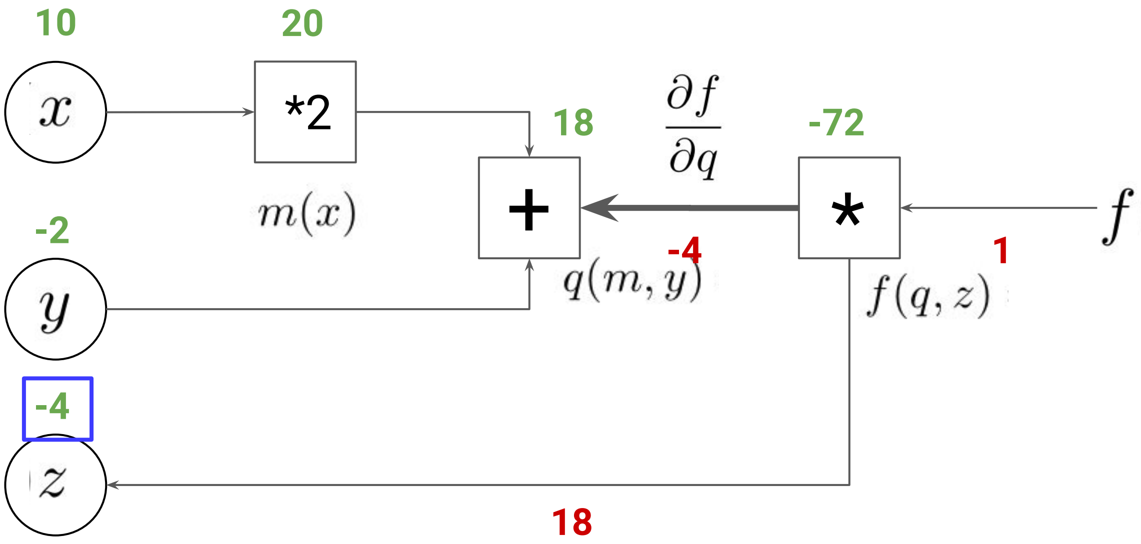
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial z} = q$$

$$\frac{\partial f}{\partial q} = z$$

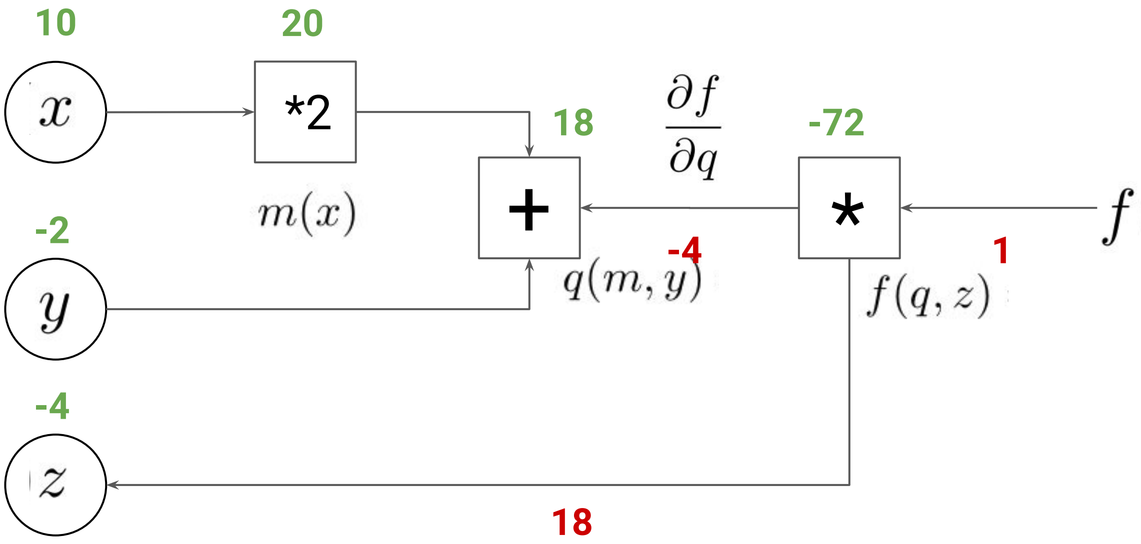
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



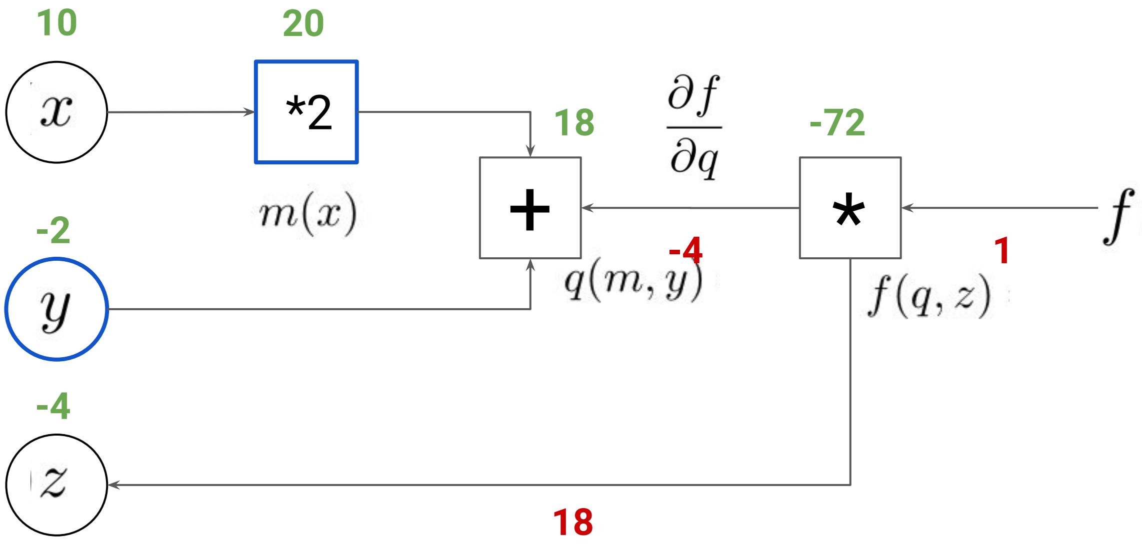
$$f(x,y,z) = (2x+y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m,y) = m + y$$

$$f(q,z) = qz$$



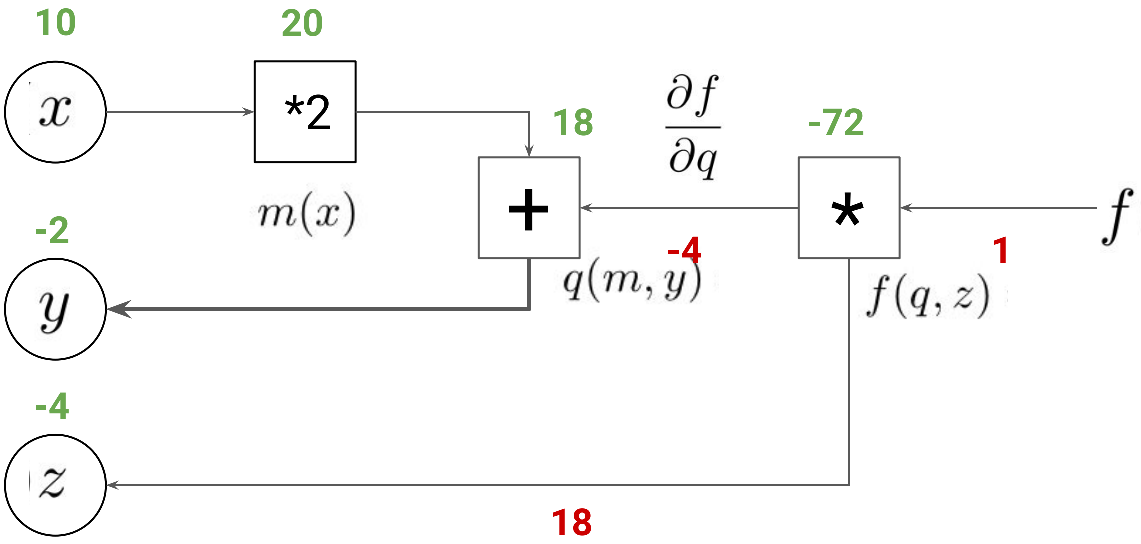
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial y}$$

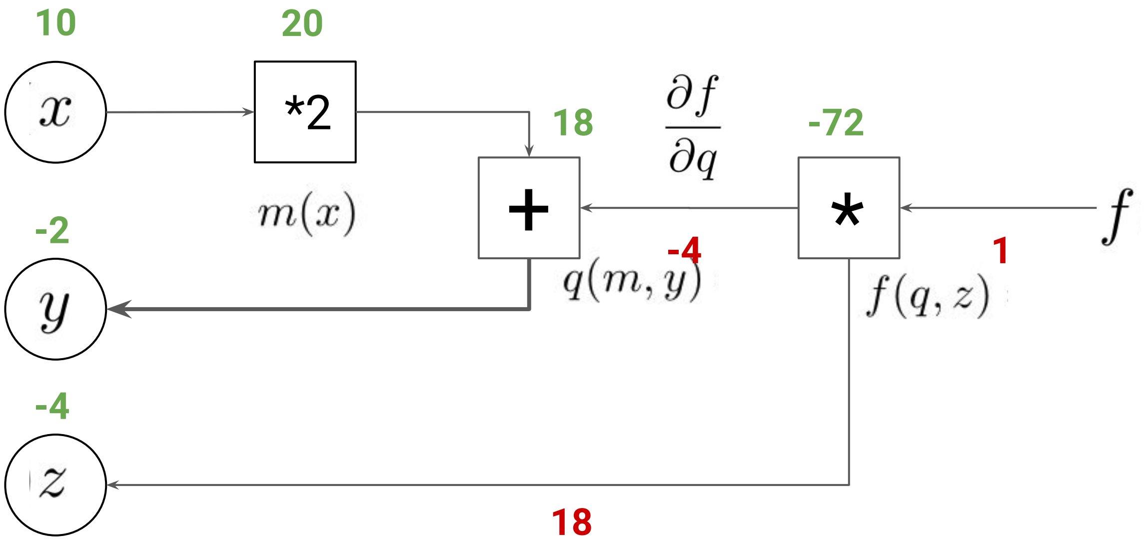
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q}\boxed{\frac{\partial q}{\partial y}}$$

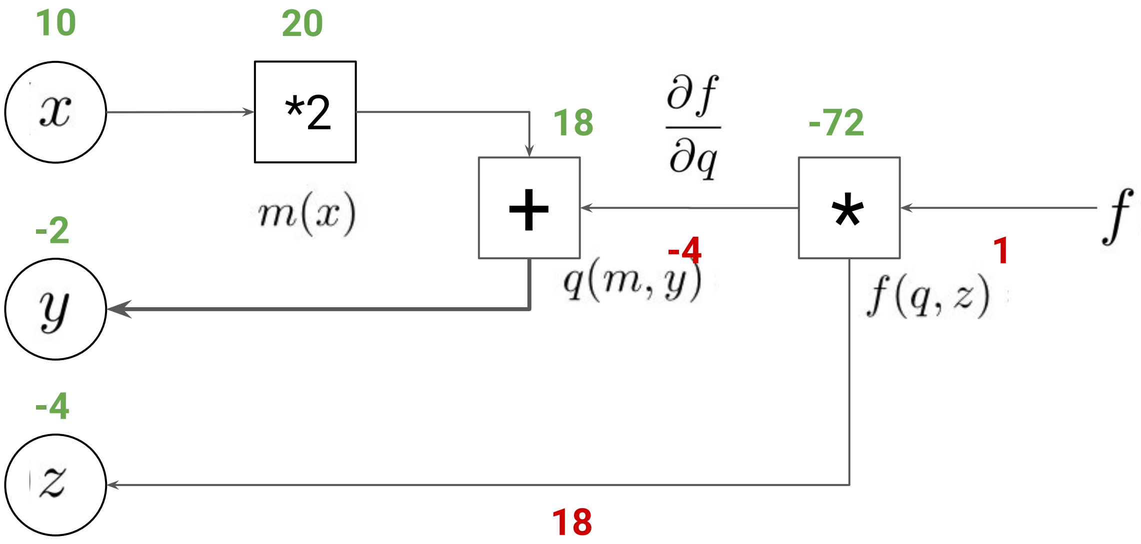
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial y} = \boxed{\frac{\partial f}{\partial q}}\frac{\partial q}{\partial y}$$

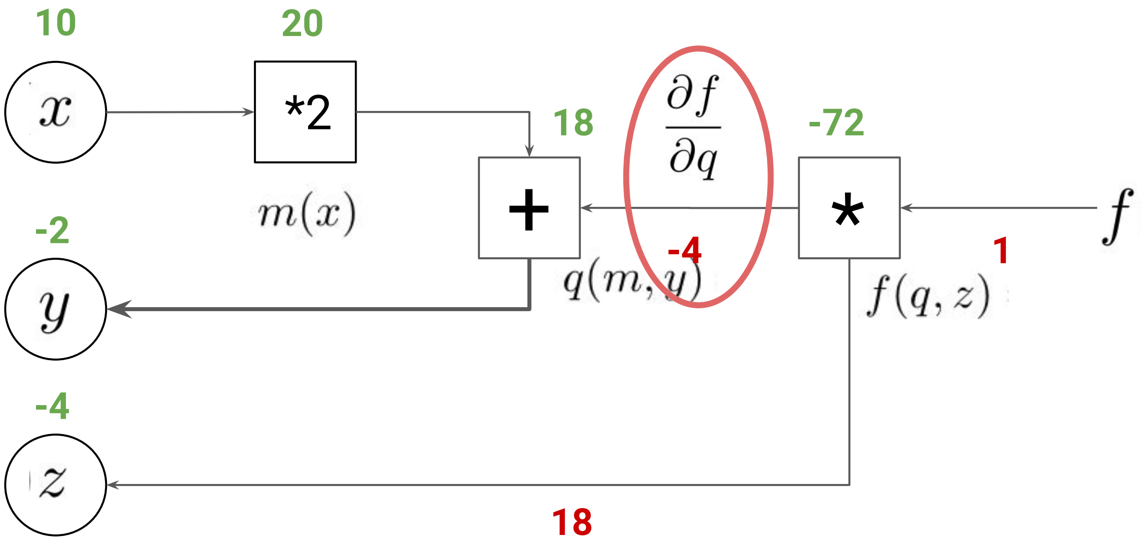
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial y} = \boxed{\frac{\partial f}{\partial q}} \frac{\partial q}{\partial y}$$

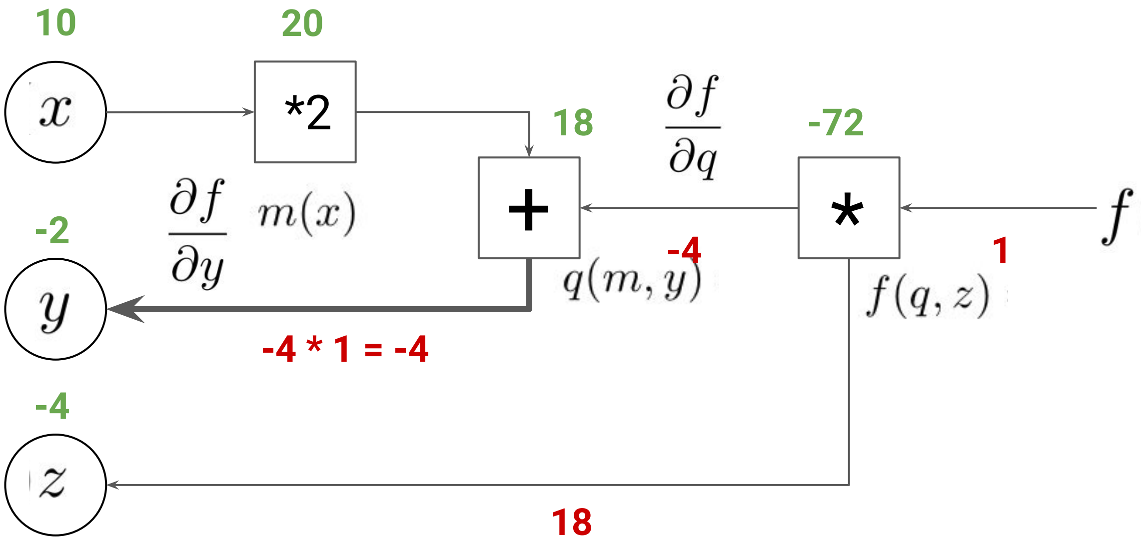
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$\frac{\partial f}{\partial y} = \boxed{\frac{\partial f}{\partial q}}\frac{\partial q}{\partial y}$$

