Введение в глубинное обучение

Куликов В.А. в соавторстве с Лемпицким В.С.

Skoltech

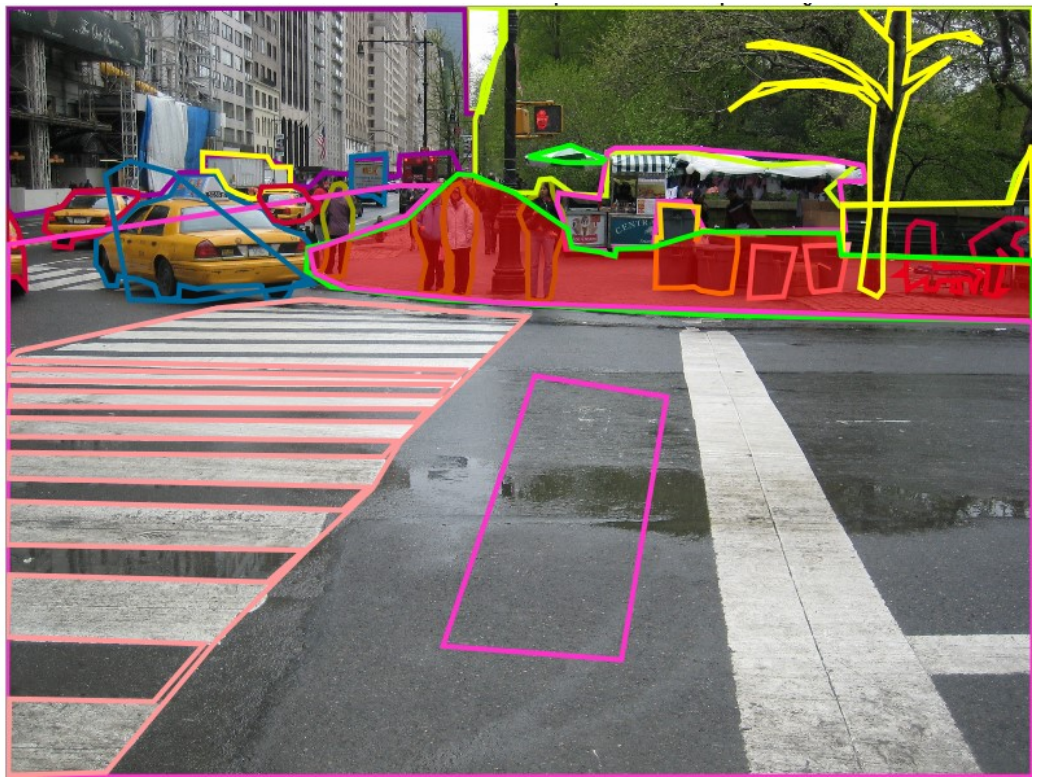
v.kulikov@skoltech.ru

О курсе

- Основы машинного обучения
  - Линейная регрессия
  - Логистическая регрессия
  - Нейронные сети
- Сверточные нейронные сети
  - Классификация изображений
  - Представление внутри сетей
- Глубокое обучение в компьютерном зрении
  - Семантическая сегментация
  - Детекторы объектов

Краткая история машинного обучения

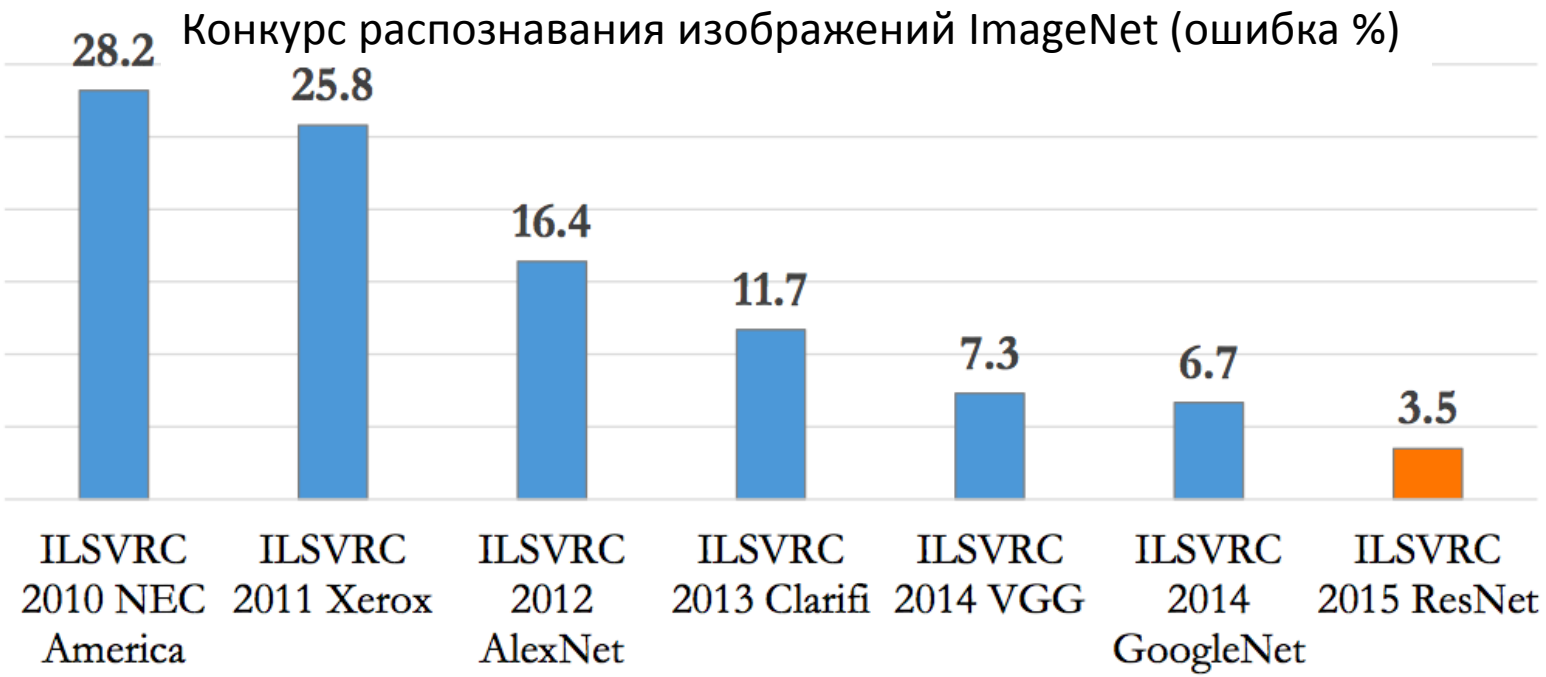
Семантические задачи (что есть на изображении)

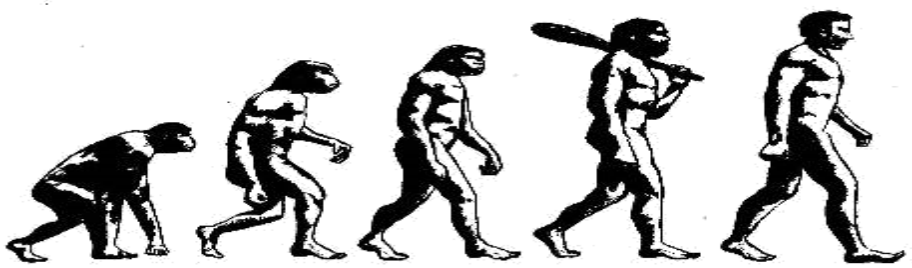


Метрические задачи (геометрия и описание сцены)

Семантическая задача: собака или кошка?

Глубинное обучение - революция





Этапы развития глубинного обучения

- 2009: Распознавание речи
- 2012: Компьютерное зрение
- 2014: Автоматический перевод

- 1. Данные (огромное количество данным, которые можно использовать для обучения)
- 2. Графические ускорители

Completed • Swag • 215 teams

Dogs vs. Cats

Wed 25 Sep 2013 – Sat 1 Feb 2014 (8 months ago)

Dashboard

Private Leaderboard - Dogs vs. Cats

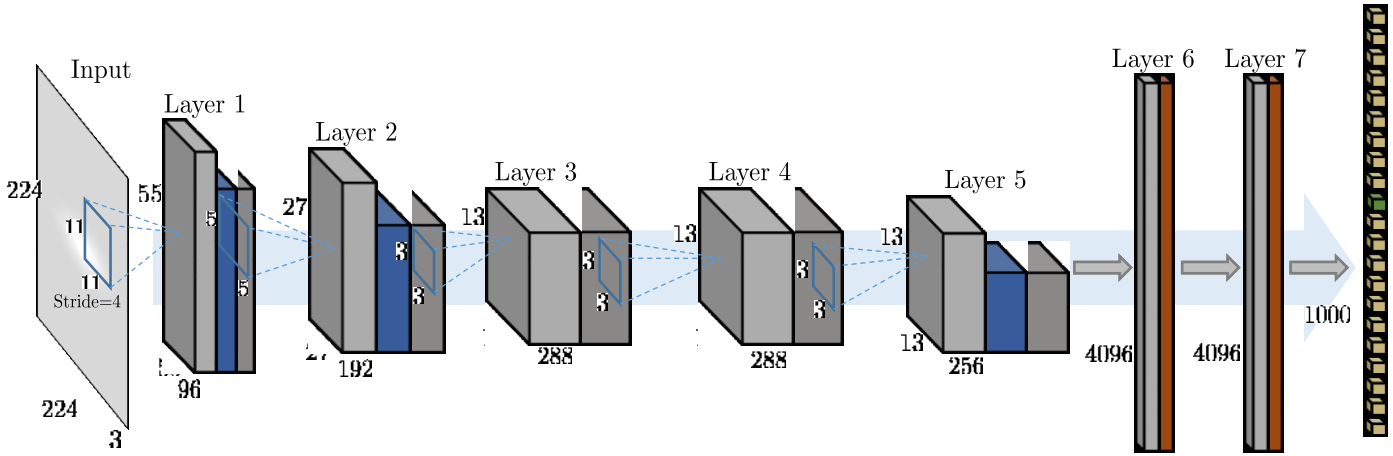
This competition has completed. This leaderboard reflects the final standings.