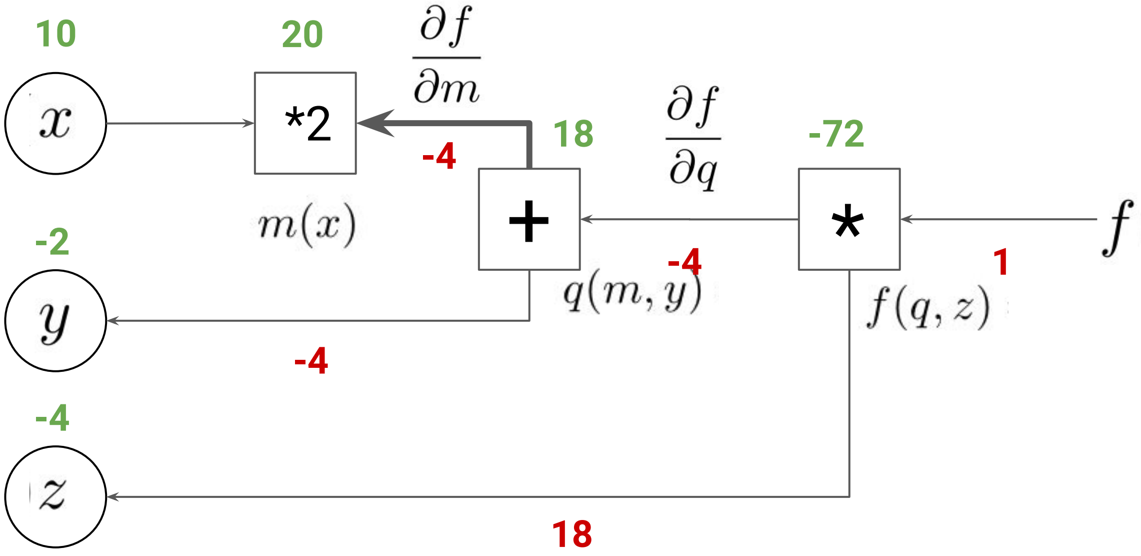
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

$$q(m, y) = m + y$$

$$f(q, z) = qz$$



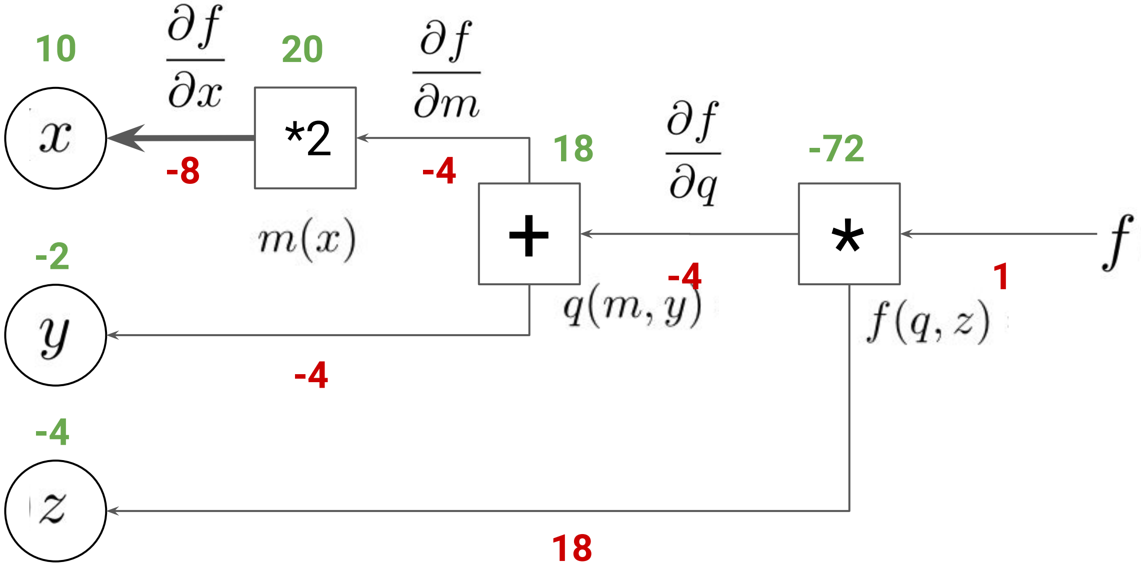
$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

$$m(x) = 2x$$

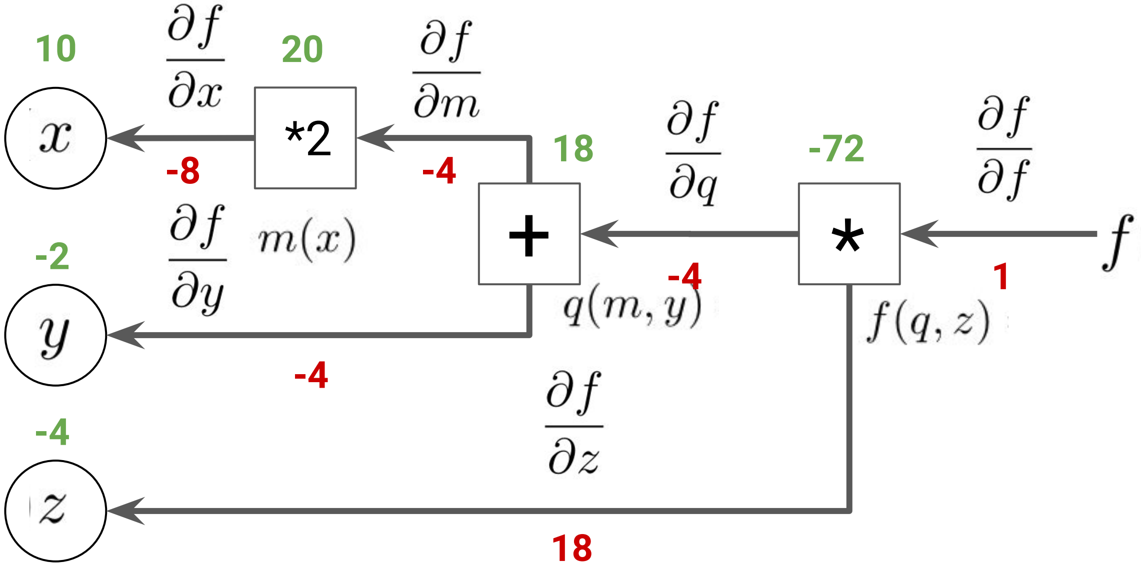
$$q(m, y) = m + y$$

$$f(q, z) = qz$$



$$f(x, y, z) = (2x + y)z$$

$$x = 10, y = -2, z = -4$$

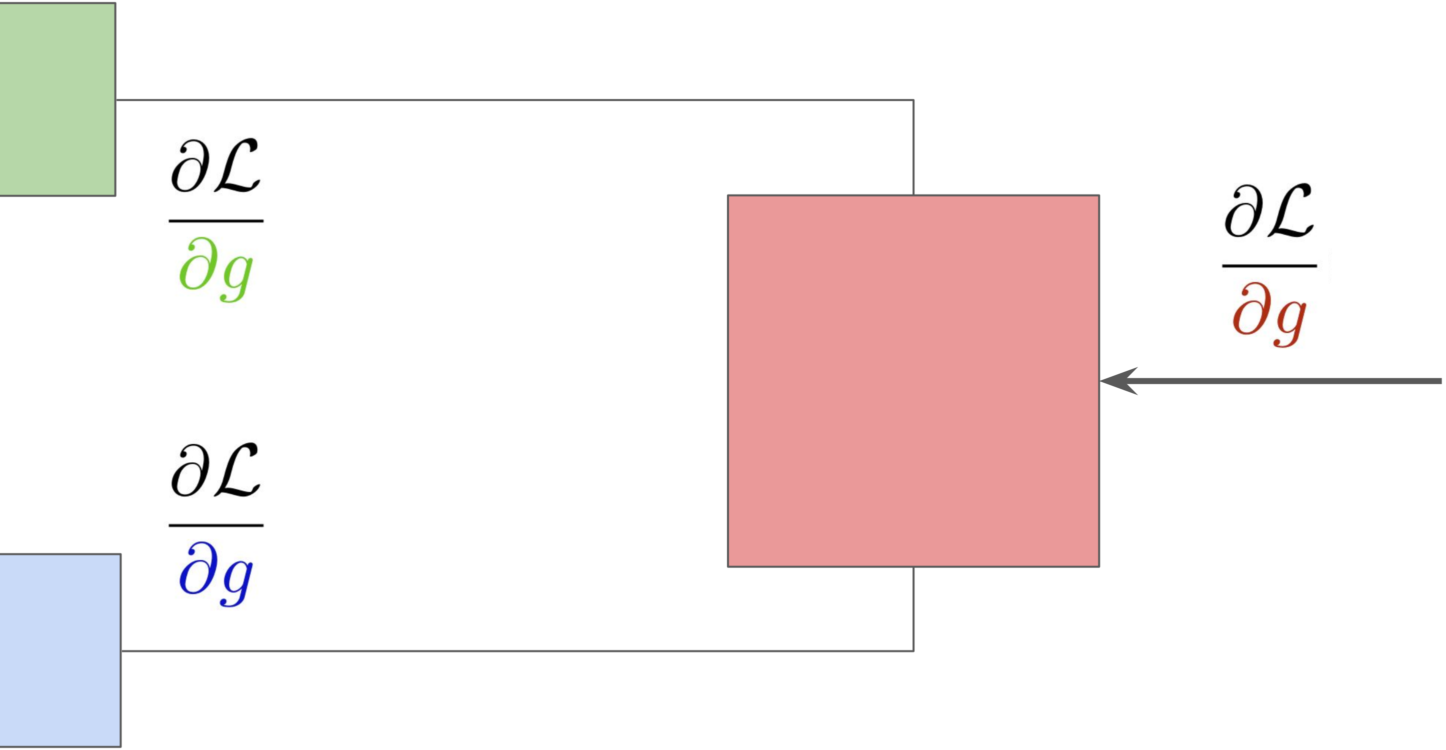


$$\frac{\partial f}{\partial x} = 2z = -8$$

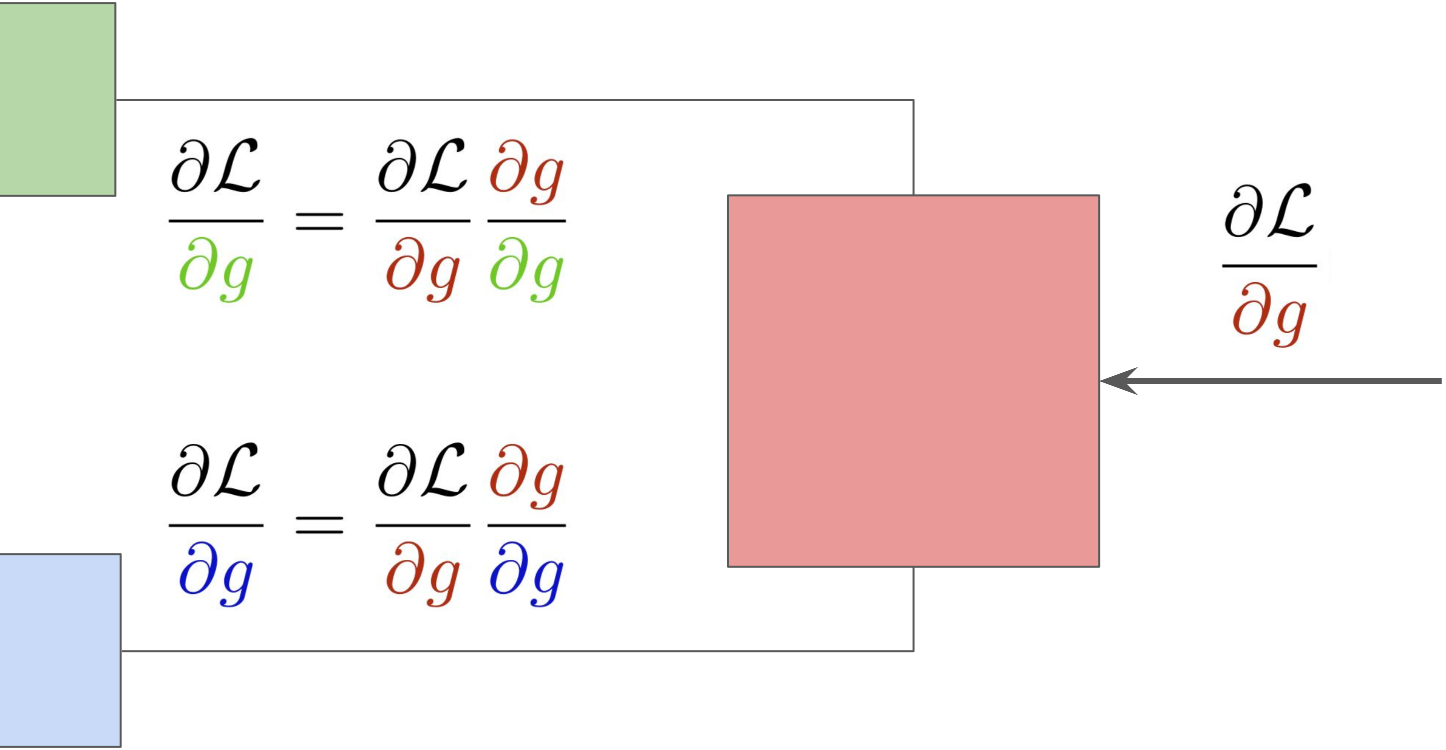
$$\frac{\partial f}{\partial y} = z = -4$$

$$\frac{\partial f}{\partial z} = 2x + y = 18$$

Что должен уметь гейт



Что должен уметь гейт



$$\frac{\partial \mathcal{L}}{\partial g} = \frac{\partial \mathcal{L}}{\partial g} \frac{\partial g}{\partial g}$$

$$\frac{\partial \mathcal{L}}{\partial g} = \frac{\partial \mathcal{L}}{\partial g} \frac{\partial g}{\partial g}$$

$$\frac{\partial \mathcal{L}}{\partial g}$$

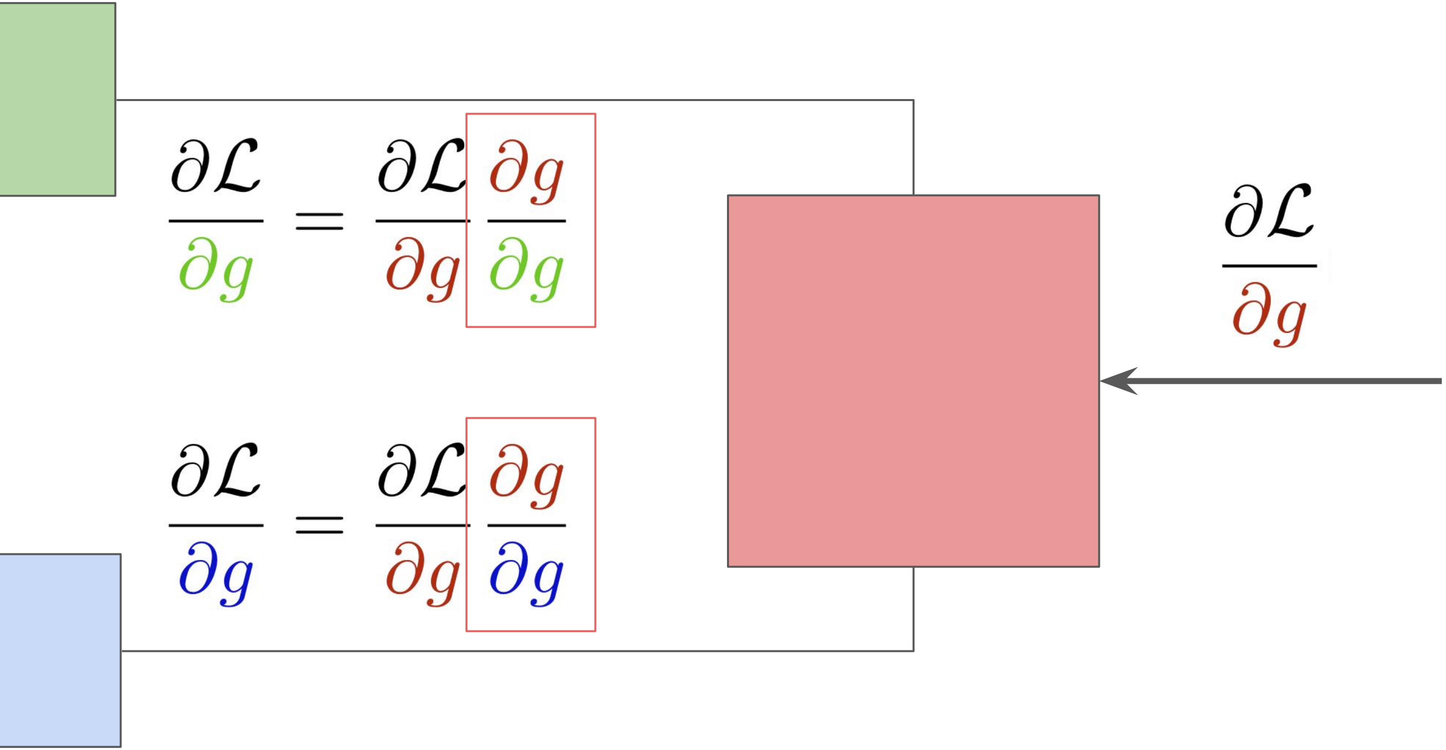
Что должен уметь гейт

$$\frac{\partial \mathcal{L}}{\partial g} = \boxed{\frac{\partial \mathcal{L}}{\partial g}}\frac{\partial g}{\partial g}$$

$$\frac{\partial \mathcal{L}}{\partial g} = \boxed{\frac{\partial \mathcal{L}}{\partial g}}\frac{\partial g}{\partial g}$$

$$\boxed{\frac{\partial \mathcal{L}}{\partial g}}$$

Что должен уметь гейт



$$\frac{\partial \mathcal{L}}{\partial g} = \frac{\partial \mathcal{L}}{\partial g}\boxed{\frac{\partial g}{\partial g}}$$

$$\frac{\partial \mathcal{L}}{\partial g} = \frac{\partial \mathcal{L}}{\partial g}\boxed{\frac{\partial g}{\partial g}}$$

$$\frac{\partial \mathcal{L}}{\partial g}$$

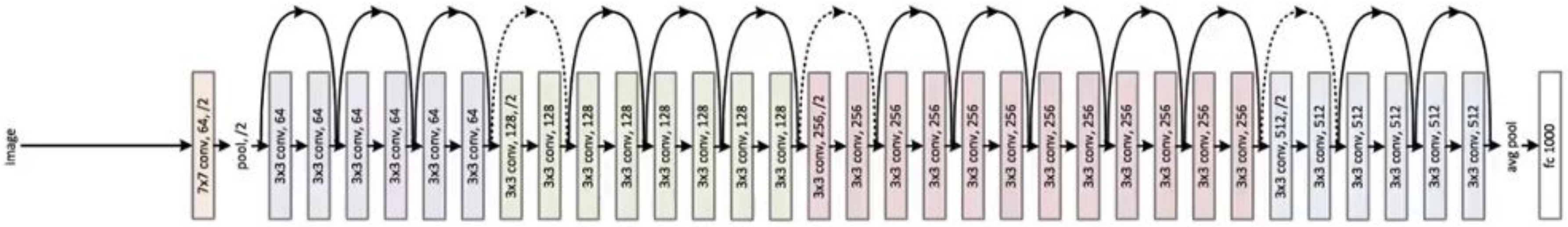
```python
class MultGate(Layer):
    def __init__(self):
        self.x = None
        self.y = None

    def forward(self, inputs):
        self.x, self.y = inputs
        z = self.x * self.y
        return z

    def backward(self, dL_dz):
        dz_dx = self.y
        dz_dy = self.x

        dL_dx = dL_dz * dz_dx
        dL_dy = dL_dz * dz_dy

        return dL_dx, dL_dy
```

```python
class MultGate(Layer):
    def __init__(self):
        self.x = None
        self.y = None

    def forward(self, inputs):
        self.x, self.y = inputs
        z = self.x * self.y
        return z

    def backward(self, dL_dz):
        dz_dx = self.y
        dz_dy = self.x

        dL_dx = dL_dz * dz_dx
        dL_dy = dL_dz * dz_dy

        return dL_dx, dL_dy
```

```python
class MaxGate(Layer):
    def __init__(self):
        self.x = None
        self.y = None

    def forward(self, inputs):
        self.x, self.y = inputs

        return max(self.x, self.y)

    def backward(self, dL_dz):

        if self.x > self.y:
            dz_dx = 1
            dz_dy = 0
        else:
            dz_dx = 0
            dz_dy = 1

        dL_dx = dL_dz * dz_dx
        dL_dy = dL_dz * dz_dy

        return dL_dx, dL_dy
```

Skillbox

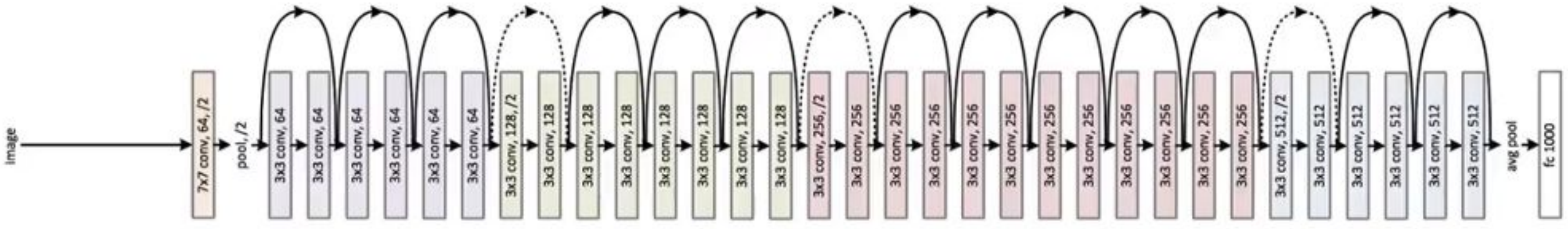
Нейронная сеть как вычислительный граф



Forward каждого слоя по-очереди

Skillbox

Нейронная сеть как вычислительный граф



Backward каждого слоя в обратном порядке

Итог

- Функции можно представлять в виде вычислительного графа
- Forward pass -- считает значение функции
- Backward pass -- производные
- Back propagation позволяет быстро и эффективно считать производные сложных функций
- Эффективность достигается за счет того, что мы используем часть информации полученной в forward pass
- Для реализации гейта (слоя) нужно определить два метода: forward и backward

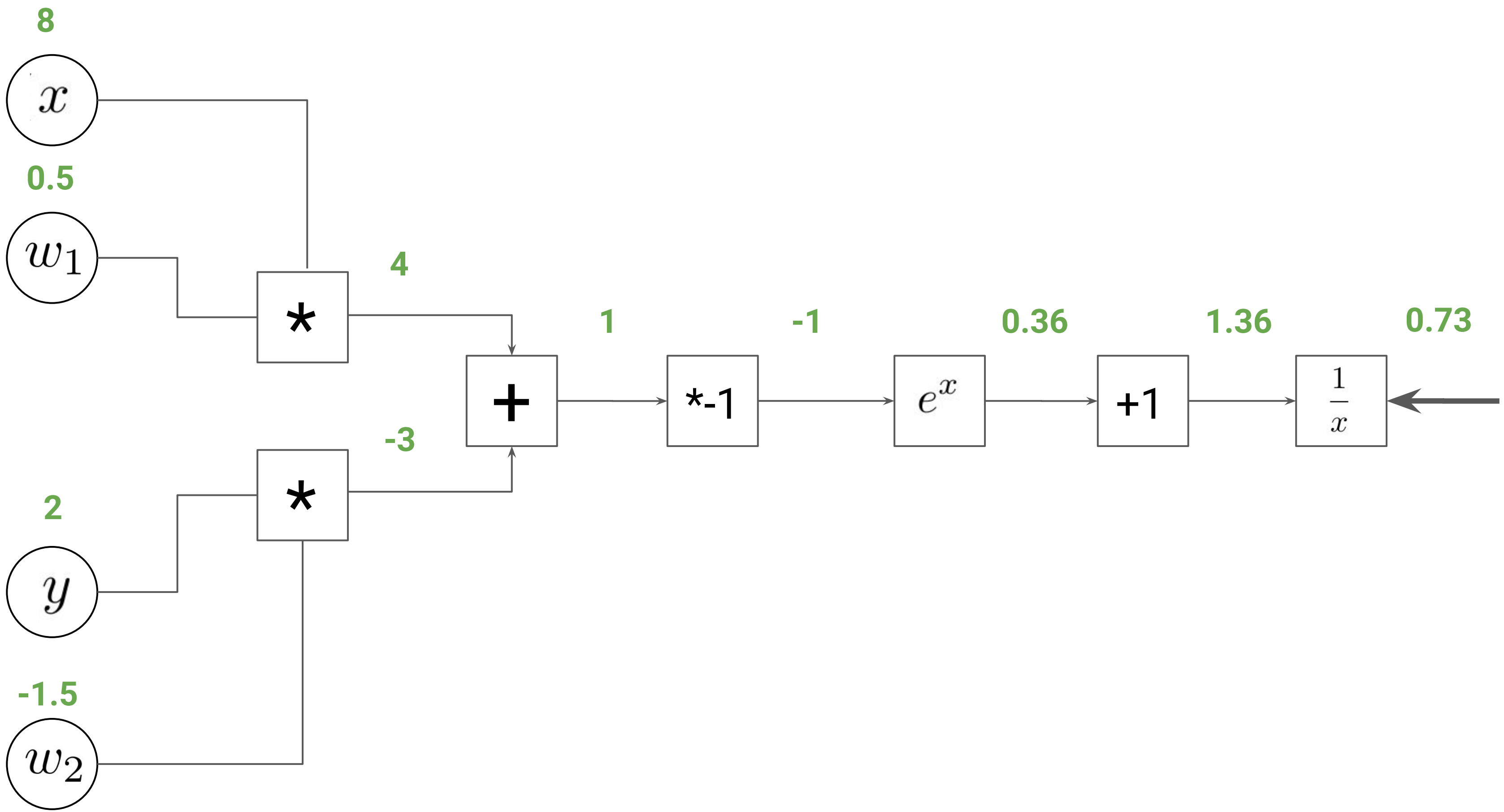
On backpropagation:

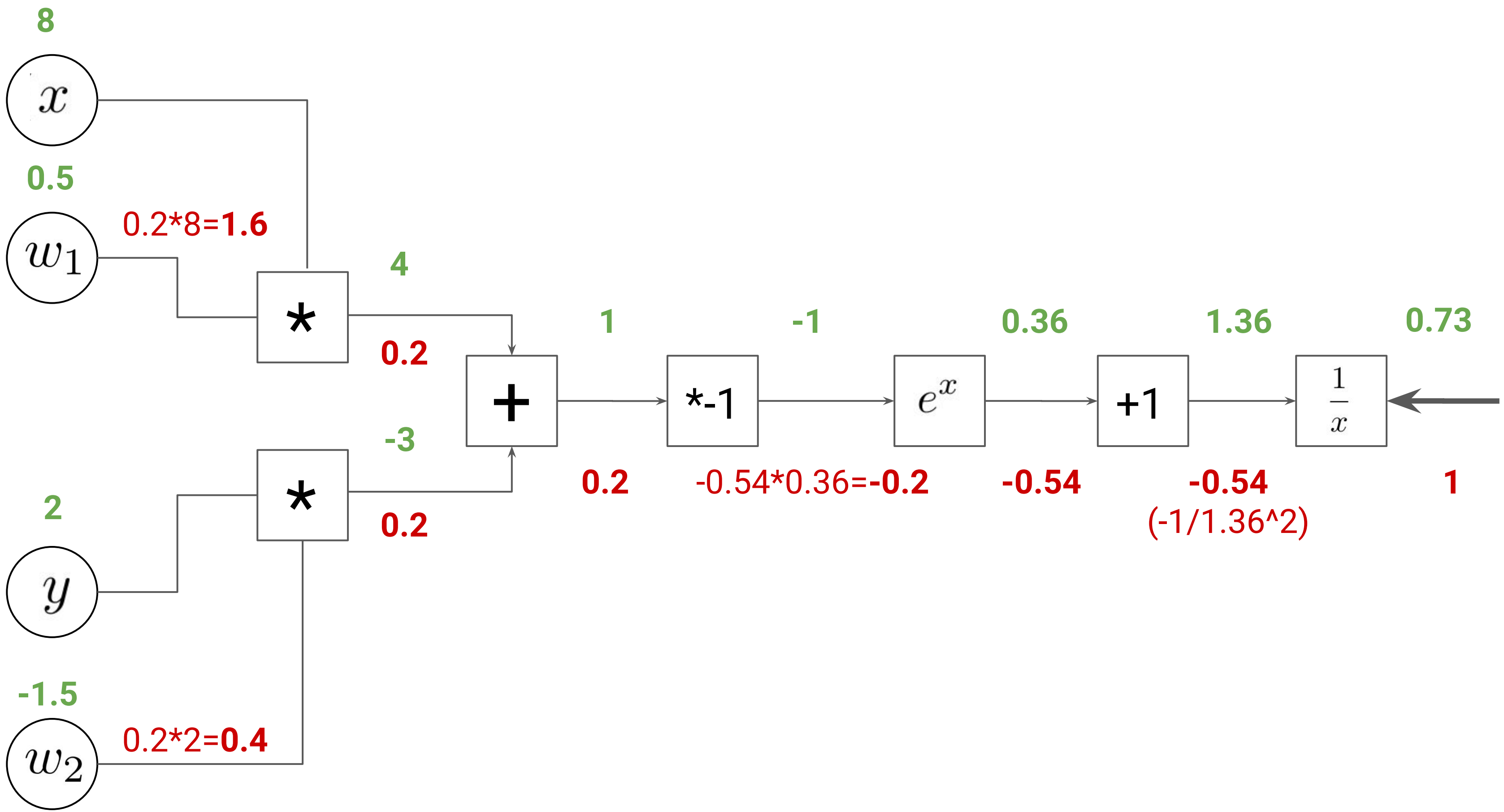
"My view is throw it all away and start again."

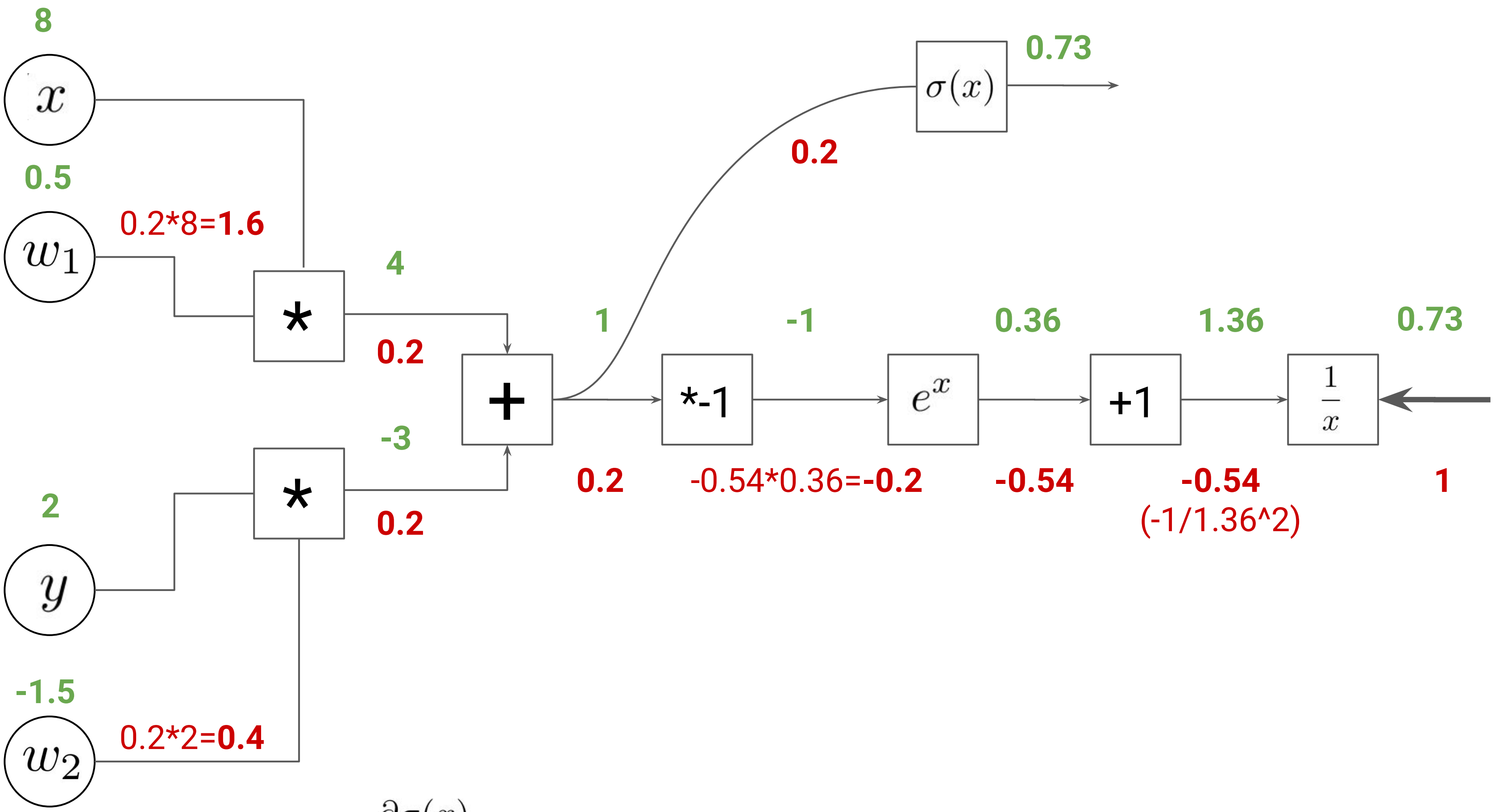
"The future depends on some graduate student who is deeply suspicious of everything I have said."

Geoffrey Hinton

"Godfather of Deep Learning"


8
x
0.5
w1
4
*
1
-1
0.36
1.36
0.73
+
*-1
e^x
+1
1/x
-3
*
2
y
-1.5
w2


8
x
0.5
w1
0.2*8=1.6
4
*
0.2
+
-3
*
0.2
2
y
-1.5
w2
0.2*2=0.4
1
0.2
*-1
-1
-0.54*0.36=-0.2
e^x
0.36
-0.54
+1
1.36
-0.54
(-1/1.36^2)
1/x
0.73
1



$$\frac{\partial \sigma(x)}{\partial x} = \sigma(x)\left(1 - \sigma(x)\right)$$

Skillbox

Обучение нейронных сетей

Стохастический градиентный спуск

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\boldsymbol{x}_i, \boldsymbol{\theta}))$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:
$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$
   a. **После этого шага параметры изменились!**
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\boldsymbol{x}_i, \boldsymbol{\theta}))$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:
$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$
   a. **После этого шага параметры изменились!**
5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

Градиентный спуск

1. Инициализировать параметры случайным образом (выбрать θ_0)
2. Вычислить значение функционала $\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\boldsymbol{x}_i, \boldsymbol{\theta}))$
3. Вычислить $\frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$ при текущем значении параметра
4. Сделать шаг оптимизации:

$$\theta_i = \theta_{i-1} - \lambda \frac{\partial \mathcal{L}}{\partial \theta}(\theta_{i-1})$$

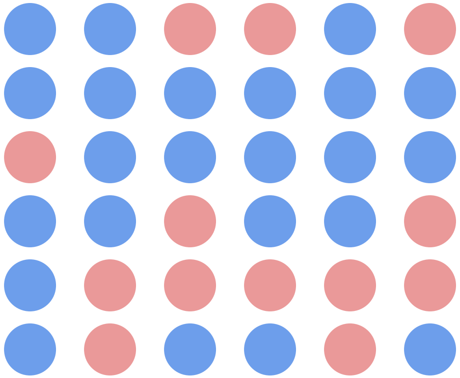
   a. **После этого шага параметры изменились!**

5. Повторять 2-4 пока $\mathcal{L}$ не перестанет меняться

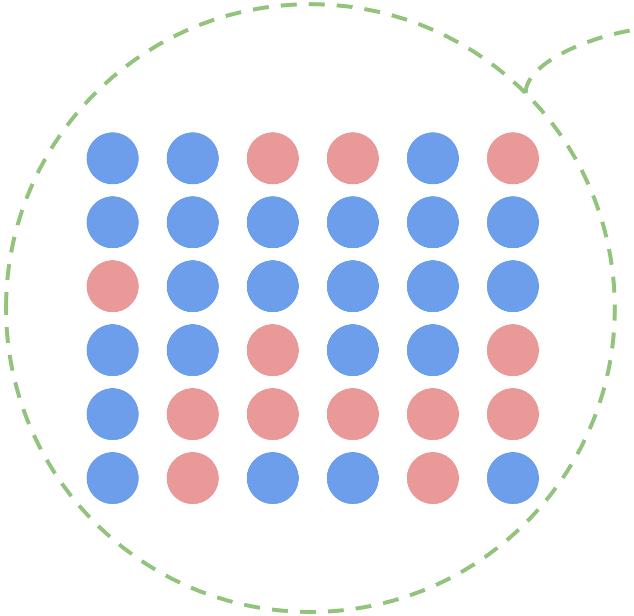
Нужно применить f ко всей выборке!
Это может быть очень долго...