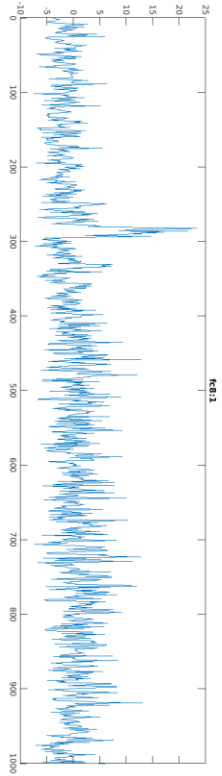
See someone

#	Δ1w	Team Name * in the money	Score	Entries	Last Submission UTC (Best – Last
1	—	Pierre Sermanet *	0.98914	5	Sat, 01 Feb 2014 21:43:19 (-
2	↑26	orchid *	0.98309	17	Sat, 01 Feb 2014 23:52:30
3	—	Owen	0.98171	15	Sat, 01 Feb 2014 17:04:40 (-
4	new	Paul Covington	0.98171	3	Sat, 01 Feb 2014 23:05:20
5	↓3	Maxim Milakov	0.98137	24	Sat, 01 Feb 2014 18:20:58

Текущий чемпион по классификации изображений: сверточная нейронная сеть

- Операции:
  - Свертки
  - Pooling (изменение разрешения)
  - Нелинейности
  - Произведение матриц





Задачи машинного обучения

- Обучение с учителем (supervised) (известны ответы)
  - Классификация
  - Регрессия

- Обучение без учителя (unsupervised) (нету ответов)
  - Кластеризация
  - Автоэнкодеры
  - Поиск аномалий

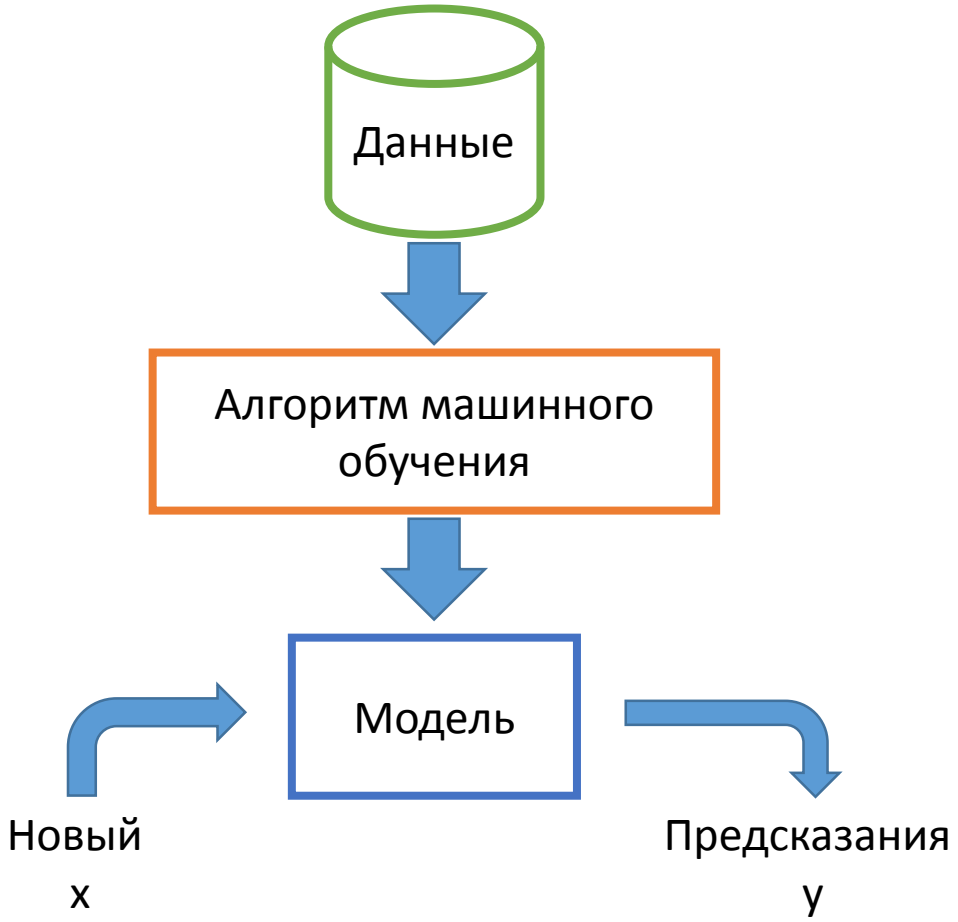
Задача предсказания цены дома по его параметрам

Задача: построить зависимость между параметрами дома и его ценой

Даны пары: площадь, цена

Площадь	Цена
1180	221900
2570	538000
770	180000
1960	604000

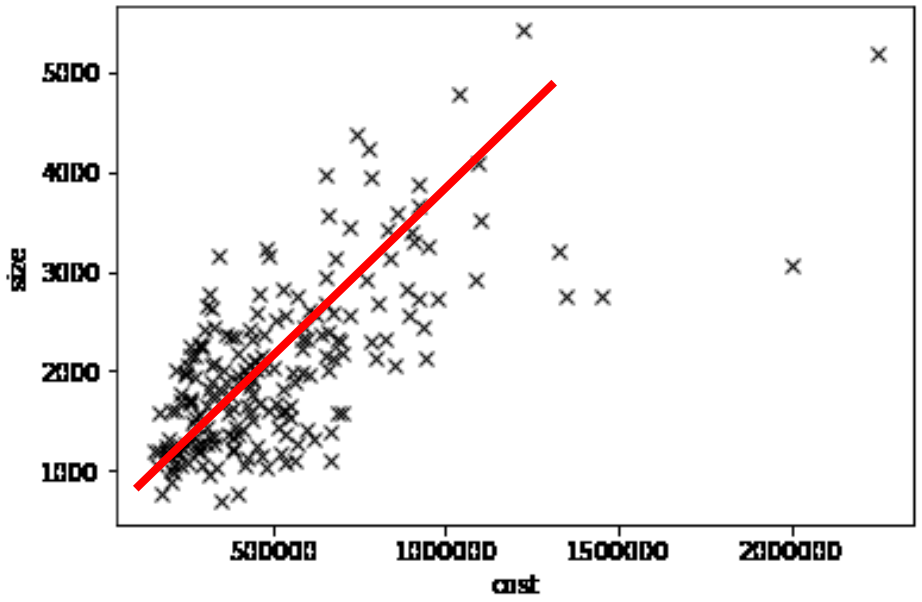
Линейная регрессия, понятие модели



Линейная модель

- $h_w(x) = w_0 + w_1 x$

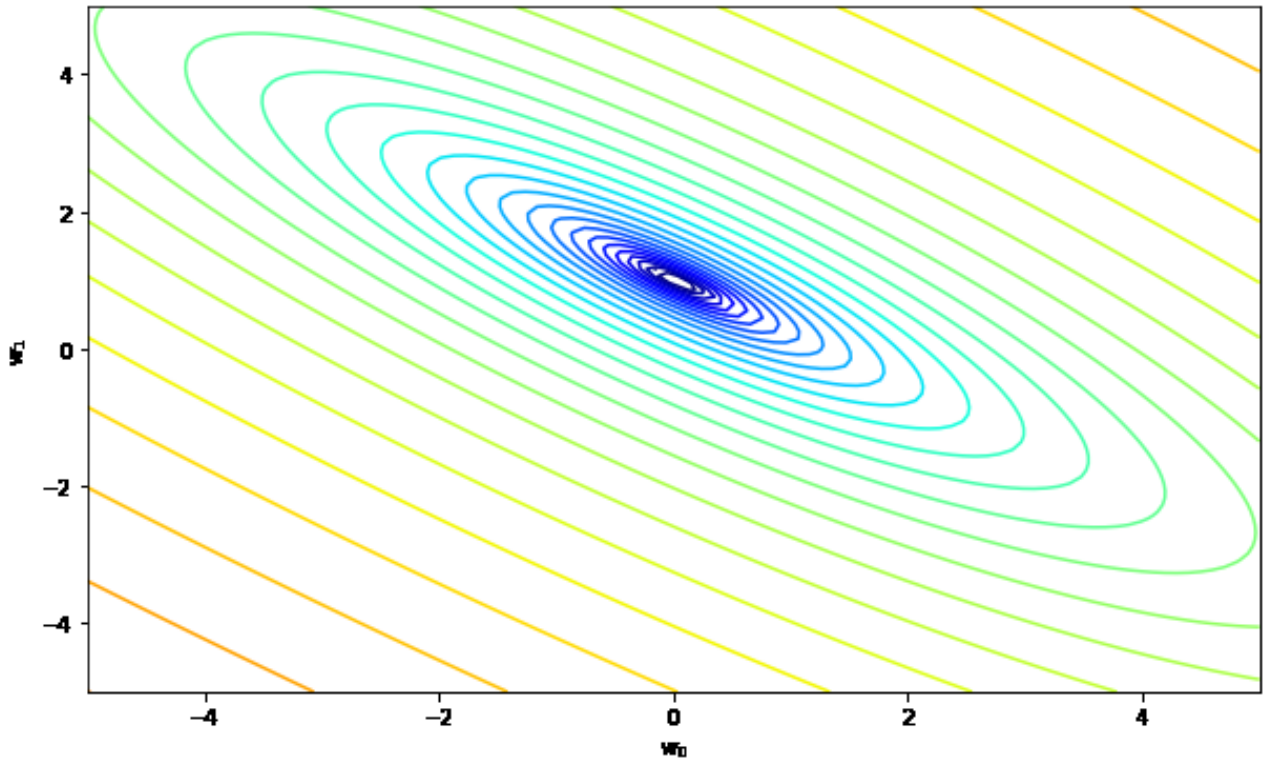
w_0 и w_1 праметры модели



Функция потерь (Loss function)

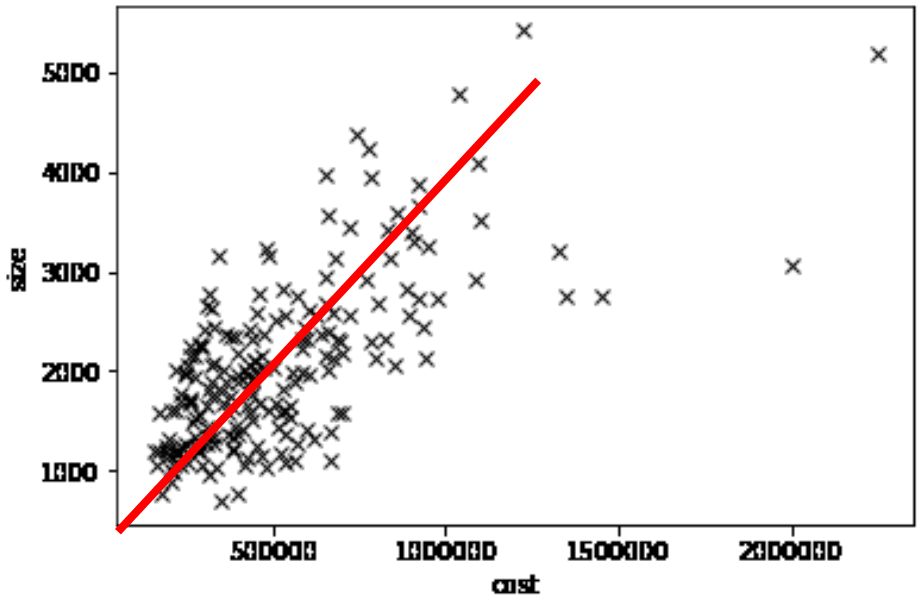
- $J(w) = \frac{1}{2N}\sum_{i=1}^{N}(h_w(x_i) - y_i)^2$