Skillbox

Градиентный спуск



Градиентный спуск



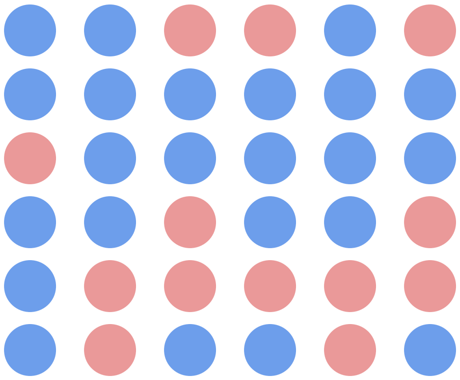
$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

Skillbox

Стохастический градиентный спуск

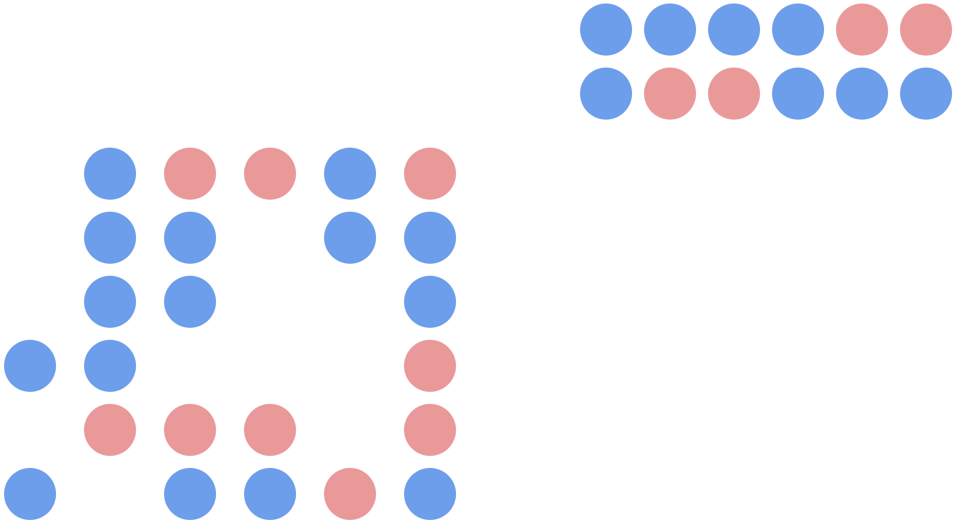
Stochastic Gradient Descent (SGD)



Skillbox

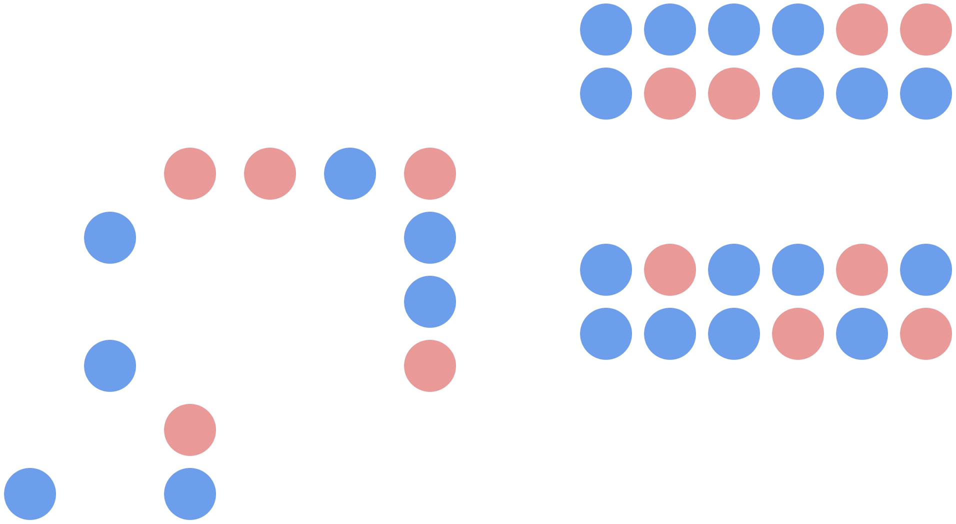
Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)



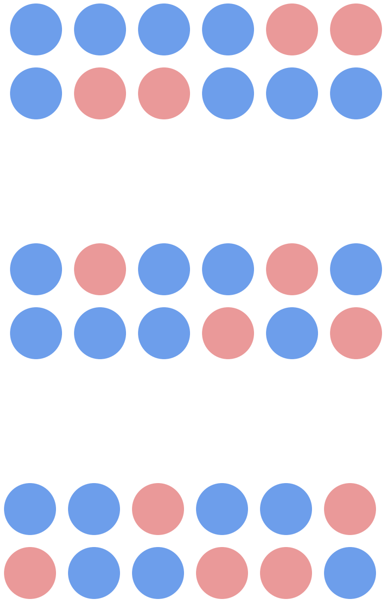
Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)



Стохастический градиентный спуск

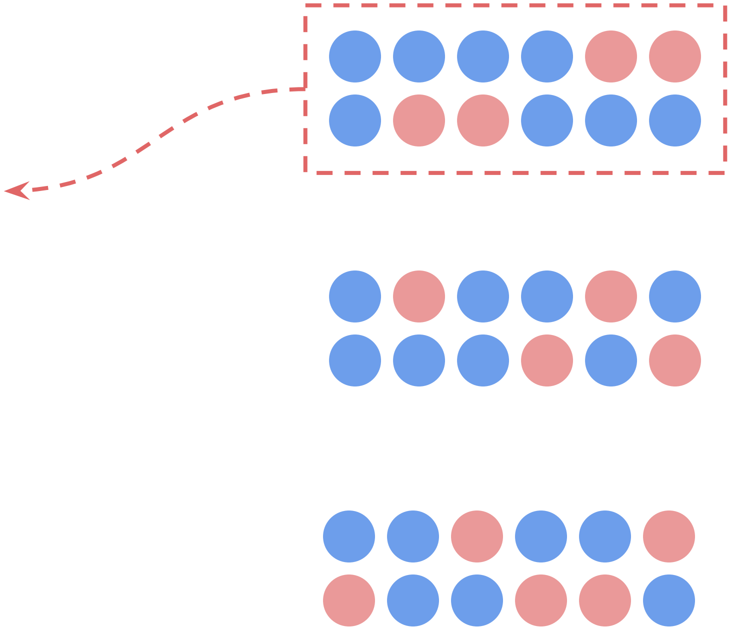
Stochastic Gradient Descent (SGD)



Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)

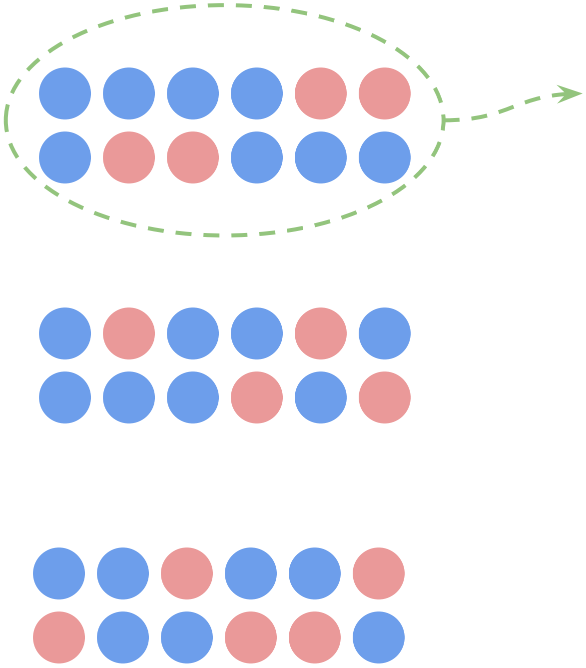
Batch, mini-batch
Batch size = 12



Skillbox

Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)

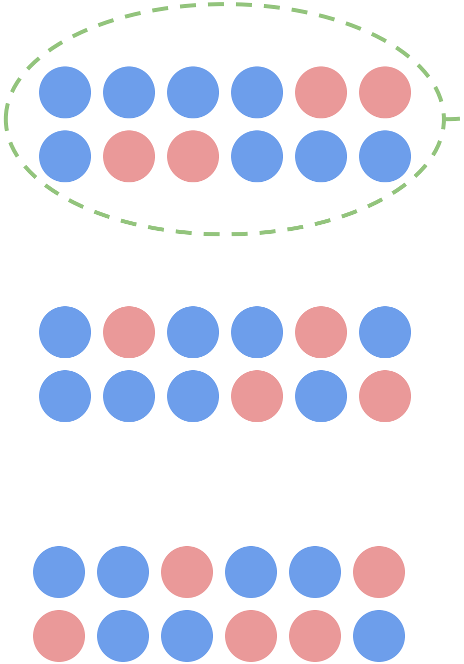


$$\mathcal{L} = \frac{1}{n_1} \sum_{i \in [\text{batch 1 ids}]} L(y_i, \hat{y}_i)$$

Skillbox

Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)

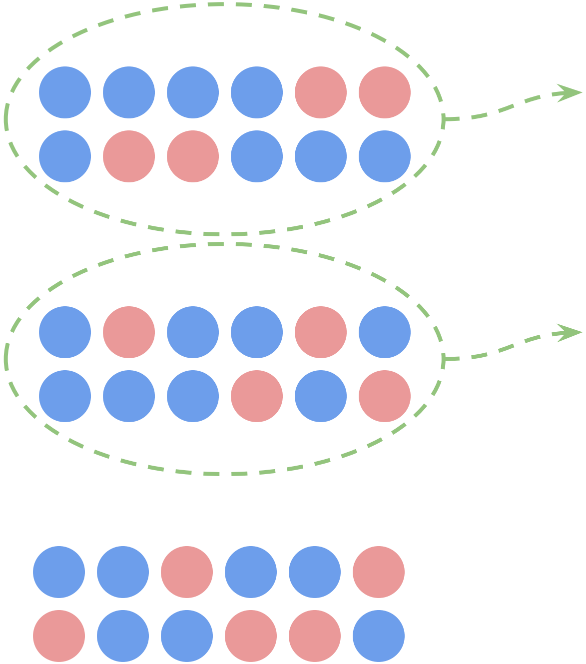


$$\mathcal{L} = \frac{1}{n_1} \sum_{i \in [\text{batch 1 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)



$$\mathcal{L} = \frac{1}{n_1} \sum_{i \in [\text{batch 1 ids}]} L(y_i, \hat{y}_i)$$

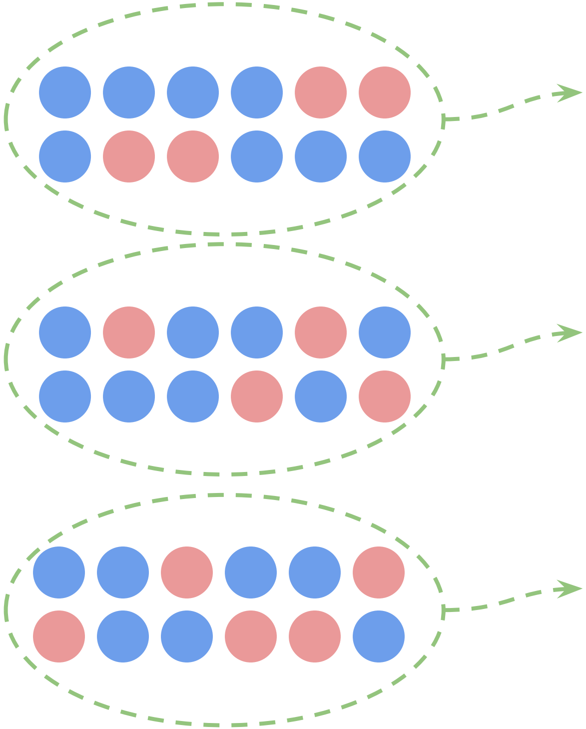
\+ Шаг оптимизации

$$\mathcal{L} = \frac{1}{n_2} \sum_{i \in [\text{batch 2 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)



$$\mathcal{L} = \frac{1}{n_1} \sum_{i \in [\text{batch 1 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

$$\mathcal{L} = \frac{1}{n_2} \sum_{i \in [\text{batch 2 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

$$\mathcal{L} = \frac{1}{n_3} \sum_{i \in [\text{batch 3 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

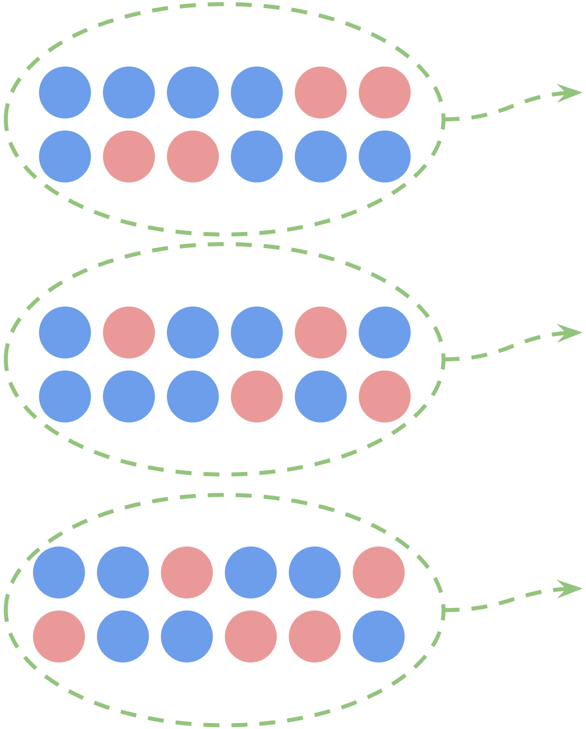
Стохастический градиентный спуск

Stochastic Gradient Descent (SGD)

Итерация №1

Итерация №2

Итерация №3



$$\mathcal{L} = \frac{1}{n_1} \sum_{i \in [\text{batch 1 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

$$\mathcal{L} = \frac{1}{n_2} \sum_{i \in [\text{batch 2 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

$$\mathcal{L} = \frac{1}{n_3} \sum_{i \in [\text{batch 3 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

Стохастический градиентный спуск

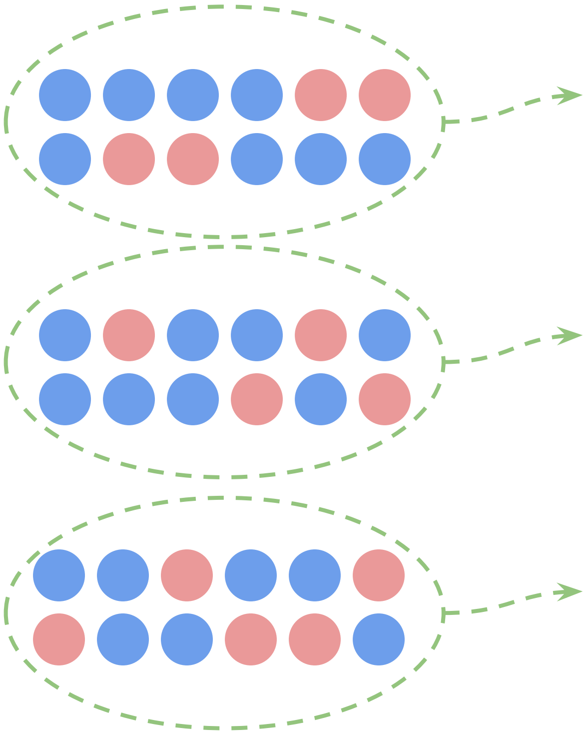
Stochastic Gradient Descent (SGD)

Эпоха №1

Итерация №1

Итерация №2

Итерация №3



$$\mathcal{L} = \frac{1}{n_1} \sum_{i \in [\text{batch 1 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

$$\mathcal{L} = \frac{1}{n_2} \sum_{i \in [\text{batch 2 ids}]} L(y_i, \hat{y}_i)$$

\+ Шаг оптимизации

$$\mathcal{L} = \frac{1}{n_3} \sum_{i \in [\text{batch 3 ids}]} L(y_i, \hat{y}_i)$$

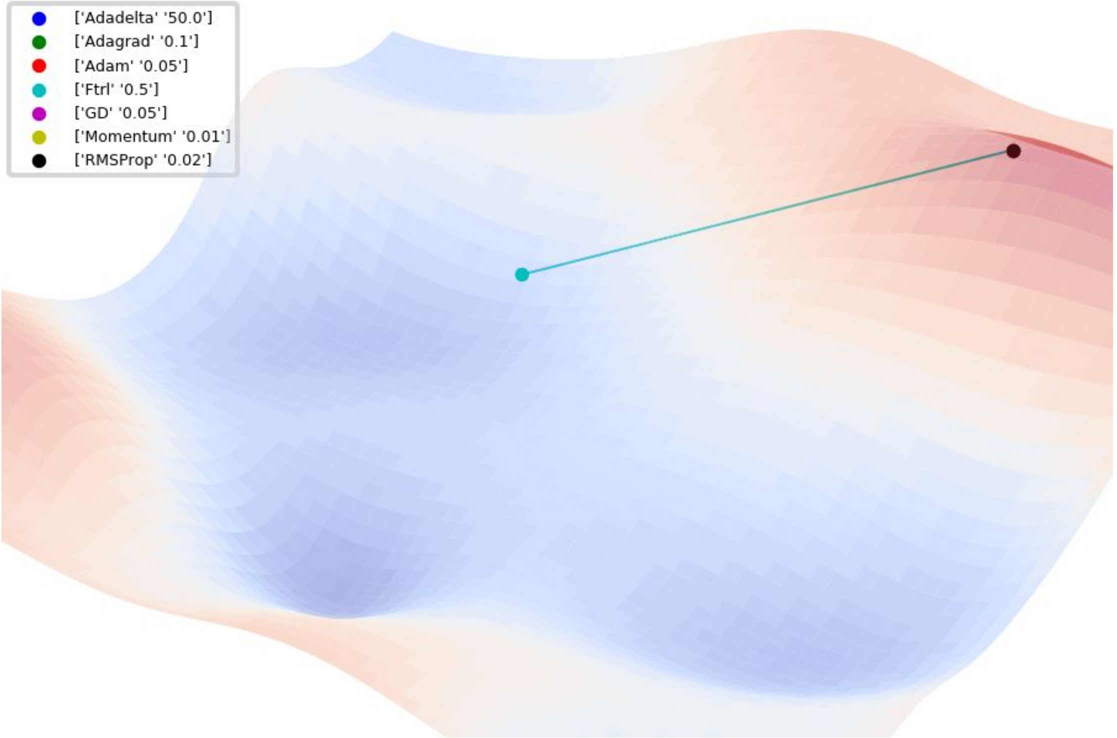
\+ Шаг оптимизации

Достоинства и недостатки SGD

- Достоинства:
  - Позволяет делать шаги оптимизации чаще
  - Легко реализуется
- Недостатки:
  - Застревание в локальных минимумах
  - Медленная сходимость

На практике чаще используются модификации SGD (например, Adam)

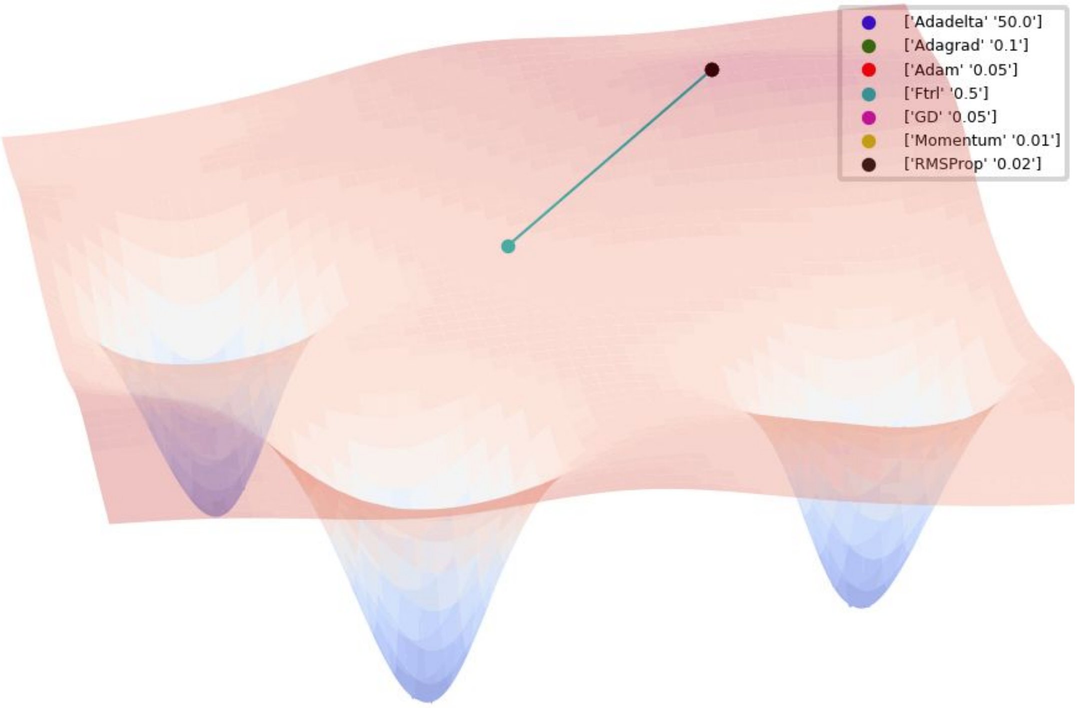
Модификации градиентного спуска



Источник: https://github.com/Jaewan-Yun/optimizer-visualization

Skillbox

Модификации градиентного спуска



Источник: https://github.com/Jaewan-Yun/optimizer-visualization

ADAM (упрощенно)

GD update: $$\boxed{\theta_t \leftarrow \theta_{t-1} - \alpha \nabla_\theta f_t(\theta_{t-1})}$$

ADAM (упрощенно)

GD update: $$\theta_t \leftarrow \theta_{t-1} - \alpha \nabla_\theta f_t(\theta_{t-1})$$

ADAM update: $$\boxed{\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)}$$

Skillbox

ADAM (упрощенно)

GD update: $$\theta_t \leftarrow \theta_{t-1} - \alpha \nabla_\theta f_t(\theta_{t-1})$$

ADAM update: $$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \boxed{\widehat{m}_t} / (\sqrt{\boxed{\widehat{v}_t}} + \epsilon)$$

Экспоненциальное скользящее среднее градиента
(**“идем примерно туда куда шли”**)

Экспоненциальное скользящее среднее квадрата градиента
(**“если сильно болтает -- делаем шаги меньше”**)

Итог

- Классический градиентный спуск требует расчета ошибки по всей выборке данных для того чтобы сделать шаг оптимизации
- Стохастический градиентный спуск работает с “кусочками” данных, делая обновления чаще
- Тем не менее он склонен к застреванию в локальных минимумах (как и классический вариант)
- Есть множество модификаций алгоритма, с одним из самых популярных из которых мы познакомились на лекции

Skillbox

Обучение нейронных сетей

Функции активации

Нейрон

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

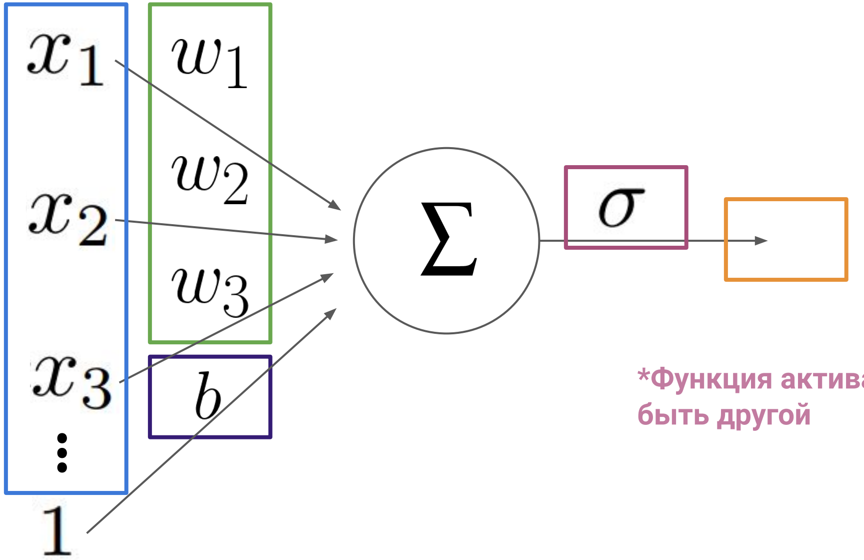
Вход нейрона (признаки, inputs)

Коэффициенты (веса, weights)

Смещение (bias)

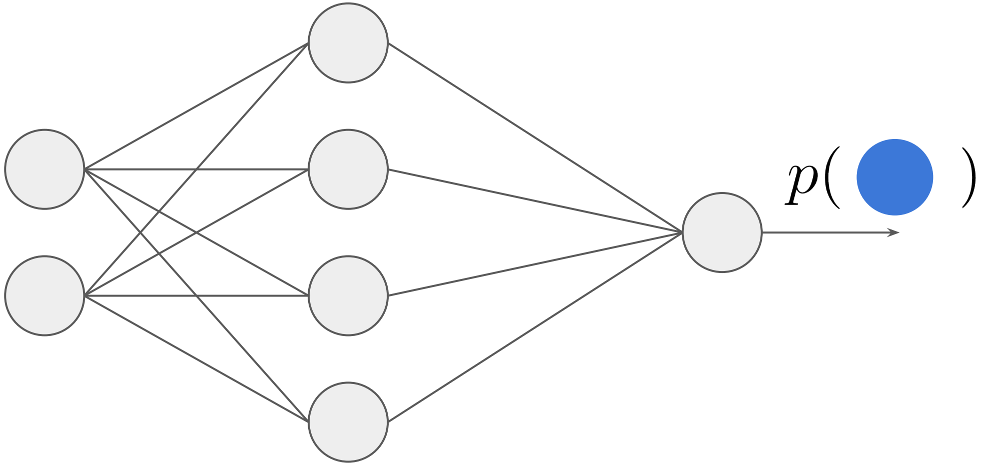
Функция активации

Выход нейрона



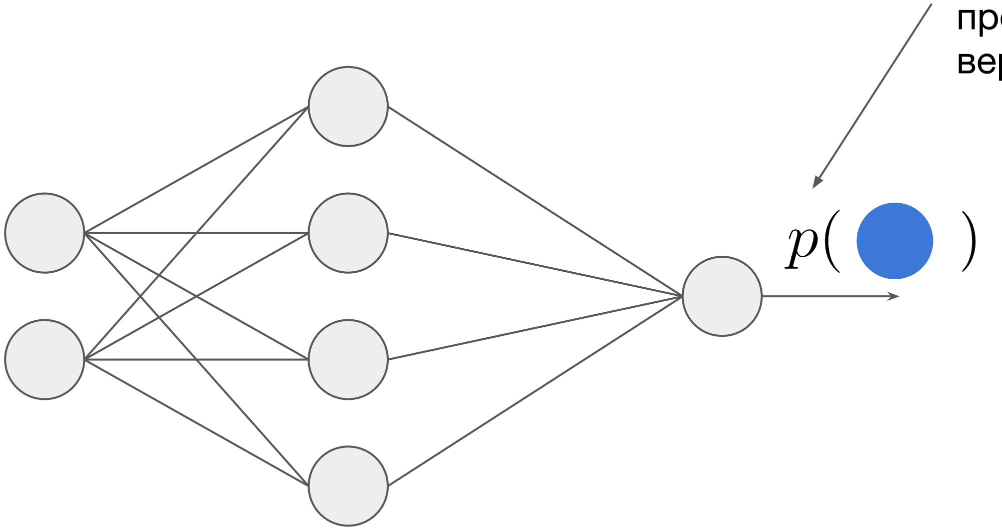
*Функция активации может быть другой

Функции активации



Функции активации

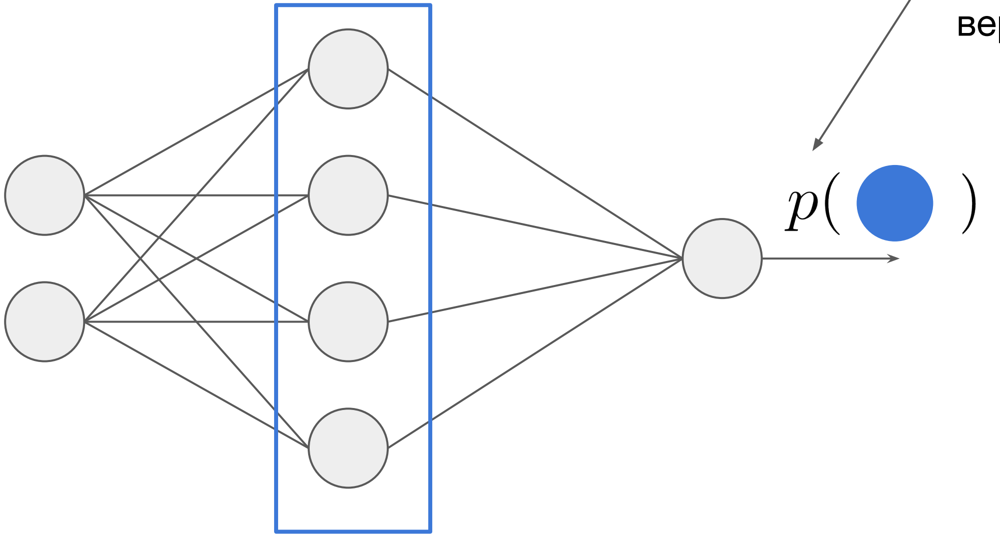
Сигмоида, потому что предсказываем вероятность



Функции активации

Обязательна ли сигмоида в промежуточных слоях?

Сигмоида, потому что предсказываем вероятность

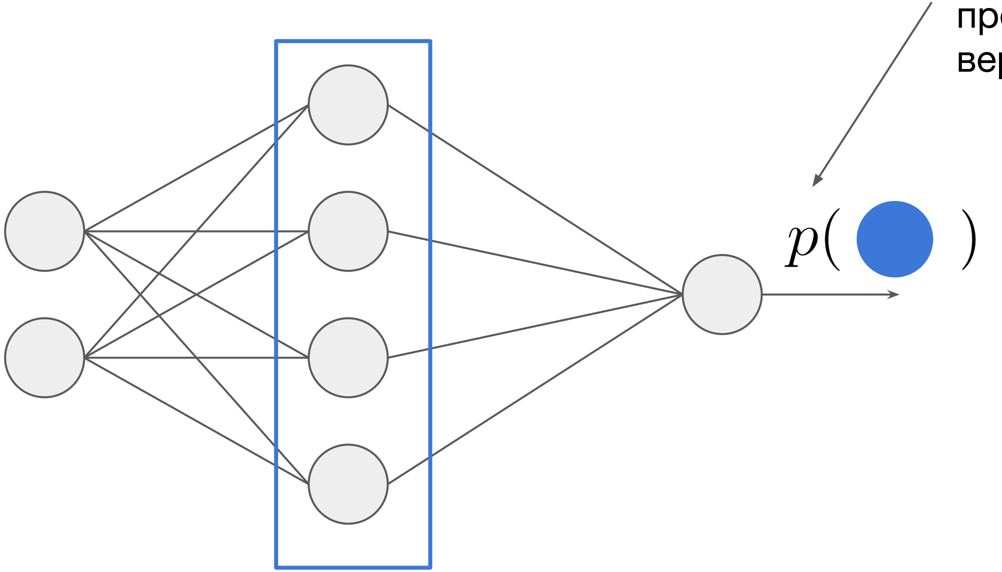


Функции активации

Обязательна ли сигмоида в промежуточных слоях?

- **Нет**

Сигмоида, потому что предсказываем вероятность



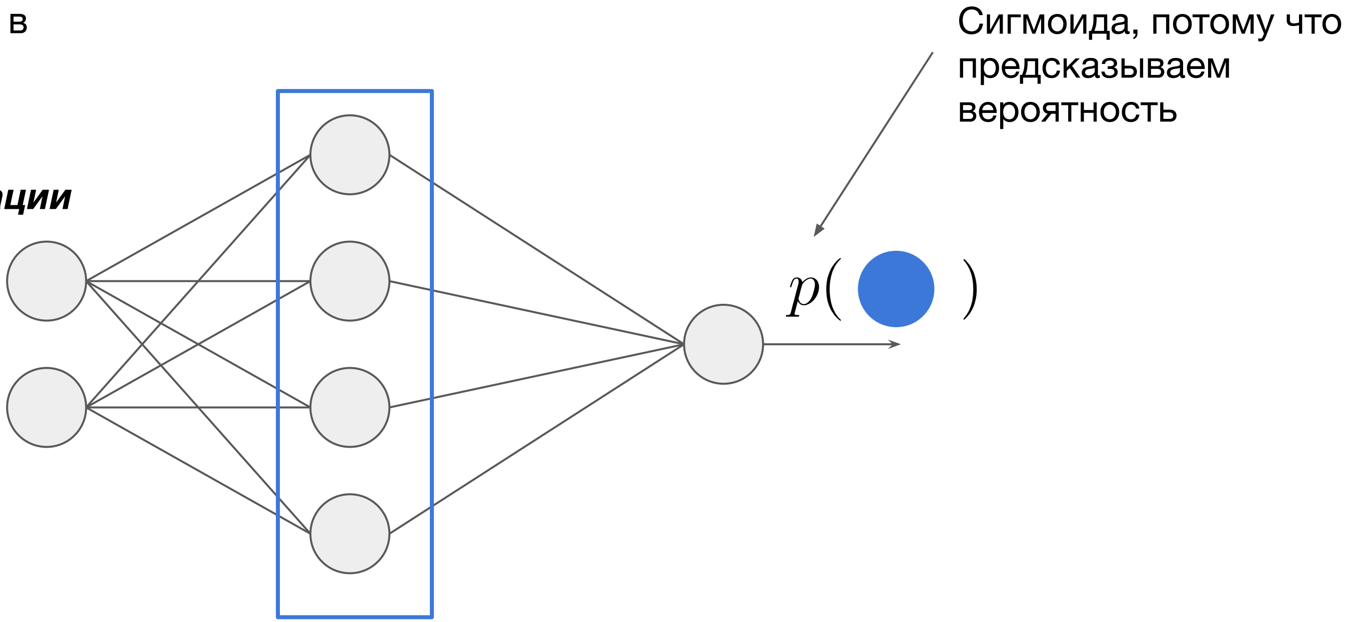
Skillbox

Функции активации

Обязательна ли сигмоида в промежуточных слоях?

- **Нет**

Можно ли убрать активации вообще?