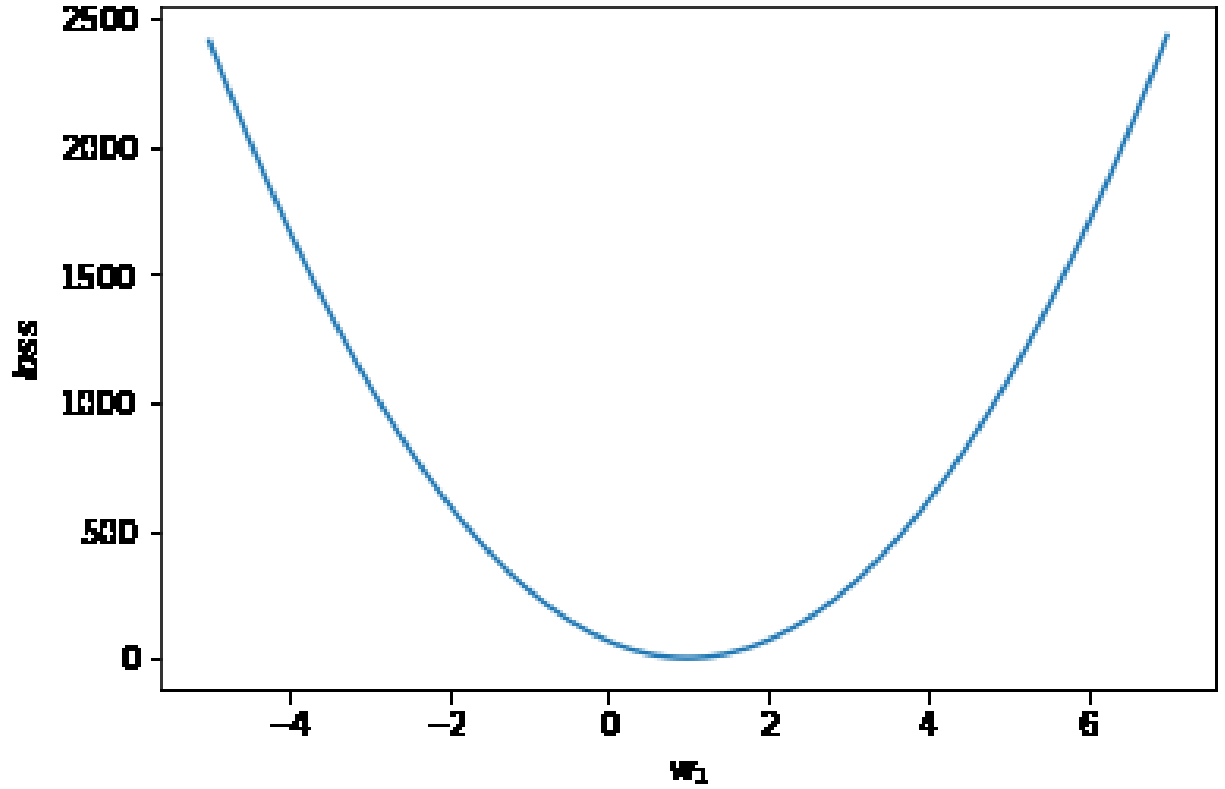
- Выведем функцию $J(w)$



Одномерный случай

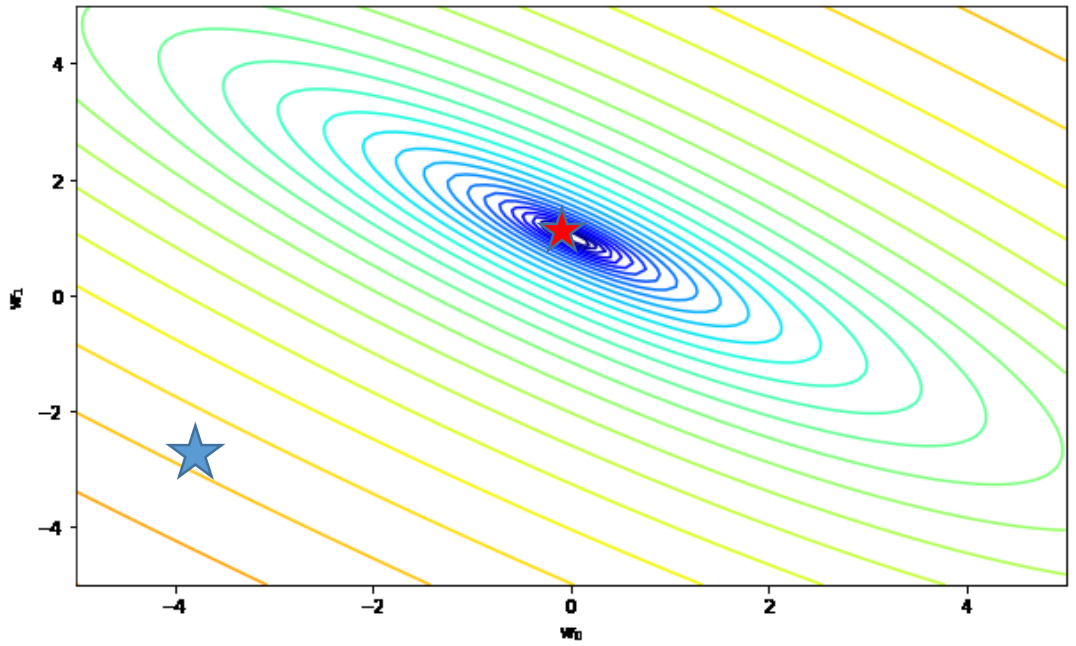
- Модель
  - $h_w(x) = w_1 x$





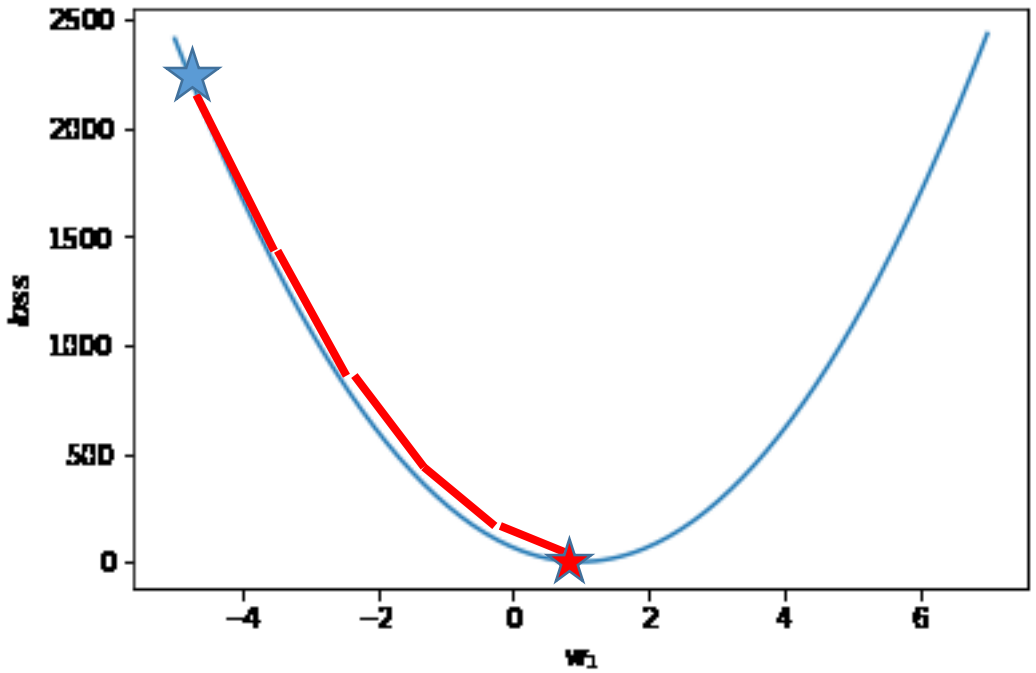
Поиск оптимального решения

- Модель
  - $h_w(x) = w_0 + w_1 x$
- Параметры
  - (w_0, w_1)
- Функция потерь
  - $J(w) = \frac{1}{2N} \sum_{i=1}^{N} (h_w(x_i) - y_i)^2$
- Задача
  - $\min_{w_0, w_1} J(w)$



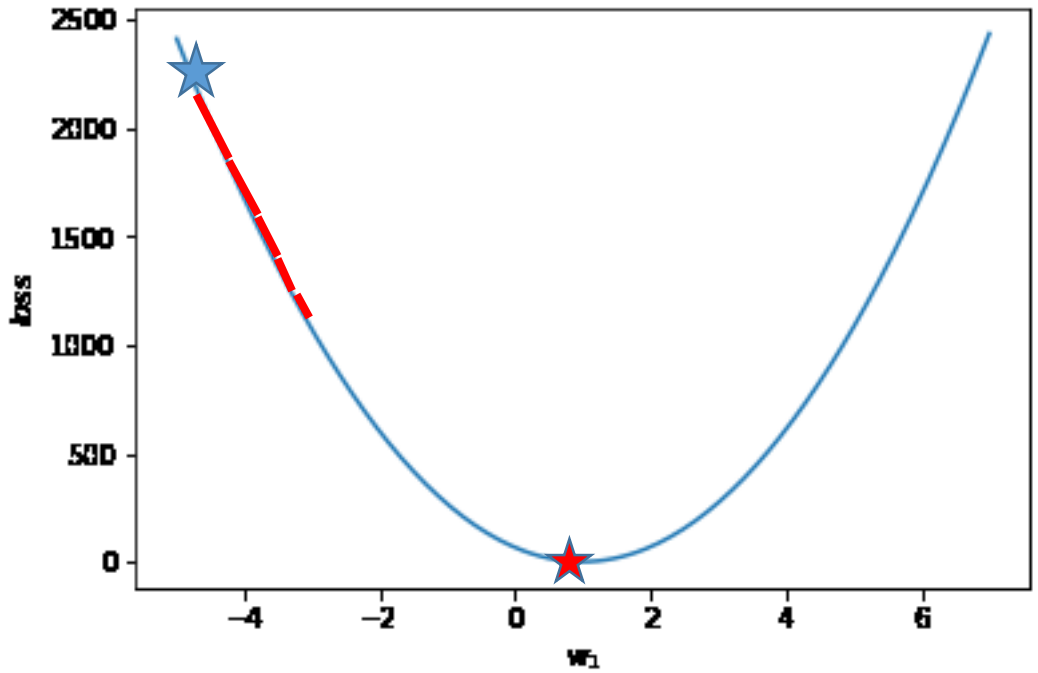
Градиентный спуск

- Модель
  - $h_w(x) = w_1 x$
- Параметры
  - Модели w_1, Алгоритма lr
- For n_iter:
  - $J(w) = \frac{1}{2N}\sum_{i=1}^{N}(h_w(x_i) - y_i)^2$
  - Градиент функции потерь:
  - $\frac{dJ}{dw_1} = \frac{1}{N}\sum_{i=1}^{N}(w_1 x_i - y_i)x$
  - $w_1 = w_1 - lr\frac{dJ}{dw_1}$



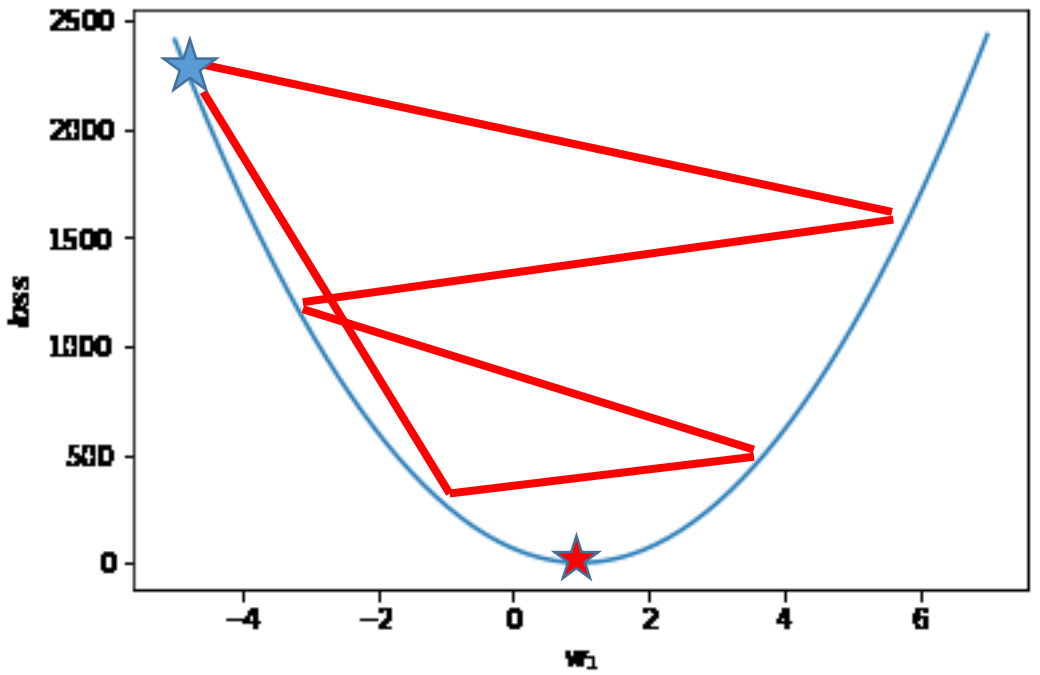
Градиентный спуск если lr маленький

- Модель
  - $h_w(x) = w_1 x$
- Параметры
  - Модели w_1, Алгоритма lr
- For n_iter:
  - $J(w) = \frac{1}{2N}\sum_{i=1}^{N}(h_w(x_i) - y_i)^2$
  - Градиент функции потерь:
  - $\frac{dJ}{dw_1} = \frac{1}{N}\sum_{i=1}^{N}(w_1 x_i - y_i)x$
  - $w_1 = w_1 - lr\frac{dJ}{dw_1}$



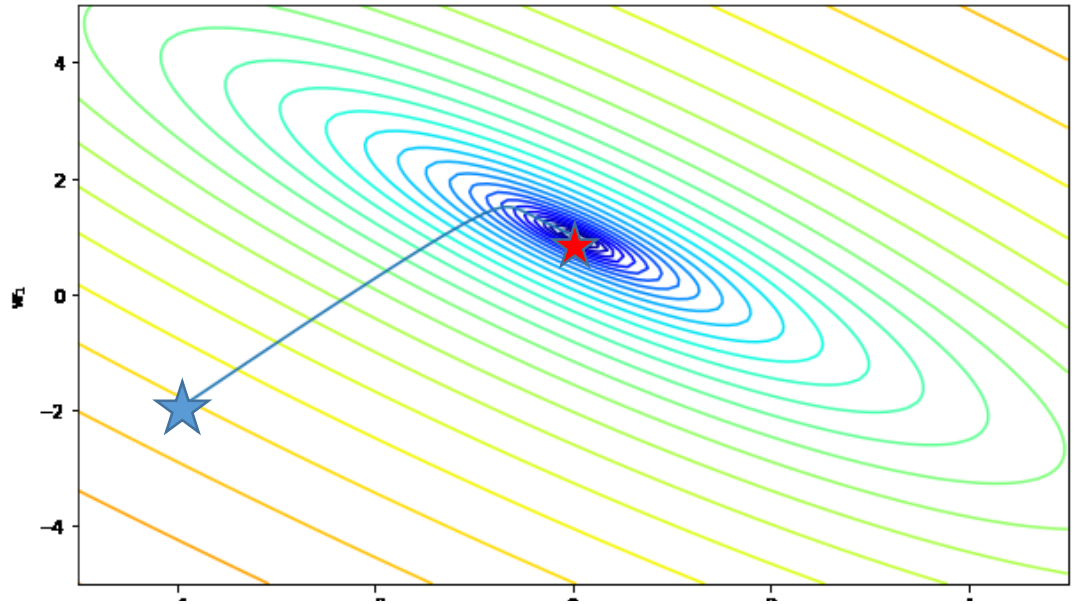
Градиентный спуск если lr большой

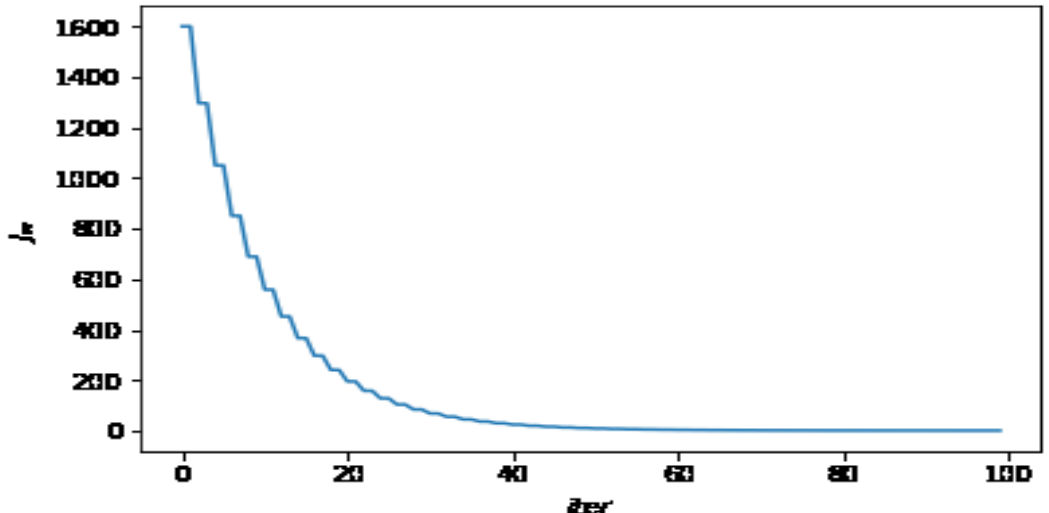
- Модель
  - $h_w(x) = w_1 x$
- Параметры
  - Модели w_1, Алгоритма lr
- For n_iter:
  - $J(w) = \frac{1}{2N}\sum_{i=1}^{N}(h_w(x_i) - y_i)^2$
  - Градиент функции потерь:
  - $\frac{dJ}{dw_1} = \frac{1}{N}\sum_{i=1}^{N}(w_1 x_i - y_i)x$
  - $w_1 = w_1 - lr\frac{dJ}{dw_1}$



Градиентный спуск от двух параметров

- Модель
  - $h_w(x) = w_0 + w_1 x$
- Параметры
  - (w_0, w_1),lr
- For n_iter:
  - $J(w) = \frac{1}{2N}\sum_{i=1}^{N}(h_w(x_i) - y_i)^2$
  - $\frac{dJ}{dw_0} = \frac{1}{N}\sum_{i=1}^{N}\left((h_w(x_i) - y_i\right)$
  - $\frac{dJ}{dw_1} = \frac{1}{N}\sum_{i=1}^{N}\left((h_w(x_i) - y_i\right)x$
  - $w_0 = w_0 - lr\frac{dJ}{dw_0}$
  - $w_1 = w_1 - lr\frac{dJ}{dw_1}$





Практика линейная регрессия

- 1.Реализовать функцию потерь
- 2.Реализовать производные функции потерь по параметрам

Многомерная линейная регрессия

- Модель: $h_w(x) = w_0 + w_1 x_1 + w_2 x_2$

Площадь	Год постройки	Цена
1180	1980	221900
200	2010	138000
770	2011	180000
1960	1990	604000

Скаляры, векторы, матрицы и тензоры

- $h_w(x) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$
- Пусть $x_0 = 1$
- Тогда:

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} \in R^{n+1} \qquad \mathrm{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \dots \\ w_n \end{bmatrix} \in R^{n+1} \qquad [w_0 \quad w_1 w_2 \quad \dots \quad w_n \,] \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

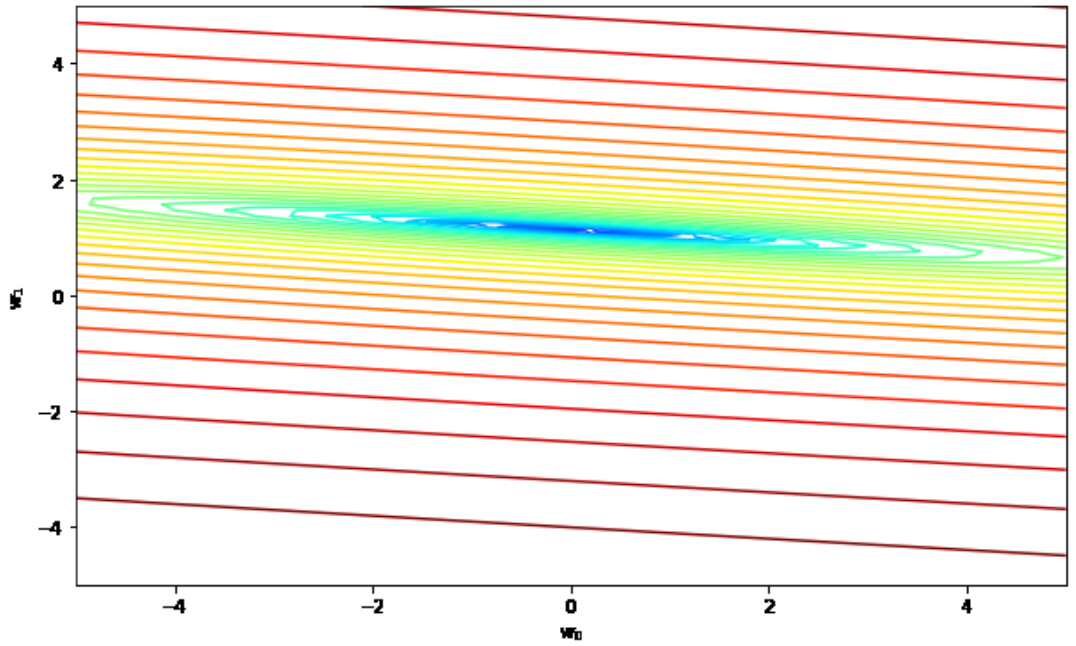
- $h_w(x) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = w^T x$

Многомерный градиентный спуск

- Модель: $h_w(x) = w^T x = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$
- Параметры: $w = w_0, w_1, \ldots, w_n$
- Функция потерь
  - $J(w) = \frac{1}{2N} \sum_{i=1}^{N} (h_w\left(x^{(i)}\right) - y^{(i)})^2$ - i номер примера в выборке
- Вычисляем частные производные:
  - $\frac{dJ}{dw_j} = \frac{1}{N} \sum_{i=1}^{N} (h_w\left(x^{(i)}\right) - y^{(i)})\, x_j^{(i)}$
- Градиентный спуск:
  - $w_j = w_j - lr \frac{1}{N} \sum_{i=1}^{N} (h_w\left(x^{(i)}\right) - y^{(i)})\, x_j^{(i)}$

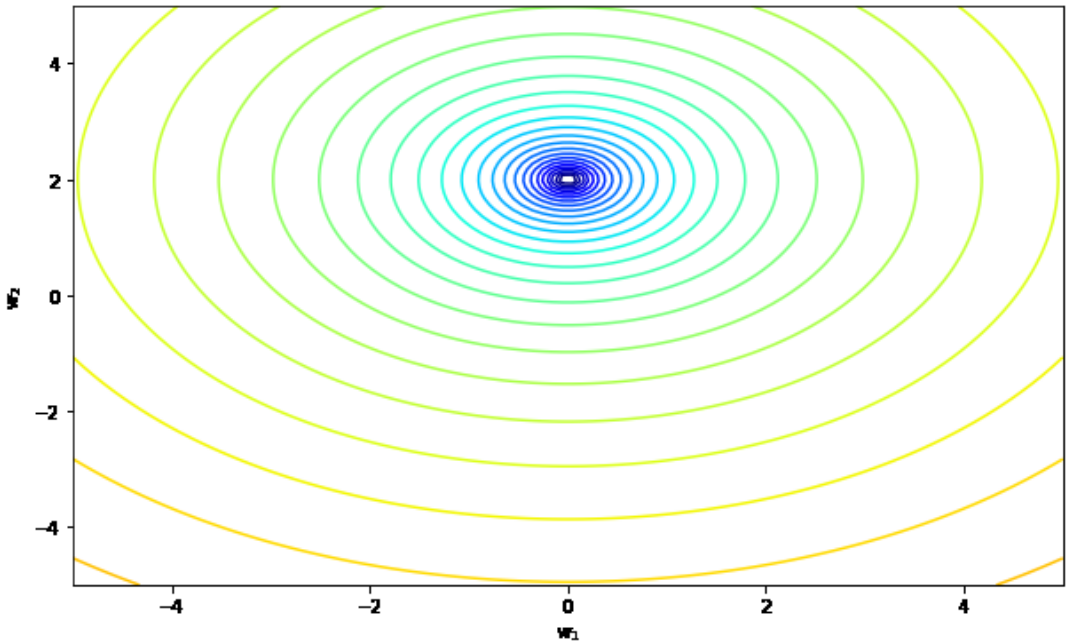
Многомерный градиентный спуск

- Модель: $h_w(x) = w_1 x_1 + w_2 x_2$
- Проблема:
  - Разные масштабы величин по осям
  - Надо выбрать маленький lr (скорость обучения)