Функции активации

$$xw_1 + b_1$$

Линейная функция (один нейрон без активации, один признак на входе)

Функции активации

$$\sigma((xw_1 + b_1)w_2 + b_2)$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность

Функции активации

$$\sigma((xw_1 + b_1)w_2 + b_2)$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность
Сделали мы функцию сложнее?

Функции активации

$$\sigma((xw_1 + b_1)w_2 + b_2)$$

$$xw_1w_2 + b_1w_2 + b_2$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность
Сделали мы функцию сложнее?

Функции активации

$$\sigma((xw_1 + b_1)w_2 + b_2)$$

$$w = w_1 w_2, b = b_1 w_2 + b_2 \qquad x\boxed{w_1 w_2} + \boxed{b_1 w_2 + b_2}$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность
Сделали мы функцию сложнее?

Функции активации

$$\sigma((xw_1 + b_1)w_2 + b_2)$$

$$w = w_1w_2, b = b_1w_2 + b_2 \qquad x\boxed{w_1w_2} + \boxed{b_1w_2 + b_2} \longrightarrow xw + b$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность
Сделали мы функцию сложнее?

Функции активации

$$\sigma((xw_1 + b_1)w_2 + b_2)$$

$$w = w_1 w_2, b = b_1 w_2 + b_2 \qquad x\boxed{w_1 w_2} + \boxed{b_1 w_2 + b_2} \longrightarrow xw + b$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность
Сделали мы функцию сложнее?
Нет! Линейная функция от линейной функции -- линейная функция!

Функции активации

$$\sigma((xw_1 + b_1)w_2 + b_2)$$

$$w = w_1 w_2, b = b_1 w_2 + b_2 \qquad x\boxed{w_1 w_2} + \boxed{b_1 w_2 + b_2} \longrightarrow xw + b$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность
Сделали мы функцию сложнее?
Нет! Линейная функция от линейной функции -- линейная функция!

Чтобы придумывать более сложные признаки нам нужна нелинейность

Функции активации

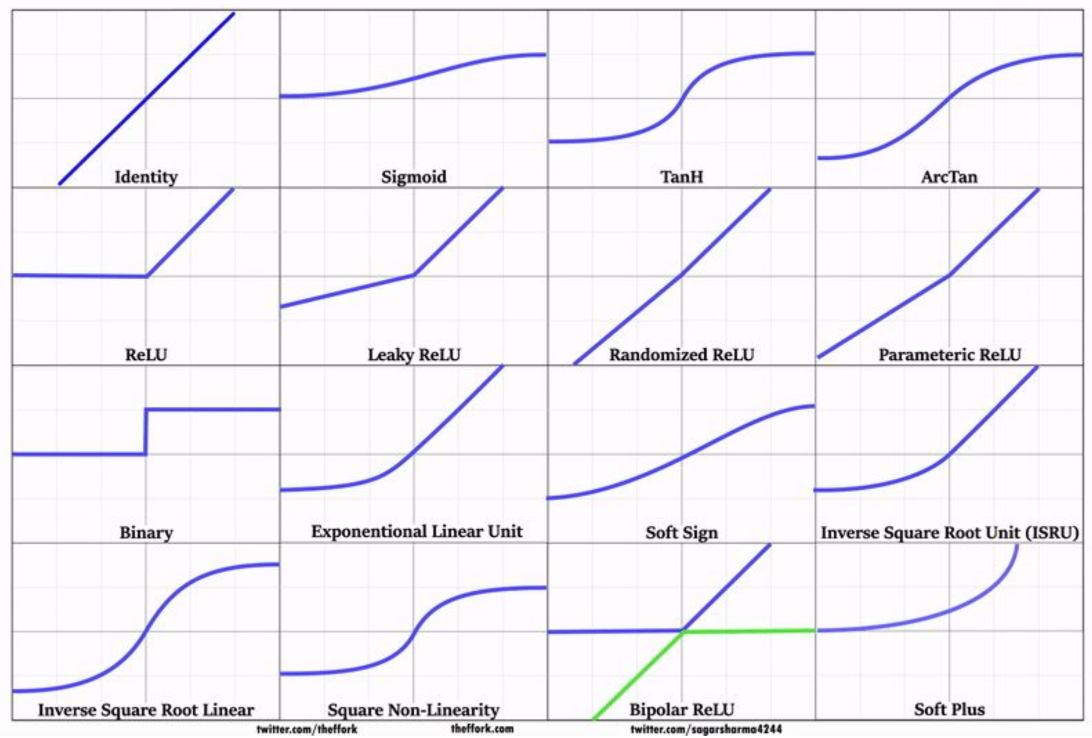
$$\sigma((xw_1 + b_1)w_2 + b_2)$$

$$w = w_1 w_2, b = b_1 w_2 + b_2 \qquad x\boxed{w_1 w_2} + \boxed{b_1 w_2 + b_2} \longrightarrow xw + b$$

Применили второй нейрон и сигмоиду, чтобы получить вероятность
Сделали мы функцию сложнее?
Нет! Линейная функция от линейной функции -- линейная функция!

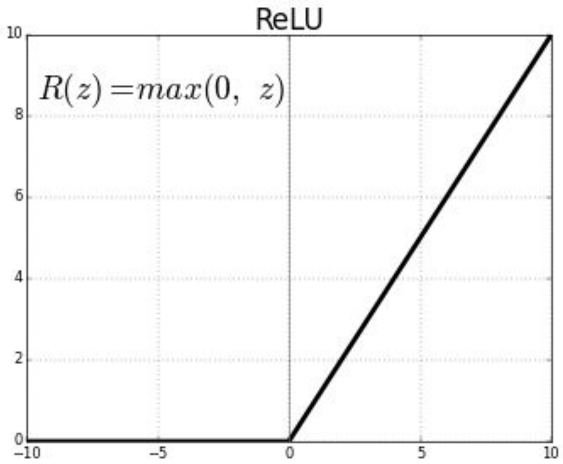
Чтобы придумывать более сложные признаки нам нужна нелинейность
Но необязательно сигмоида)

Функции активации



Источник: http://theffork.com/wp-content/uploads/2019/03/1.gif

ReLU



- Основная функции активации на сегодняшний день
- Она нелинейна
- Чем она лучше сигмоиды?
    - Короткий ответ: у нее лучше производная
    - Она быстрее считается

- Но сигмоиду все равно используем к конце, чтобы получить вероятность!

Skillbox

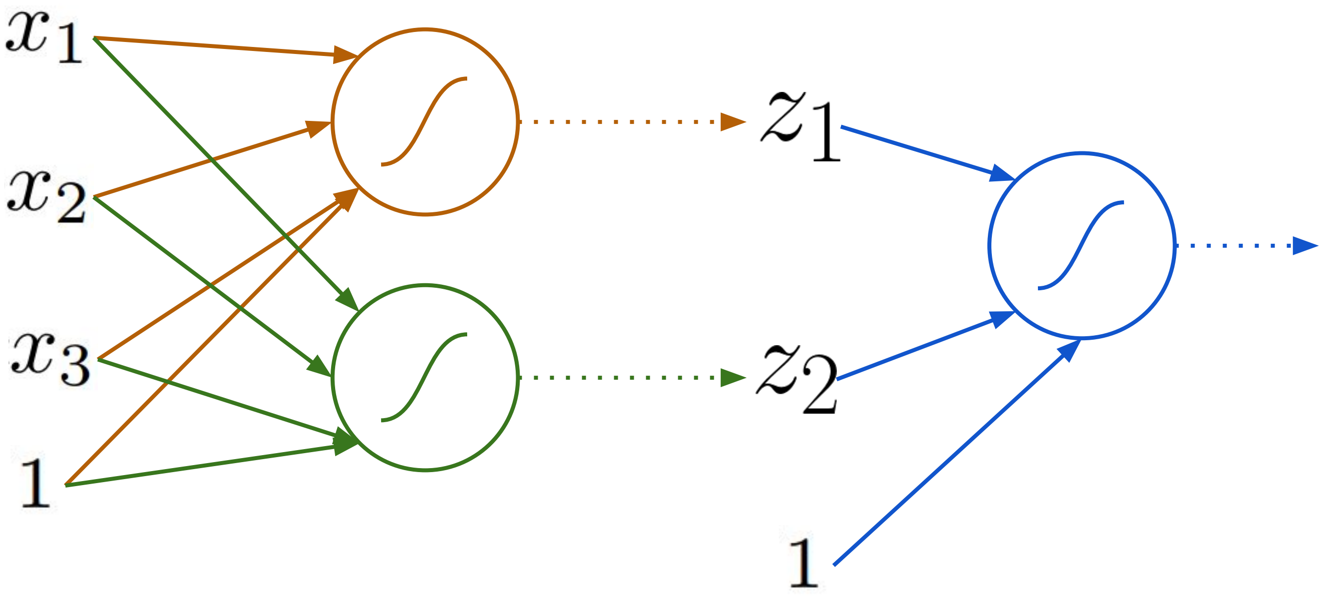
Обучение нейронных сетей

Нейронные сети: от схем к формулам

Как задать полносвязный слой в коде

- На практике прошлого блока мы реализовали нейронную сеть из трех нейронов
- Каждый мы определяли отдельно
- Как быть, если нейронов много? (а их всегда много)
- Для начала поймем как упростить запись

Нейронная сеть

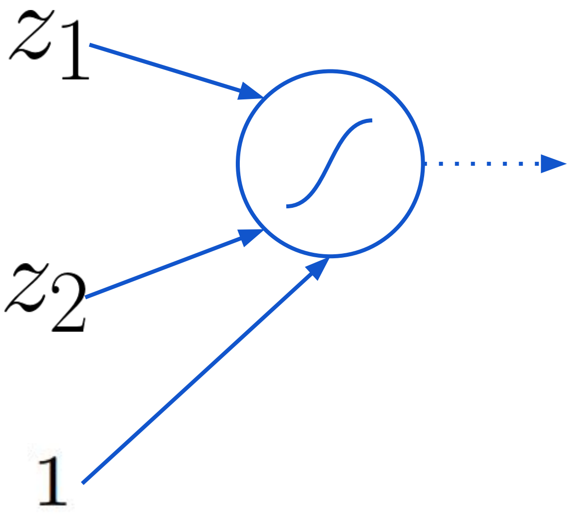


Нейронная сеть

w - вектор размера m
x - вектор размера m
b - число

$$f(\boldsymbol{w_R}^T \boldsymbol{x} + b_R) \dashrightarrow z_1$$

$$f(\boldsymbol{w_G}^T \boldsymbol{x} + b_G) \dashrightarrow z_2$$



Нейронная сеть

$$\begin{tikzcd}[column sep=small]
f(\boldsymbol{w_R}^T \boldsymbol{x} + b_R) \arrow[r, dotted] & z_1 \arrow[dr, dotted, bend left] & \\
 & & f(\boldsymbol{w_B}^T \boldsymbol{z} + b_B) \arrow[r, dotted] & {} \\
f(\boldsymbol{w_G}^T \boldsymbol{x} + b_G) \arrow[r, dotted] & z_2 \arrow[ur, dotted, bend right] &
\end{tikzcd}$$

Skillbox

$$f(\boldsymbol{w}_{\boldsymbol{R}}{}^T\boldsymbol{x} + b_R) \qquad f(\boldsymbol{w}_{\boldsymbol{G}}{}^T\boldsymbol{x} + b_G)$$

$$f(\boldsymbol{w_R}^T \boldsymbol{x} + b_R)$$

$$f(\boldsymbol{w_G}^T \boldsymbol{x} + b_G)$$

$$\begin{pmatrix} w_R^1 & w_R^2 & \dots & w_R^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + b_R$$

$$\begin{pmatrix} w_G^1 & w_G^2 & \dots & w_G^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + b_G$$

$$f(\boldsymbol{w_R}^T \boldsymbol{x} + b_R) \qquad f(\boldsymbol{w_G}^T \boldsymbol{x} + b_G)$$

$$\begin{pmatrix} w_R^1 & w_R^2 & \dots & w_R^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + b_R \qquad \begin{pmatrix} w_G^1 & w_G^2 & \dots & w_G^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + b_G$$

$$\begin{pmatrix} w_R^1 & w_R^2 & \dots & w_R^m \\ w_G^1 & w_G^2 & \dots & w_G^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + \begin{pmatrix} b_R \\ b_G \end{pmatrix}$$

$$f(\boldsymbol{w_R}^T \boldsymbol{x} + b_R) \qquad\qquad f(\boldsymbol{w_G}^T \boldsymbol{x} + b_G)$$

$$\begin{pmatrix} w_R^1 & w_R^2 & \dots & w_R^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + b_R \qquad\qquad \begin{pmatrix} w_G^1 & w_G^2 & \dots & w_G^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + b_G$$

$$\begin{pmatrix} w_R^1 & w_R^2 & \dots & w_R^m \\ w_G^1 & w_G^2 & \dots & w_G^m \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} + \begin{pmatrix} b_R \\ b_G \end{pmatrix}$$

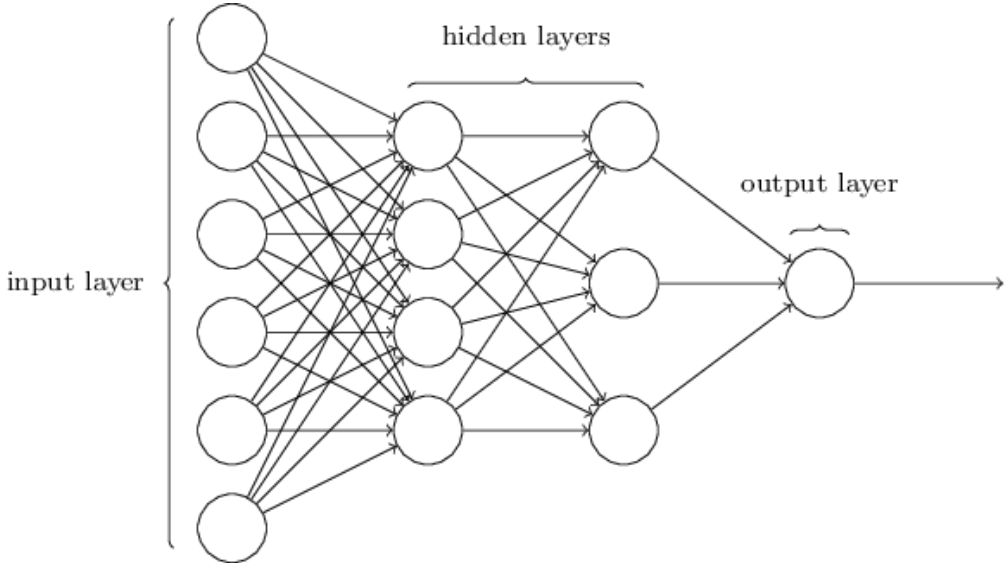
$$W\boldsymbol{x} + \boldsymbol{b}$$

$$W\boldsymbol{x} + \boldsymbol{b}$$

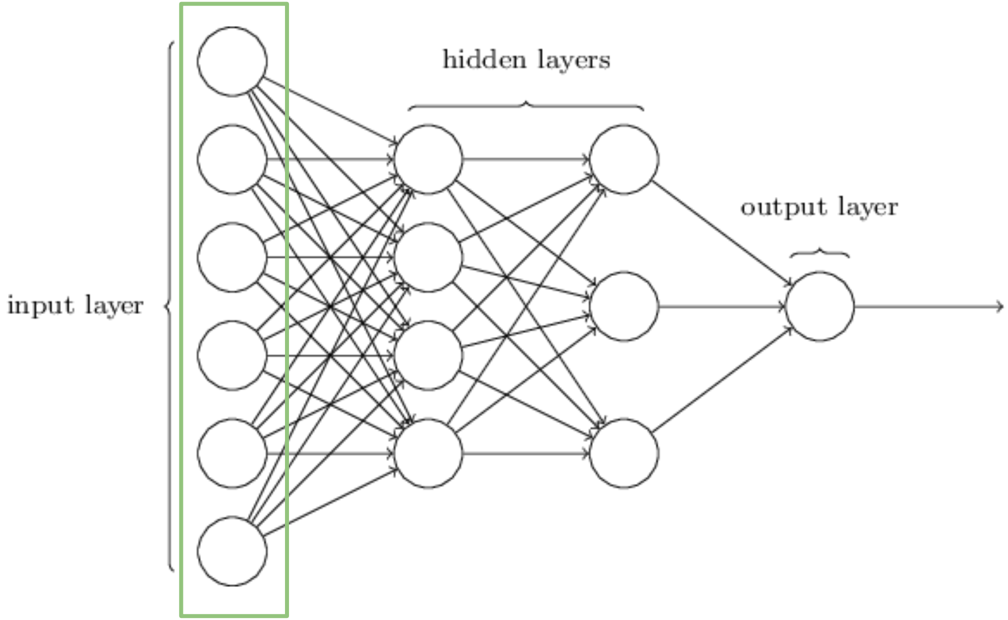
- Матрица весов: (2, m) в нашем примере
- (k, m) -- в общем случае.
- m -- количество нейронов на входе в слой, k -- на выходе