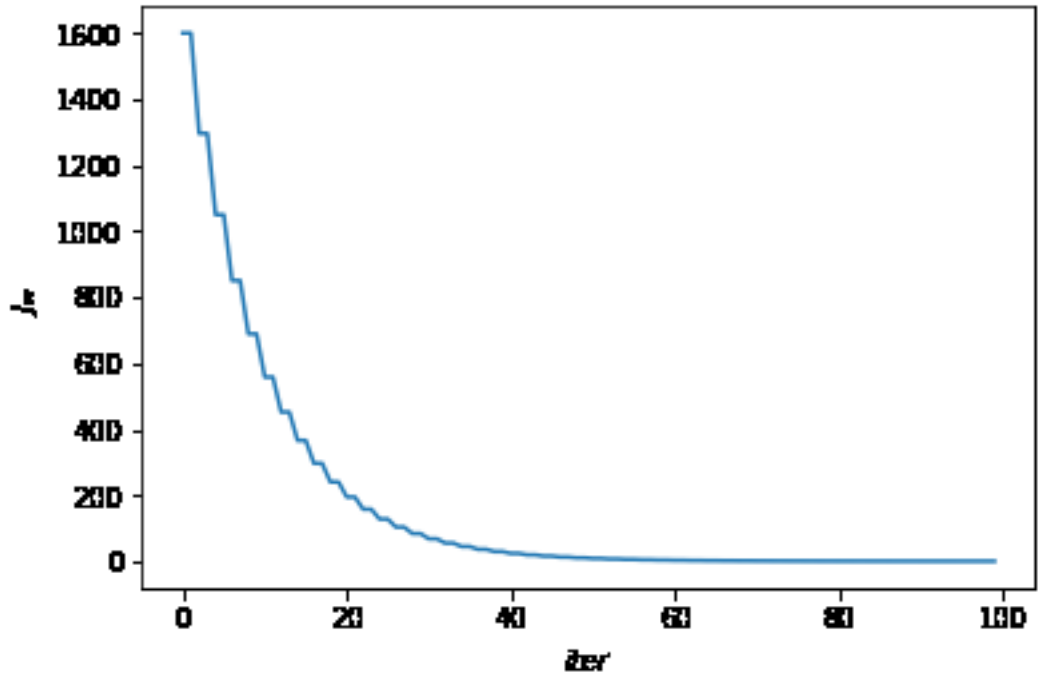
Нормирование признаков

- Модель: $h_w(x) = w_1 x_1 + w_2 x_2$
- Признаки: x
- $x = (x - mean(x))/std(x)$
- $mean(x) = \frac{1}{N}\sum_i x^{(i)}$
- $std(x) = \sqrt{\frac{1}{N}\sum_i (x^{(i)} - mean(x))^2}$



Выбор параметра lr в градиентном спуске

- Функция потерь должна убывать от количества итераций
- Если увеличивается – выбрать lr меньше
- Если не изменяется, или изменяется слишком медленно – увеличить lr



Логистическая регрессия

Задача: по цене и площади дома сказать его категорию

Даны параметры: площадь, цена, категория

Площадь	Цена	Категория
1180	221900	1
2570	538000	1
770	180000	0
1960	604000	1

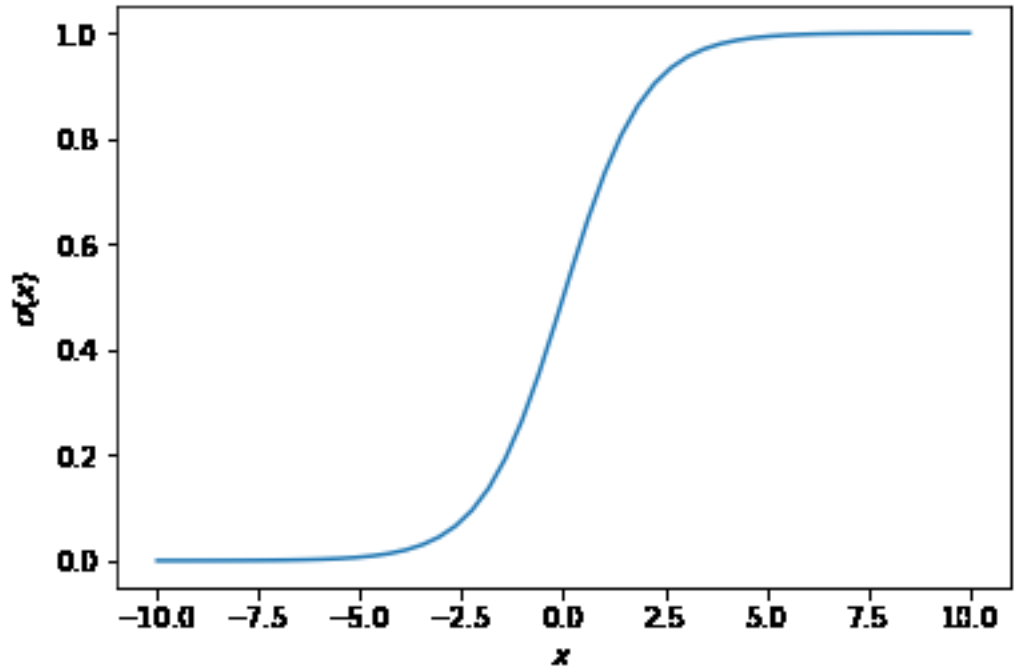
Логистическая регрессия (классификация)

- $y \in \{0,1\}$
- Где:
  - 0: «Отрицательный класс»
  - 1: «Положительный»
- Примеры:
  - не качественный / качественный дом по его характеристикам
  - нету/есть заболевание по анализу крови
  - спам/не спам – электронная почта и т.д.

- Нам нужна модель:
  - $0 \leq h_w(x) \leq 1$
- Модель линейной регрессии может принимать значения от [-inf,+inf]
- Наложить поверх линейной модели дополнительную функцию

Сигмоид (Sigmoid function)

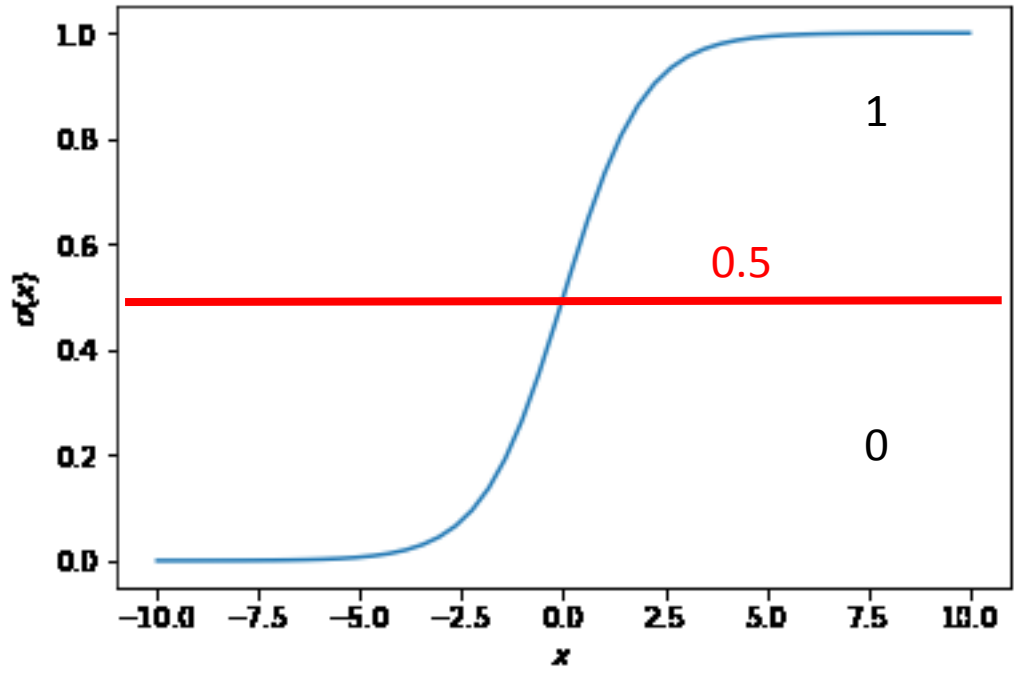
- $\sigma(x) = \frac{1}{1+e^{-x}}$
- Свойства:
  - $x \to -inf, \sigma(x) \to 0$
  - $y \to +inf, \sigma(x) \to 1$



Логистическая регрессия (классификация)

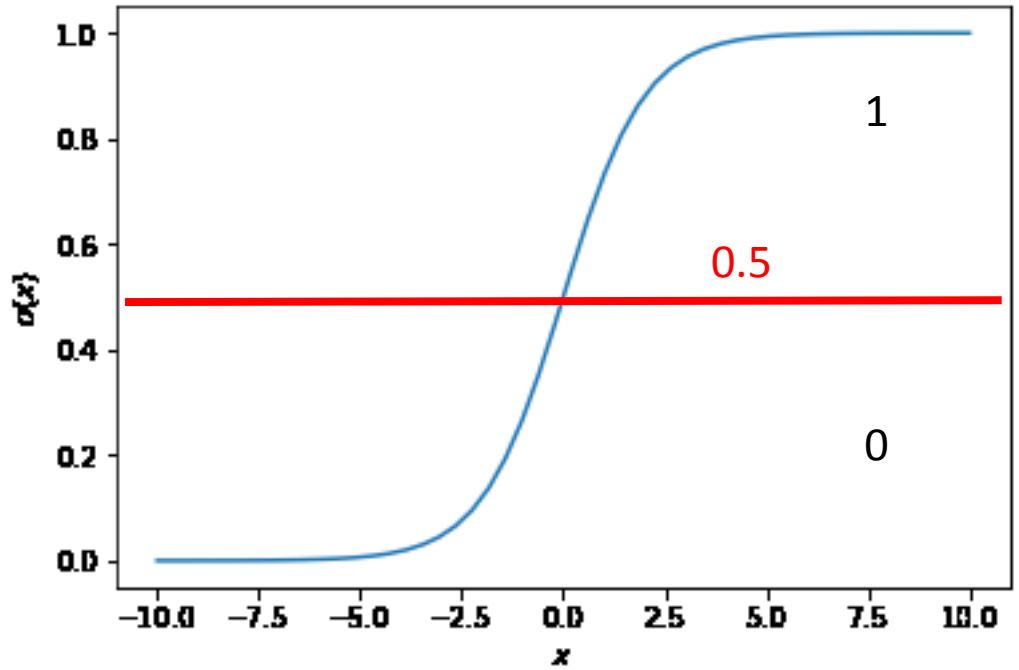
- Модель:
  - $h_w(x) = \sigma(w^T x)$
  - $h_w(x) = \frac{1}{1+e^{-w^T x}}$

- Сейчас выход нашей модели является оценкой вероятности, что y=1 при наблюдении x
- $h_w(x) = P(y = 1|x; w)$
- $P(y = 0|x; w) = 1 - h_w(x) = P(y = 1|x; w)$

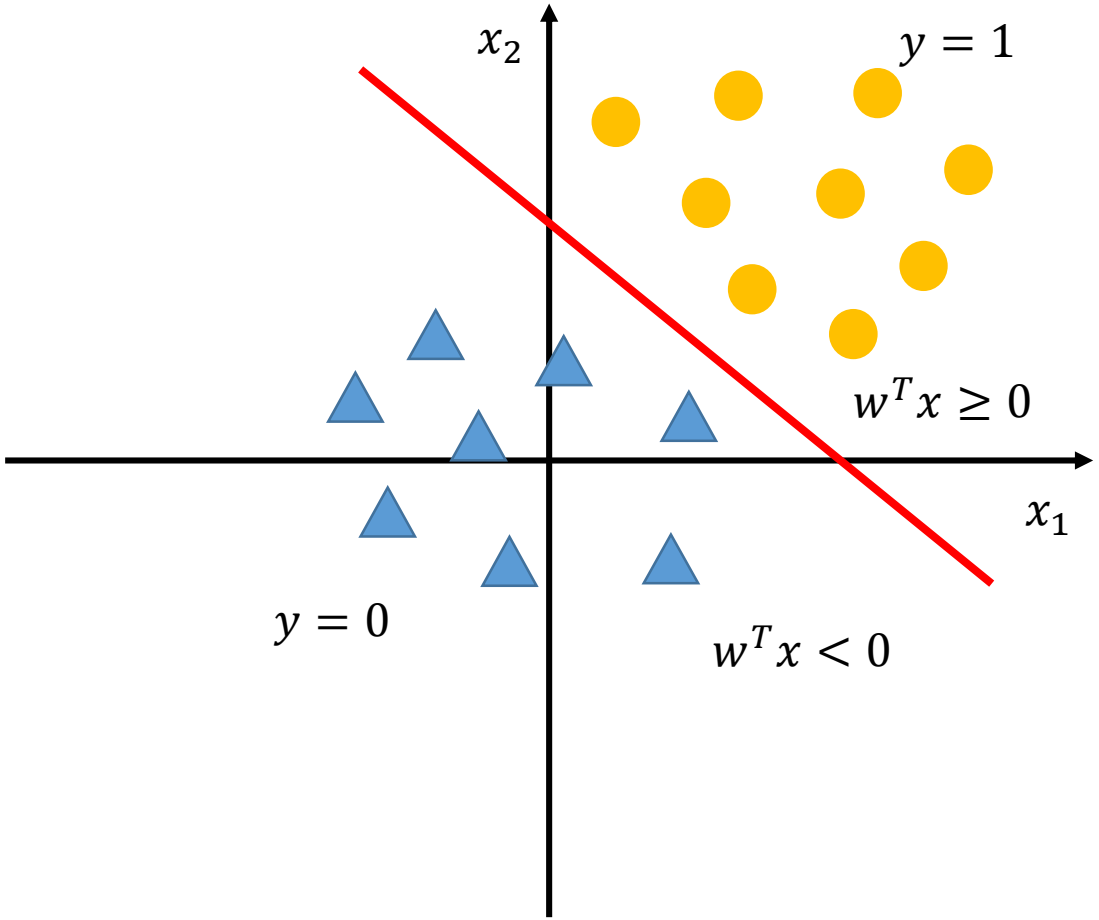


Решающее правило

- Модель:
  - $h_w(x) = \sigma(w^T x) = P(y = 1|x; w)$
- Пусть $y = 1$ если $h_w(x) \geq 0.5$
  - $w^T x \geq 0$
  - $y = 0$ если $h_w(x) < 0.5$
  - $w^T x < 0$
- Что это означает на практике?

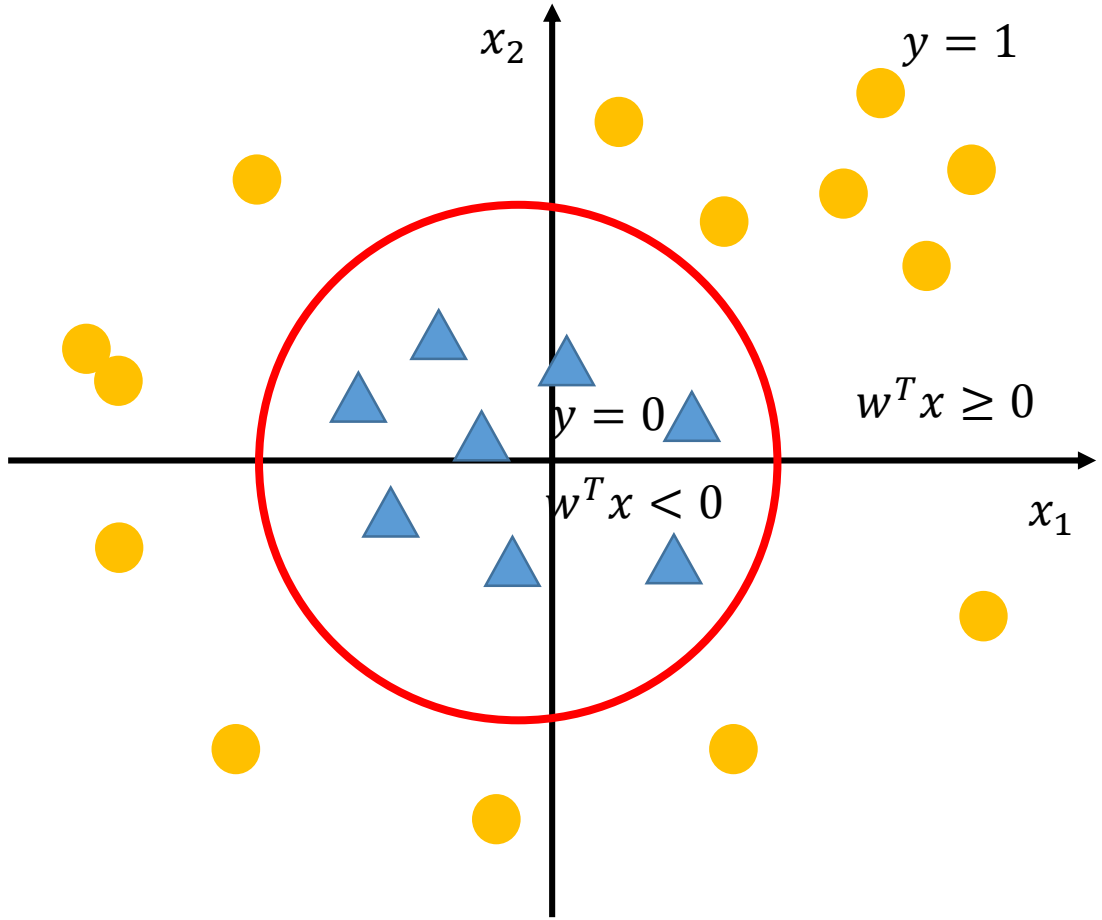


Решающее правило



- $h_w(x) = \sigma(w^T x)$
- $w^T x = w_0 + w_1 x_1 + w_2 x_2$
- Прямая!
- Классы линейно разделимы

Решающее правило



- Классы линейно разделимы
- $h_w(x) = \sigma(w^T x)$?
- $w^T x = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2$
- Окружность!

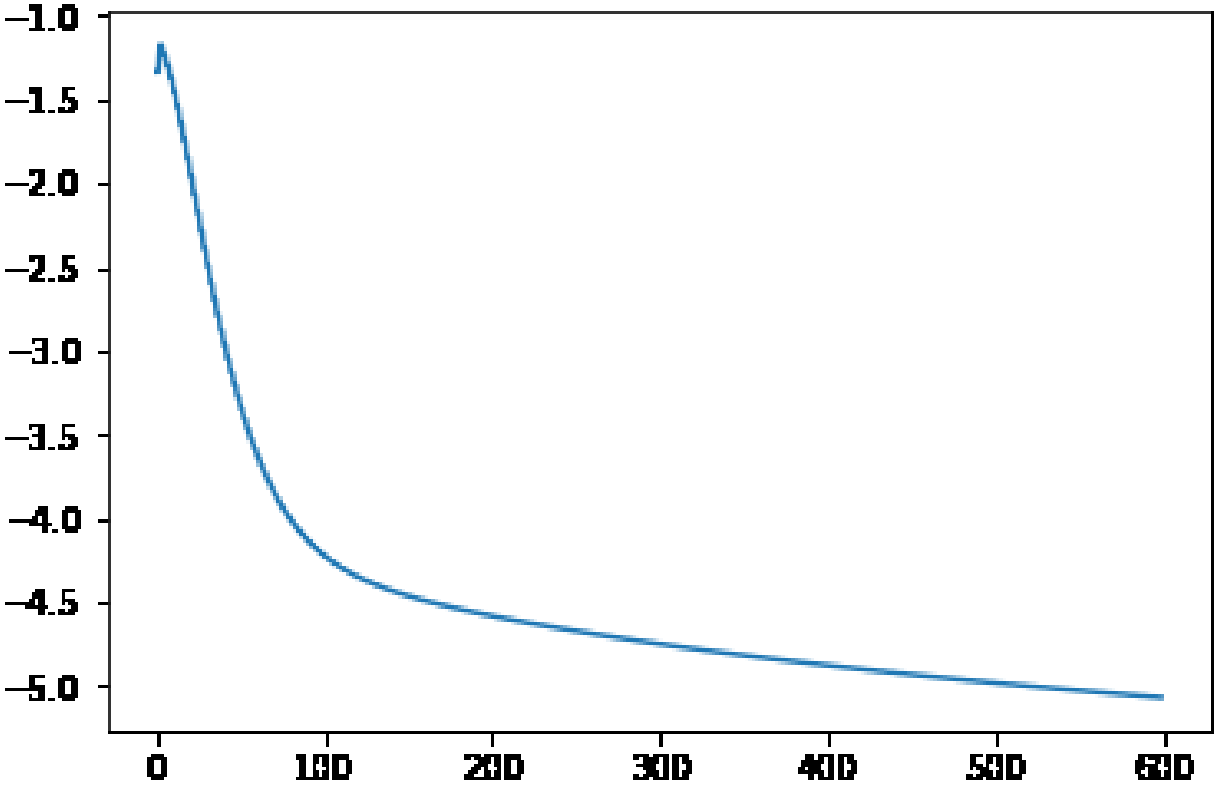
Функция потерь для логистической регрессии

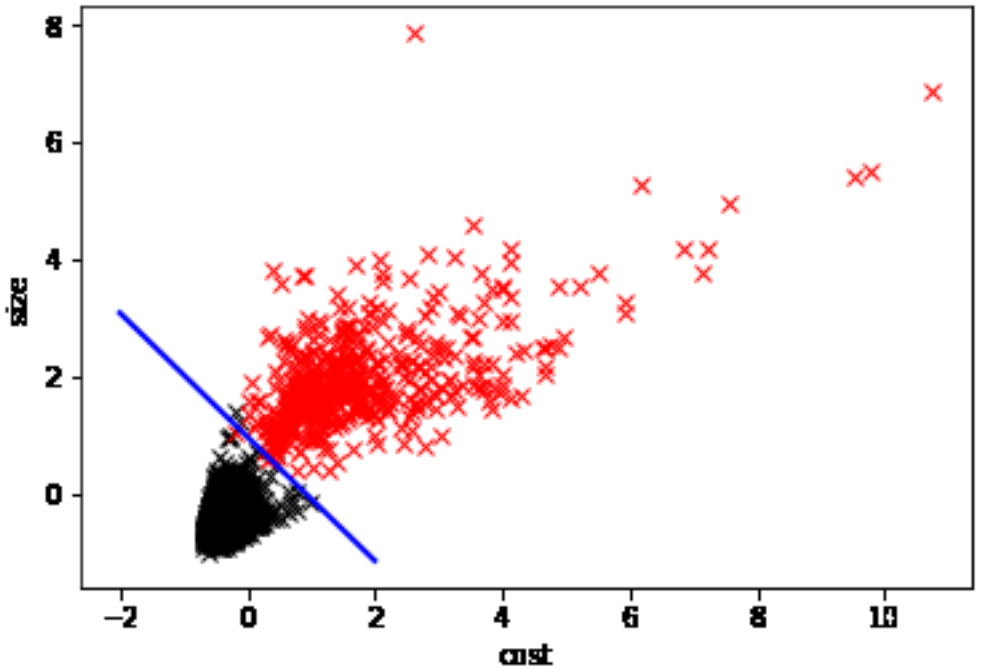
- $J(w) = \frac{1}{2N}\sum_{i=1}^{N}(\sigma(w^T x^{(i)}) - y^{(i)})^2$
- Т.к. добавляется функция сигмоид функция J(w) перестает быть выпуклой
- Нам нужна выпуклая функция потерь, которая обладает следующими свойствами $J(h_w(x), y) \to 0$:
  - Если $\sigma\left(w^T x^{(i)}\right) \to 1$ при $y^{(i)} = 1$
  - Если $\sigma\left(w^T x^{(i)}\right) \to 0$ при $y^{(i)} = 0$
- $J(h_w(x), y) = \begin{cases} -\log(h_w(x)) & \text{если } y = 1 \\ -\log(1 - h_w(x)) & \text{если } y = 0 \end{cases}$
- $J(h_w(x), y) = -y \log\left(h_w(x)\right) - (1 - y) \log(1 - h_w(x))$