- Вектор смещения: длины 2
- k -- в общем случае

Нейронная сеть

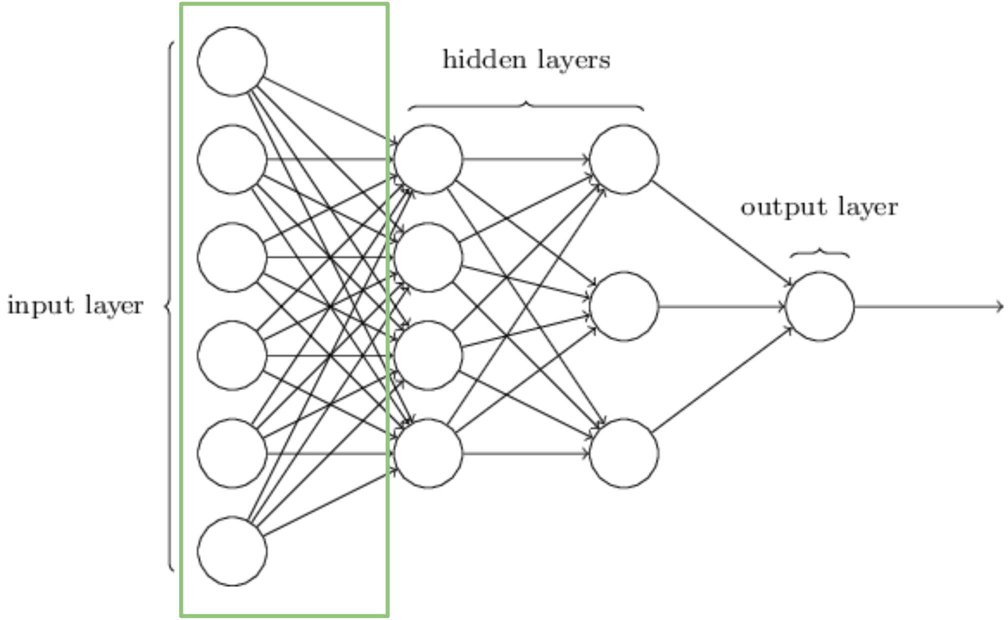


Нейронная сеть



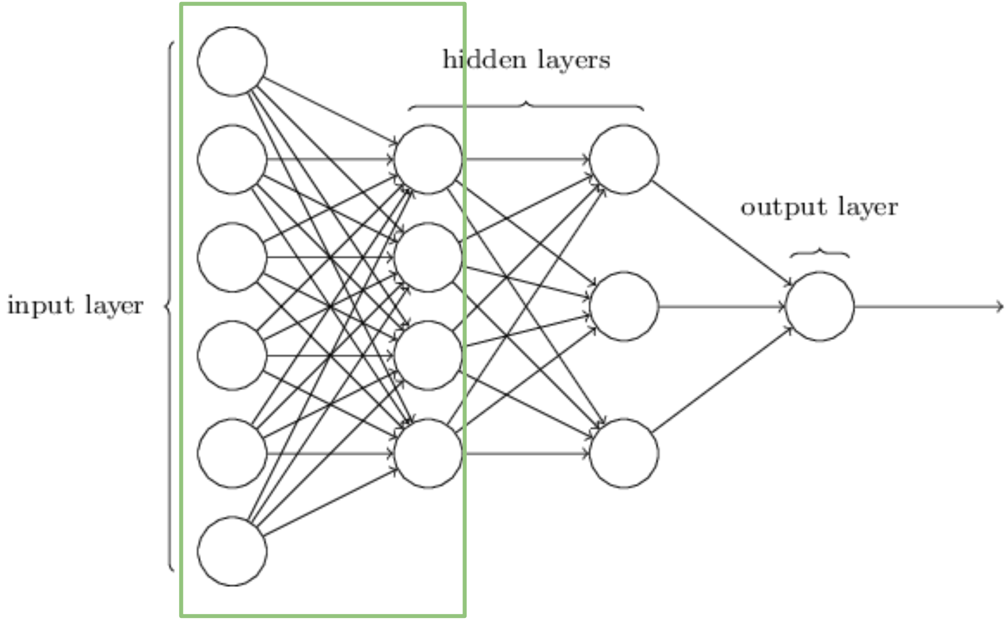
$$x$$

Нейронная сеть



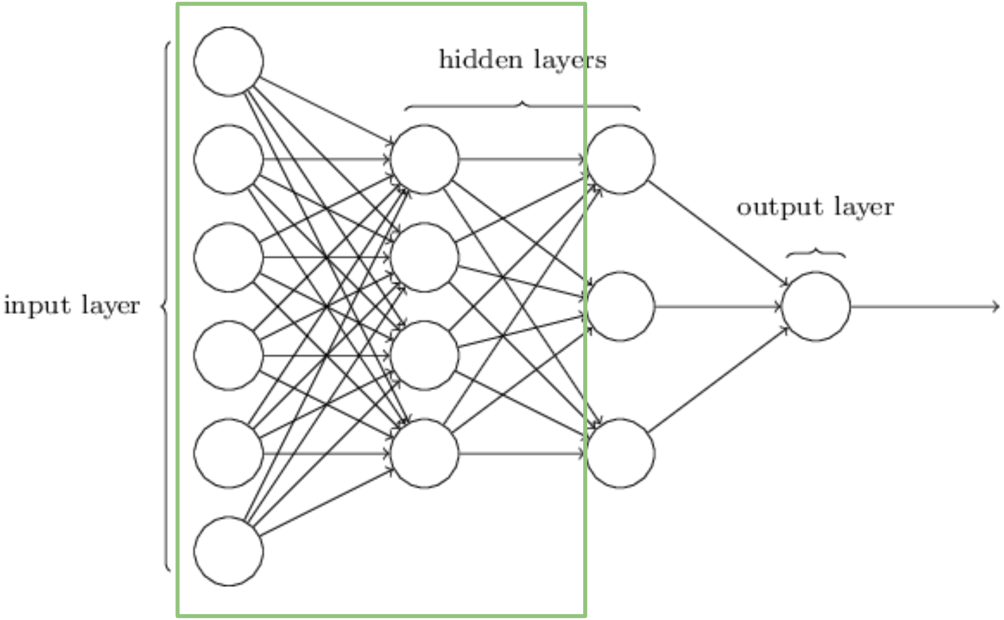
$$W_1 \boldsymbol{x} + \boldsymbol{b}_1$$

Нейронная сеть



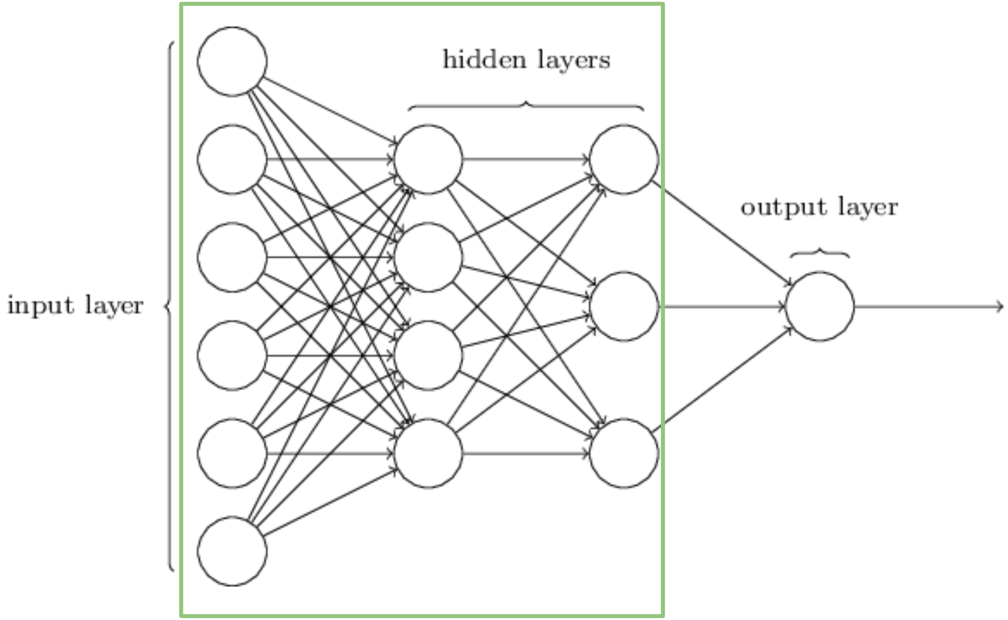
$$f(W_1\boldsymbol{x} + \boldsymbol{b}_1)$$

Нейронная сеть



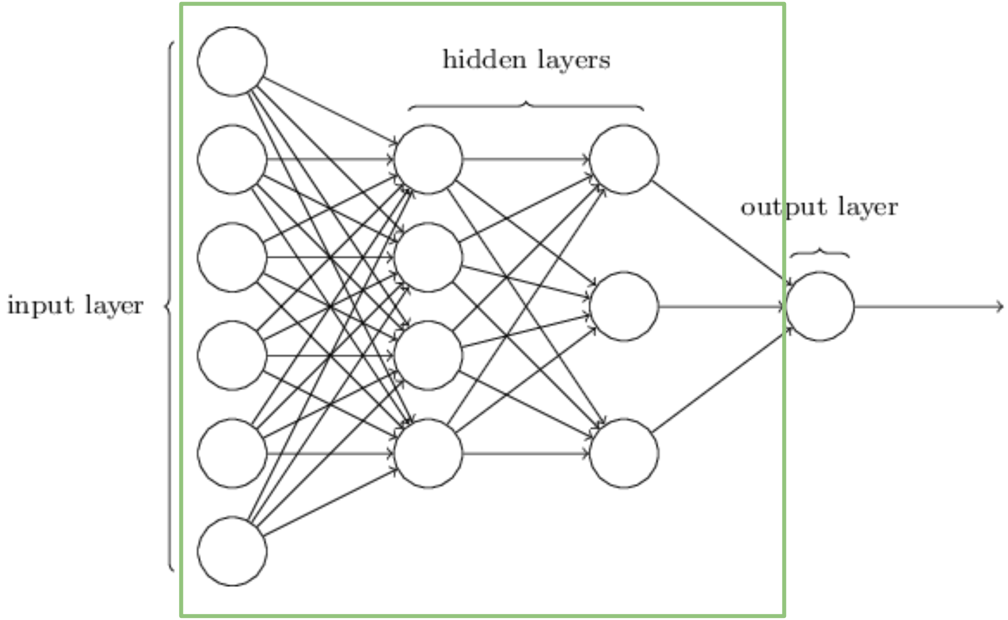
$$W_2 f(W_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2$$

Нейронная сеть



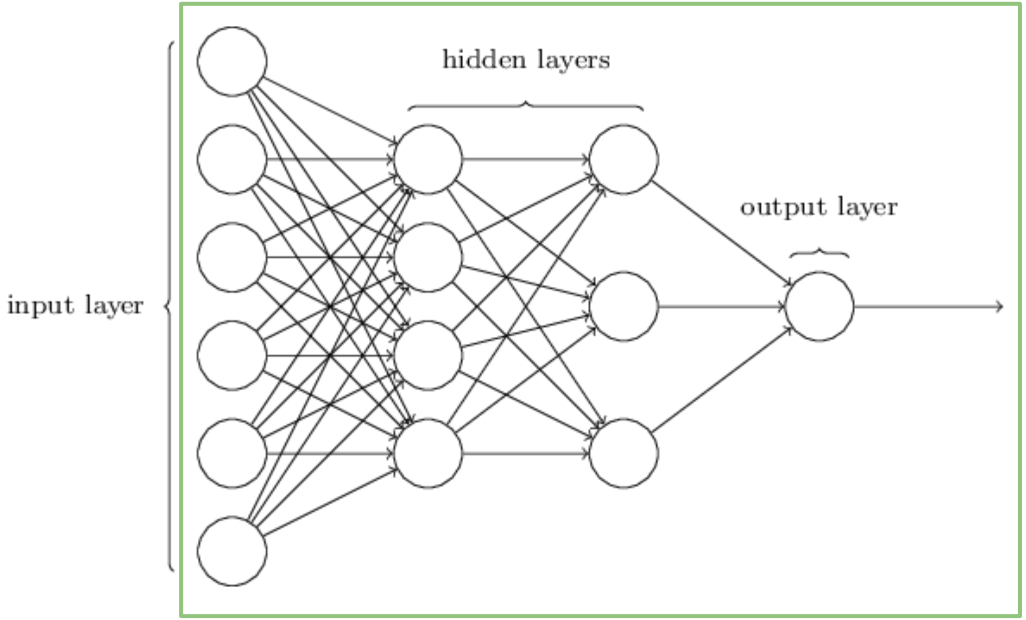
$$f(W_2 f(W_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2)$$

Нейронная сеть



$$W_3(f(W_2 f(W_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2) + \boldsymbol{b}_3$$

Нейронная сеть



$$f(W_3(f(W_2 f(W_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2) + \boldsymbol{b}_3)$$

Полносвязный слой на numpy

```
m = 10
k = 2

x = np.ones(m)
W = np.ones((k, m))
b = np.ones(k)
print(x.shape, W.shape, b.shape)
```

```
(10,) (2, 10) (2,)
```

Полносвязный слой на numpy

```
m = 10
k = 2

x = np.ones(m)
W = np.ones((k, m))
b = np.ones(k)
print(x.shape, W.shape, b.shape)
```

```
(10,) (2, 10) (2,)
```

```
np.dot(W, x) + b
```

```
array([11., 11.])
```

Как применить к нескольким объектам?

```
n = 3
m = 10
k = 2

x = np.ones((n, m))
W = np.ones((m, k))
b = np.ones(k)
print(x.shape, W.shape, b.shape)
```

```
(3, 10) (10, 2) (2,)
```

Batch dimension

Как применить к нескольким объектам?

```
n = 3
m = 10
k = 2

x = np.ones((n, m))
W = np.ones((m, k))
b = np.ones(k)
print(x.shape, W.shape, b.shape)
```

```
(3, 10) (10, 2) (2,)
```

```
np.matmul(x, W) + b
```

```
array([[11., 11.],
       [11., 11.],
       [11., 11.]])
```

Итог

- Применение полносвязного слоя эквивалентно матричному умножению входа на матрицу весов и добавление вектора смещения
- Соответственно применимы все матричные операции и векторизация в numpy
- Например посчитать выход слоя для нескольких объектов можно сразу

Skillbox

Обучение нейронных сетей

Мультиклассовая классификация

Skillbox

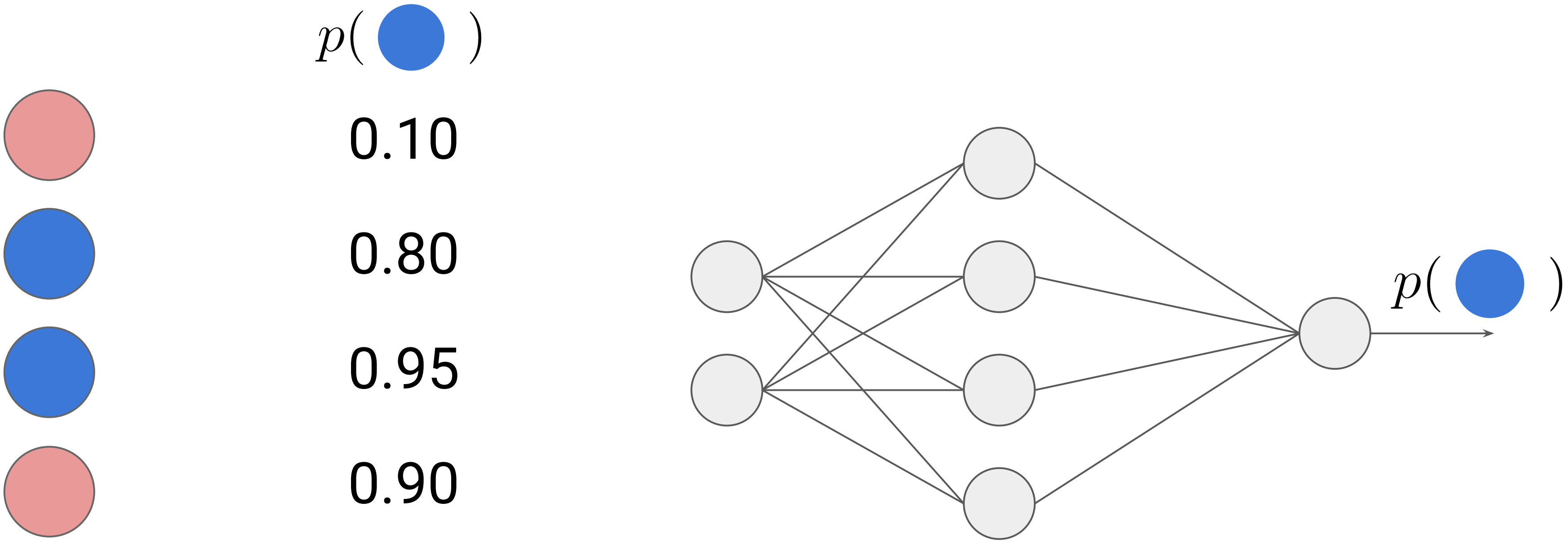
Домашнее задание

- Классификация рукописных цифр
- Датасет -- MNIST
- 60 000 изображений в трейне
- **10 классов**

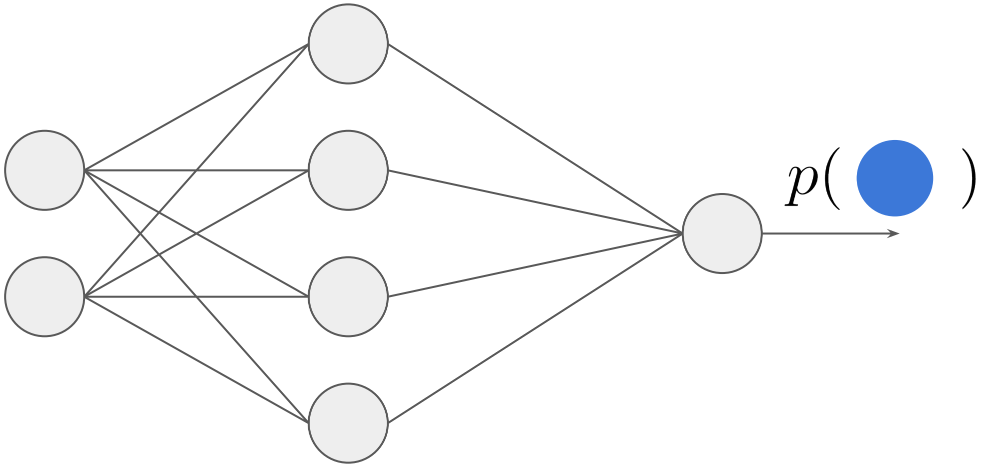
Домашнее задание

- Классификация рукописных цифр
- Датасет -- MNIST
- 60 000 изображений в трейне
- **10 классов**
  - Простое обобщение с 2х!
  - Сейчас это увидим