Градиентный спуск для логистической регрессии

- Функция потерь:
  - $J(h_w(x), y) = -\frac{1}{N}\sum y^{(i)} \log\left(h_w\left(x^{(i)}\right)\right) + (1 - y^{(i)}) \log(1 - h_w\left(x^{(i)}\right))$
- Хотим найти $\min_w J$
- Частные производные:
  - $\frac{dJ}{dw_j} = \frac{1}{N}\sum(h_w\left(x^{(i)}\right) - y^{(i)})x_j^{(i)}$
- Шаг градиента:
  - $w_j = w_j - lr\frac{1}{N}\sum(h_w\left(x^{(i)}\right) - y^{(i)})x_j^{(i)}$
- Аналогично линейной регрессии

Результат применения градиентного спуска





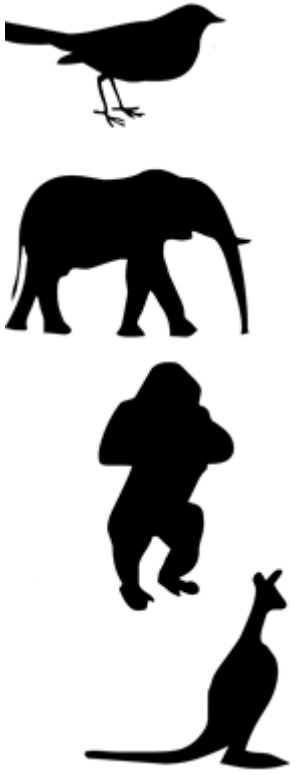
Практика логистическая регрессия

- Реализовать сигмоид
- Реализовать функцию потерь
- Градиенты

Задача множественной классификации

$$Wx = y$$

2.0
0.1
0.2
1.0



Если $x = [1, x_1, \dots, x_n]$

y - вектор размером 4

Что такое W?

W-матрица размером $4 \times n + 1$

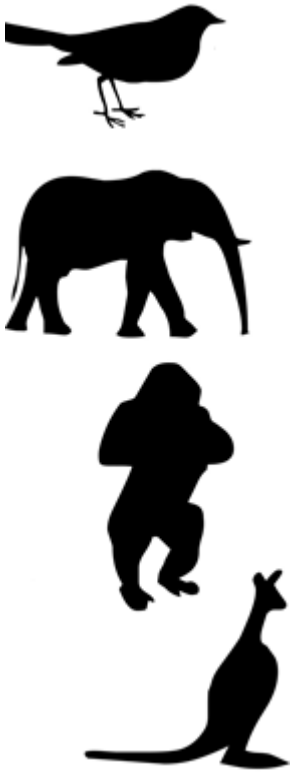
Иногда пишут $Wx + b = y$

Задача множественной классификации

$$Wx = y$$

Надо превратить выход линейной модели в вероятности!

$$2.0 \rightarrow p(y = bird|x, W) = 0.59$$
$$0.1 \rightarrow p(y = elef|x, W) = 0.08$$
$$0.2 \rightarrow p(y = kon|x, W) = 0.09$$
$$1.0 \rightarrow p(y = ken|x, W) = 0.21$$



Softmax

$$y \quad \begin{matrix} 2.0 \rightarrow \\ 0.1 \rightarrow \\ 0.2 \rightarrow \\ 1.0 \rightarrow \end{matrix}$$

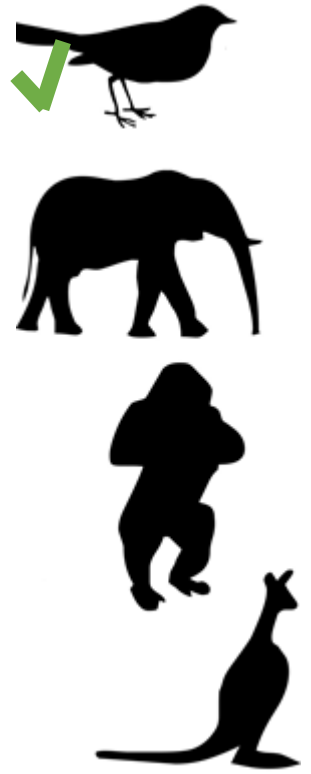
$$S(y_j) = \frac{e^{y_j}}{\sum_j e^{y_j}}$$

$$p(y = bird|x, W) = 0.59$$
$$p(y = elef|x, W) = 0.08$$
$$p(y = kon|x, W) = 0.09$$
$$p(y = ken|x, W) = 0.21$$

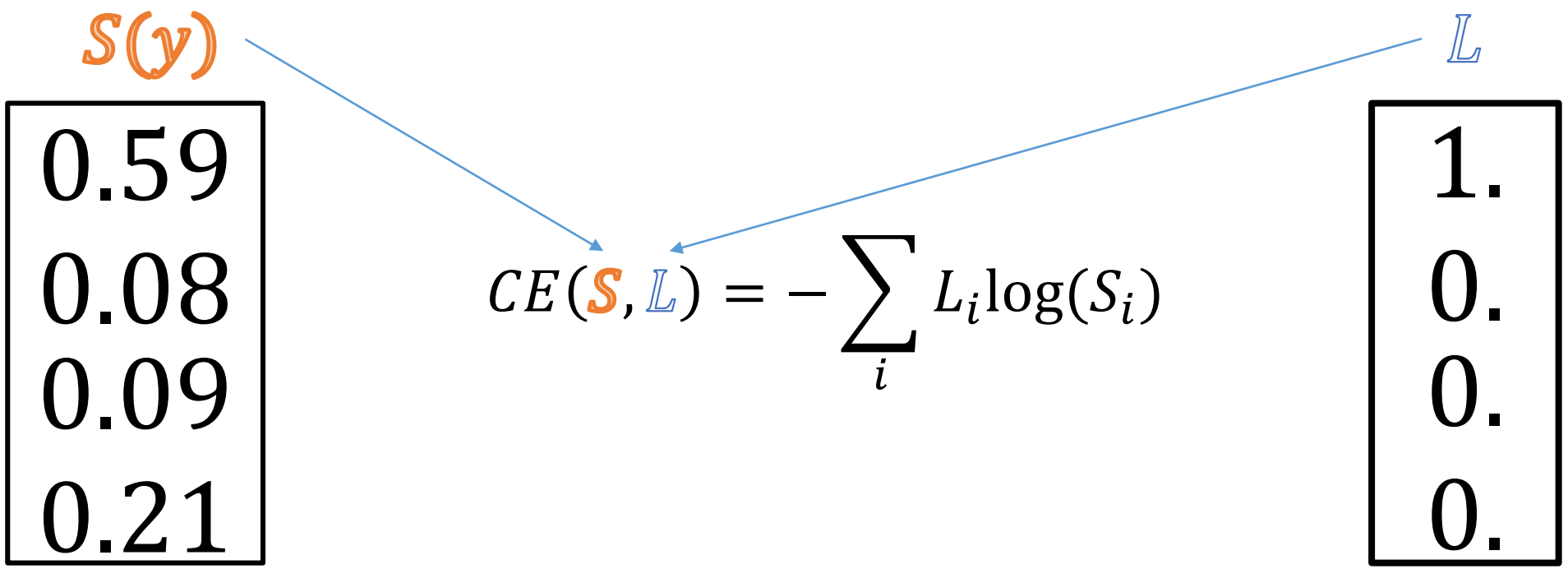
“One-hot” кодировка

0.59
0.08
0.09
0.21

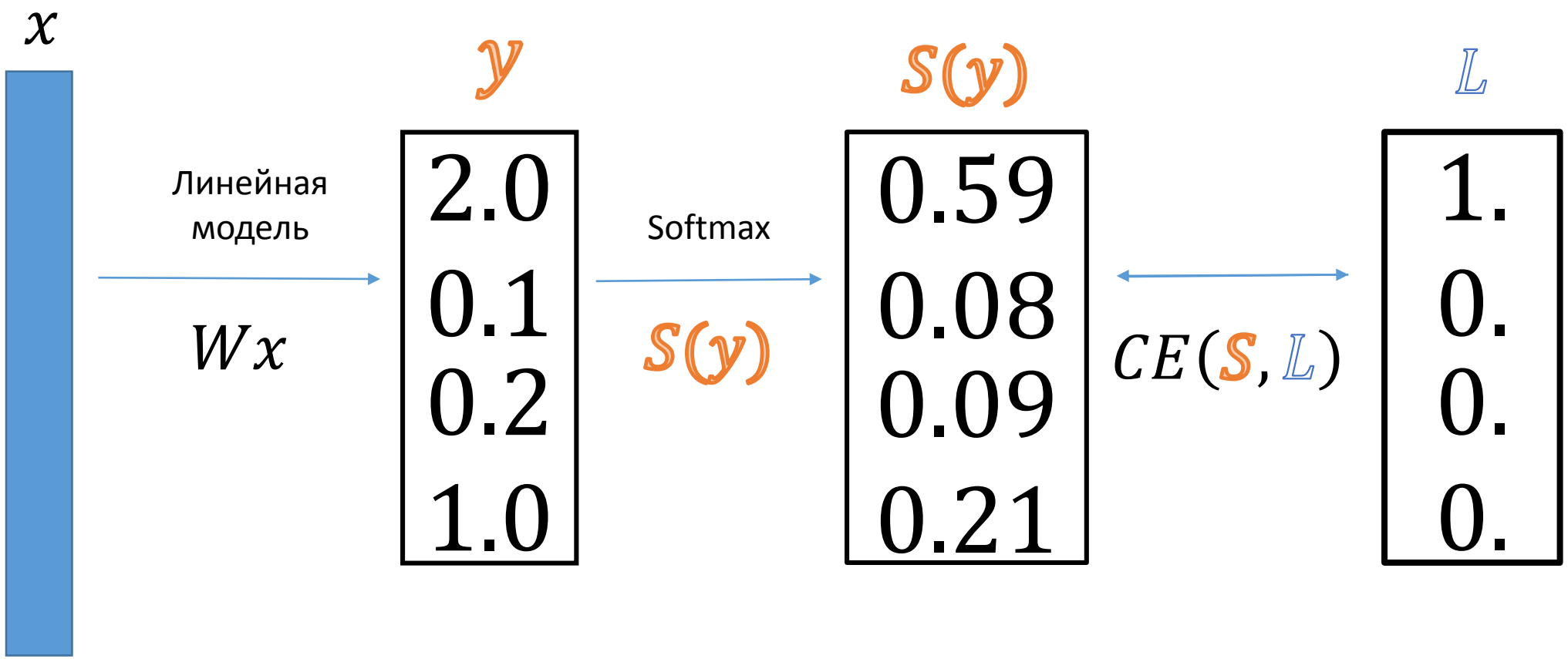
1.
0.
0.
0.