Skillbox

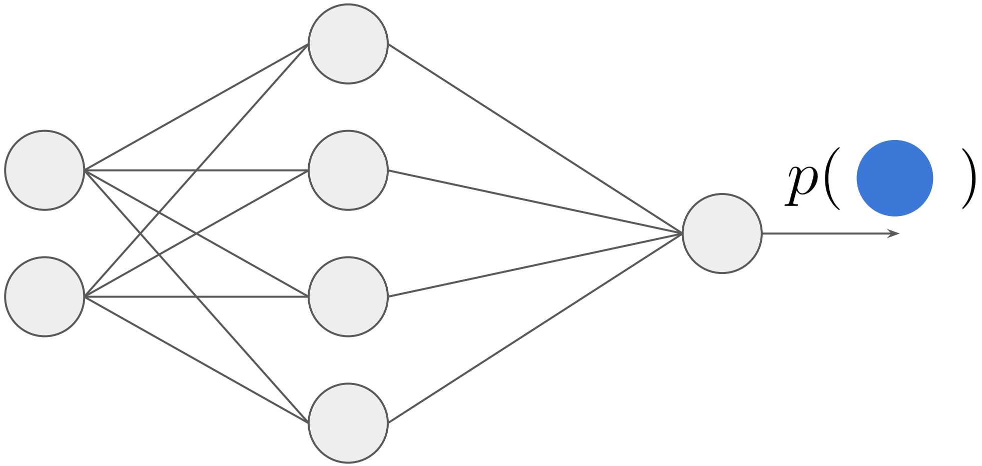


	$p(\ )$	$p(\ )$
	0.90	0.10
	0.20	0.80
	0.05	0.95
	0.10	0.90

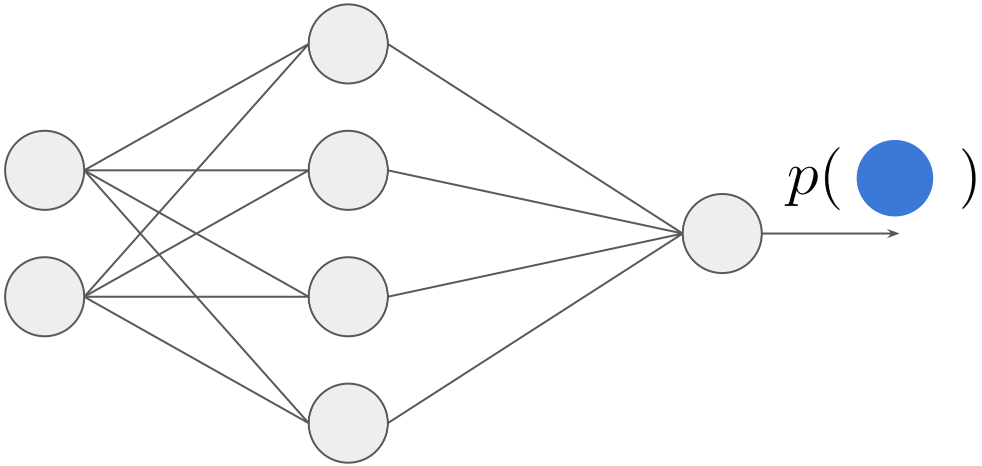


Skillbox

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

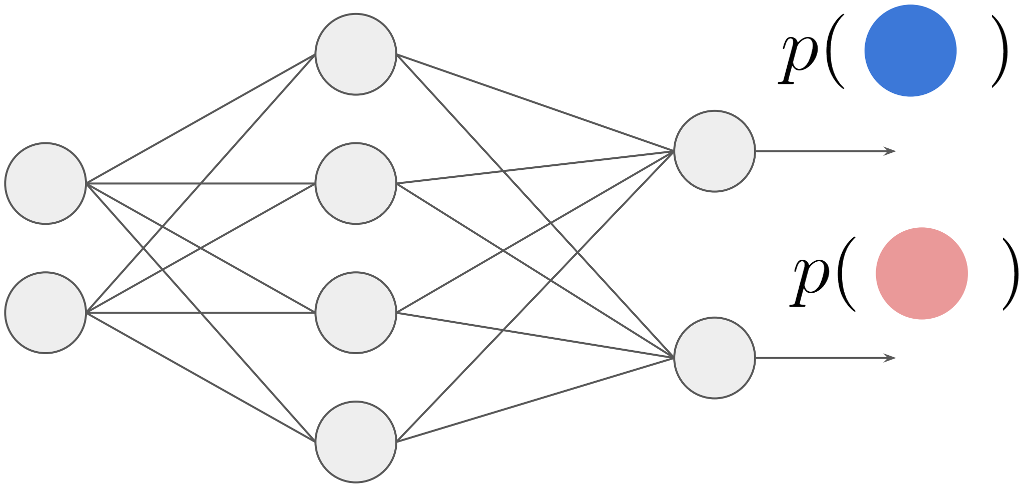


	$p(\quad)$	$p(\quad)$
	$\boxed{0.90}$	0.10
	0.20	$\boxed{0.80}$
	0.05	$\boxed{0.95}$
	$\boxed{0.10}$	0.90

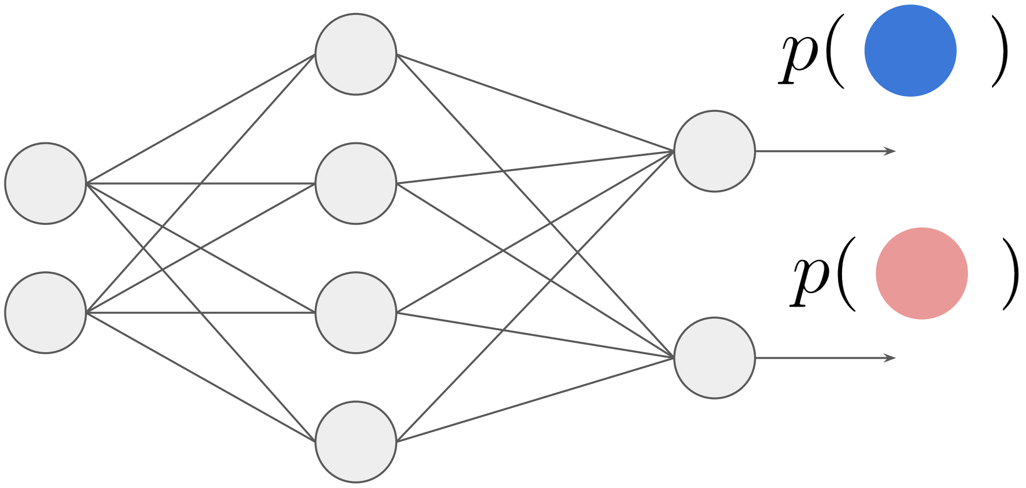


Можно сделать два выхода?

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



	$p(\;)$	$p(\;)$
	0.90	0.10
	0.20	**0.80**
	0.05	**0.95**
	0.10	0.90



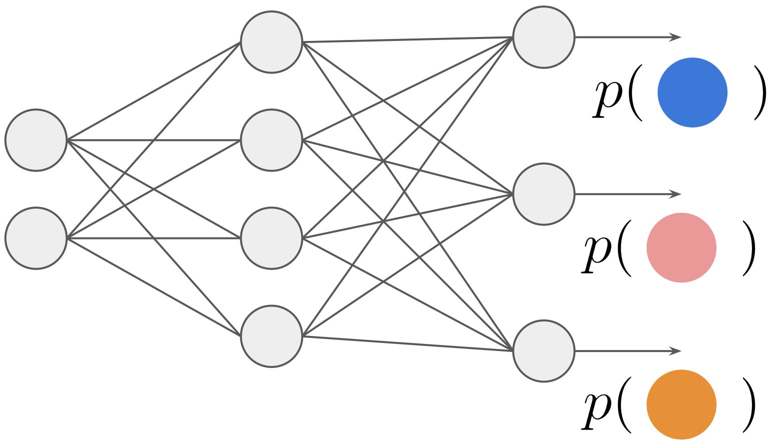
А если три класса?

Skillbox

	$p(\ )$	$p(\ )$	$p(\ )$
	0.90	0.05	0.05
	0.10	0.80	0.10
	0.05	0.92	0.03
	0.70	0.20	0.10

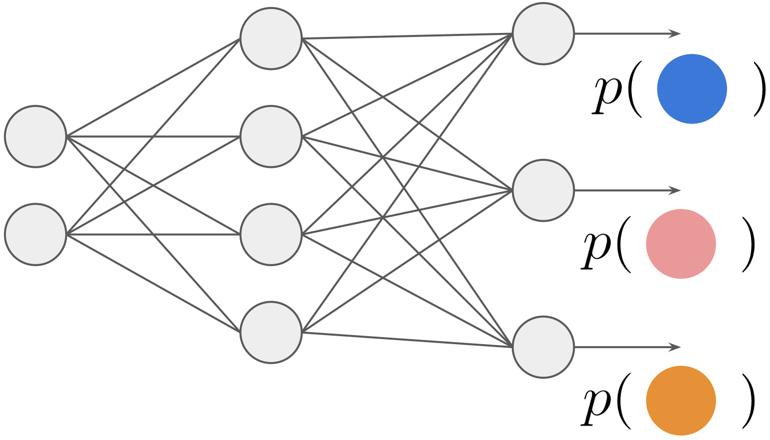
А если три класса?

	$p(\quad)$	$p(\quad)$	$p(\quad)$
	0.90	0.05	0.05
	0.10	0.80	0.10
	0.05	0.92	0.03
	0.70	0.20	0.10



А если три класса?

	$p(\ )$	$p(\ )$	$p(\ )$
	0.90	0.05	0.05
	0.10	0.80	0.10
	0.05	0.92	0.03
	0.70	0.20	0.10



А как считать ошибку?

	$p(\ )$	$p(\ )$	$p(\ )$
	0.90	0.05	0.05
	0.10	0.80	0.10
	0.05	0.92	0.03
	0.70	0.20	0.10

А как считать ошибку?

	$p(\ )$	$p(\ )$	$p(\ )$
	$\boxed{0.90}$	0.05	0.05
	0.10	$\boxed{0.80}$	0.10
	0.05	0.92	$\boxed{0.03}$
	$\boxed{0.70}$	0.20	0.10

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

А как считать ошибку?

	$p(\quad)$	$p(\quad)$	$p(\quad)$	$\hat{p}$
		0.05	0.05	0.90
	0.10		0.10	0.80
	0.05	0.92		0.03
		0.20	0.10	0.70

А как считать ошибку?

	$p(\ )$	$p(\ )$	$p(\ )$	$\hat{p}$
		0.05	0.05	$\boxed{0.90}$
	0.10		0.10	$\boxed{0.80}$
	0.05	0.92		$\boxed{0.03}$
		0.20	0.10	$\boxed{0.70}$

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

А как считать ошибку?

	$p(\;)$	$p(\;)$	$p(\;)$	$\hat{p}$
	0.90	0.05	0.05	0.90
	0.10	0.80	0.10	0.80
	0.05	0.92	0.03	0.03
	0.70	0.20	0.10	0.70

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

А как считать ошибку?

$p(\ )$	$p(\ )$	$p(\ )$	$\hat{p}$
0.90	0.05	0.05	0.90
0.10	0.80	0.10	0.80
0.05	0.92	0.03	0.03
0.70	0.20	0.10	0.70

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

А как считать ошибку?

	$p(\ )$	$p(\ )$	$p(\ )$	$\hat{p}$
	0.90	0.05	0.05	0.90
	0.10	0.80	0.10	0.80
	0.05	0.92	0.03	0.03
	0.70	0.20	0.10	0.70

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

А как считать ошибку? Аналогично случаю двух классов

$p(\ )$	$p(\ )$	$p(\ )$	$\hat{p}$
0.90	0.05	0.05	0.90
0.10	0.80	0.10	0.80
0.05	0.92	0.03	0.03
0.70	0.20	0.10	0.70

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

Какие условия на выходы сети?

	$p(\ )$	$p(\ )$	$p(\ )$	$\hat{p}$
	0.90	0.05	0.05	0.90
	0.10	0.80	0.10	0.80
	0.05	0.92	0.03	0.03
	0.70	0.20	0.10	0.70

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

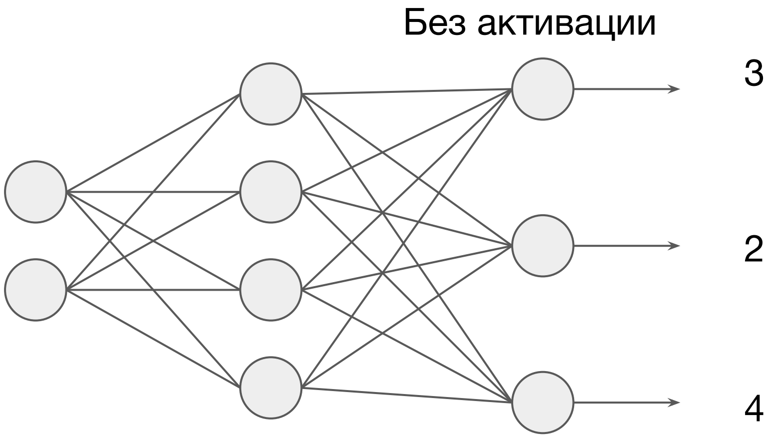
Какие условия на выходы сети? В сумме должна быть единица. Нужна нормировка ...

SoftMax

$$p_i^k = \frac{exp(logit_i^k)}{\sum_{j=1}^{m} exp(logit_i^j)}$$

SoftMax

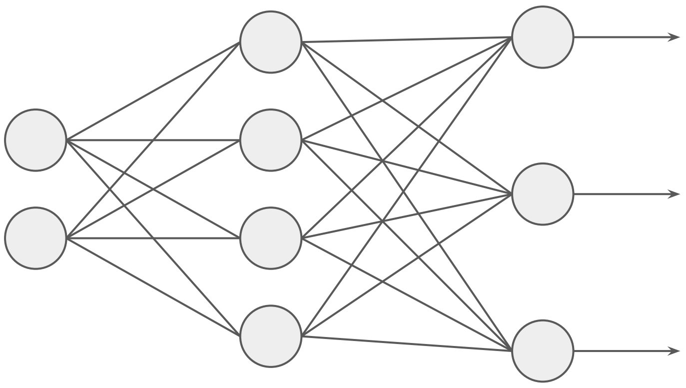
$$p_i^k = \frac{exp(logit_i^k)}{\sum_{j=1}^{m} exp(logit_i^j)}$$



SoftMax

$$p_i^k = \frac{exp(logit_i^k)}{\sum_{j=1}^{m} exp(logit_i^j)}$$

Без активации



3

2

4

exp(3) = 20

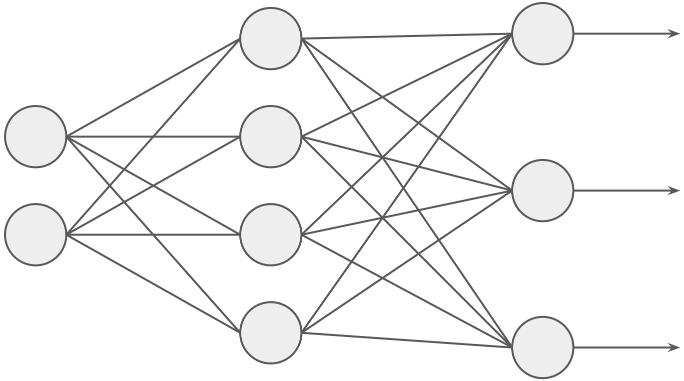
exp(2) = 7.38

exp(4) = 54.59

SoftMax

$$p_i^k = \frac{exp(logit_i^k)}{\sum_{j=1}^{m} exp(logit_i^j)}$$

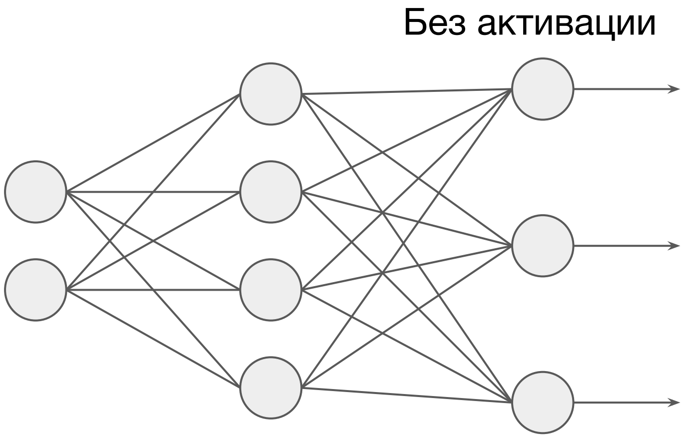
Без активации



3	exp(3) = 20	20 / (20 + 7.38 + 54.59)
2	exp(2) = 7.38	7.38 / (20 + 7.38 + 54.59)
4	exp(4) = 54.59	54.59 / (20 + 7.38 + 54.59)

SoftMax

$$p_i^k = \frac{exp(logit_i^k)}{\sum_{j=1}^{m} exp(logit_i^j)}$$



3	exp(3) = 20	20 / (20 + 7.38 + 54.59)	0.24
2	exp(2) = 7.38	7.38 / (20 + 7.38 + 54.59)	0.09
4	exp(4) = 54.59	54.59 / (20 + 7.38 + 54.59)	0.66

Итог

- Для мульти классовой классификации вместо сигмоиды используется SoftMax
- Он выполняет функцию нормировки выходов сети
    - Положительные
    - В сумме дают 1
- В качестве функции потерь используется известный нам логлосс:
    - Categorical cross entropy