Кросс энтропия



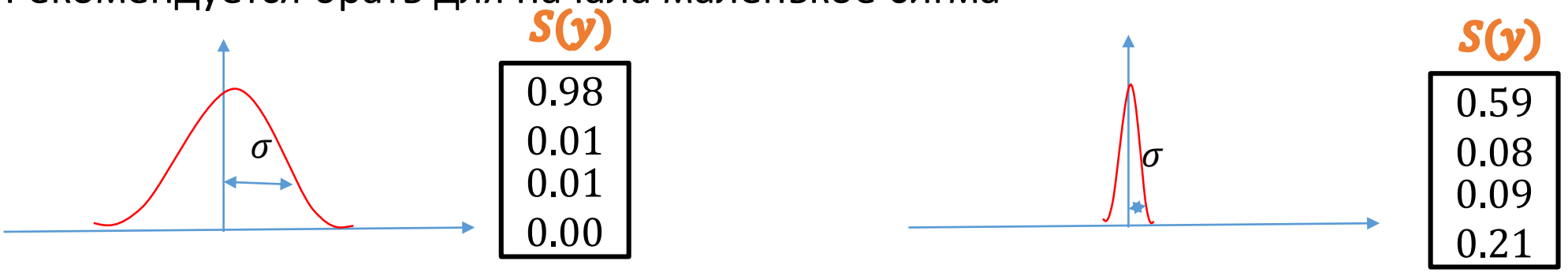
Все вместе



Ориентированный граф вычислений. Вершины – данные, дуги – операции.

Инициализация параметров

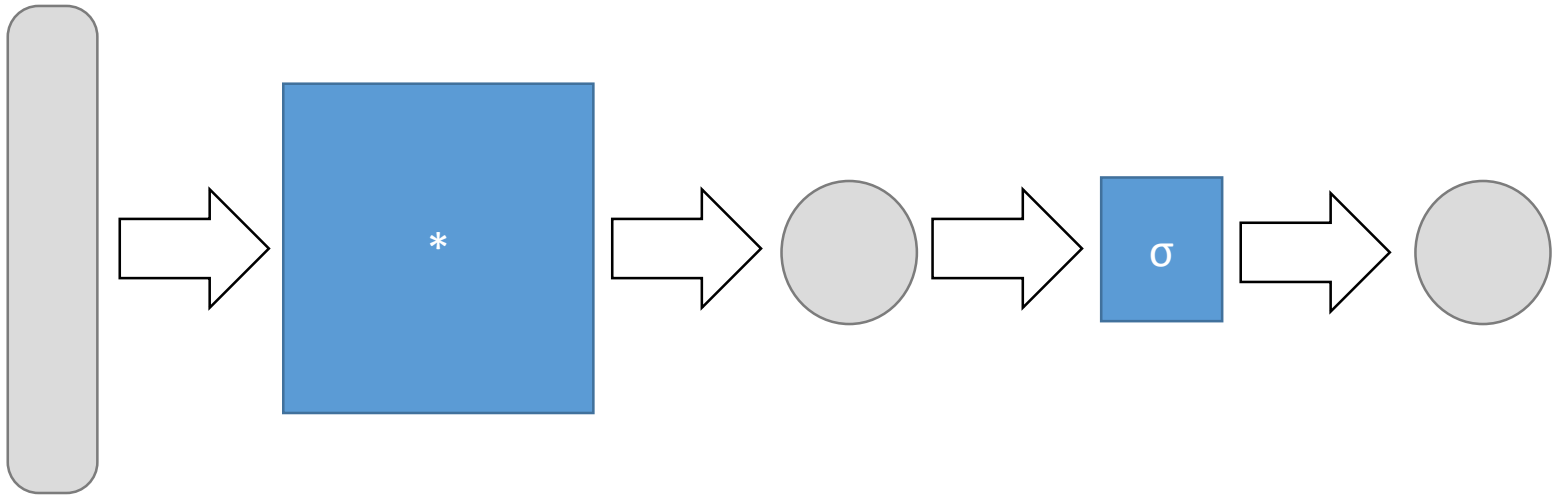
- Как правильно инициализировать W?
- Т.е. на выход нашего классификатора Softmax – начальное распределение сильно чувствительно к входным значений
- При условии, что наши данные имеют средине значение 0 и единичную дисперсию, мы можем инициализировать W нормальным распределением
- Большое σ значения на выходе будут сильно различаться, будут сильные пики в softmax
- Маленькое сигма – распределение будет более равномерным.
- Рекомендуется брать для начала маленькое сигма



Вычислительный граф

$$P(h_w(x) = 1|x, W) = \frac{1}{1 + e^{-Wx}}$$

Граф/сеть

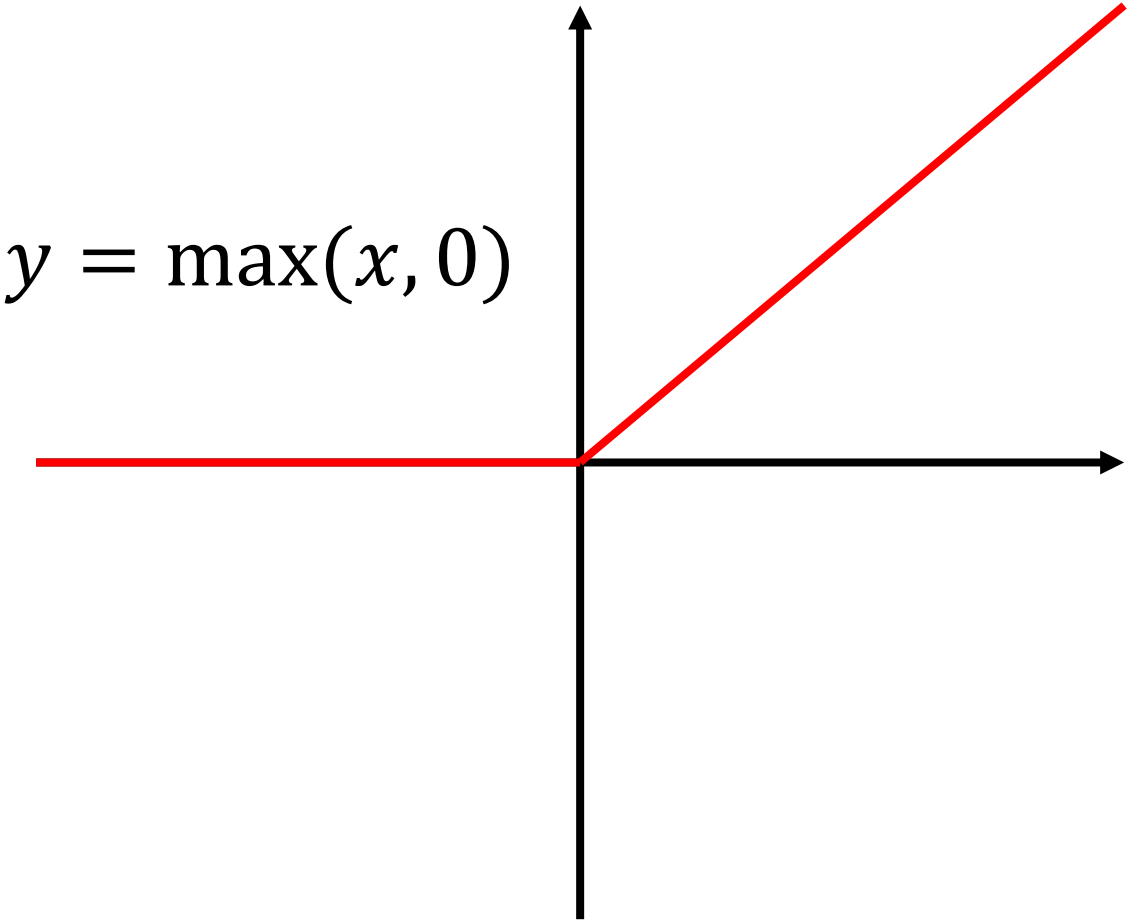


Свойства линейных моделей

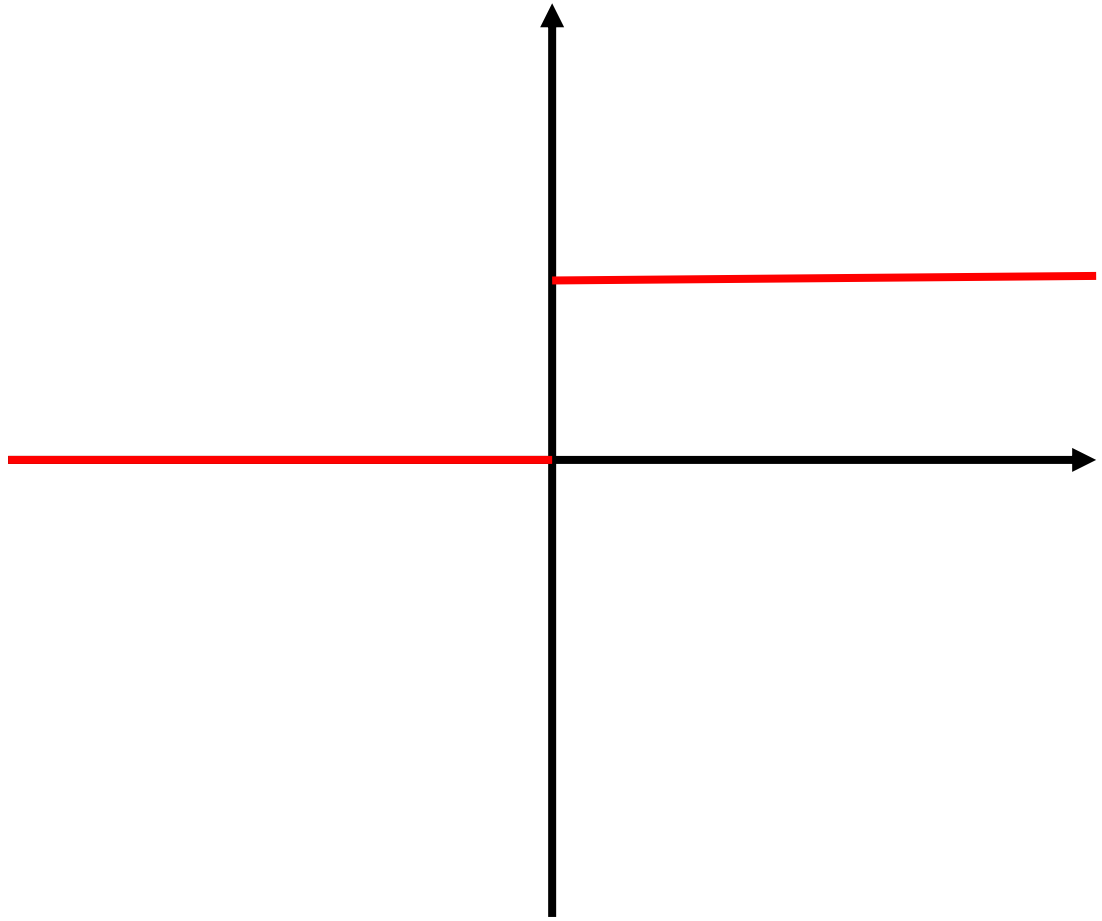
- Плюсы
  - Выпуклые функции потерь
  - Стабильность
  - $y = Wx \rightarrow y + \Delta \approx W(x + \Delta)$
  - Производные константы
  - $\frac{dy}{dx} = W^T$
  - $\frac{dy}{dw} = x^T$
- Минусы
  - Ограниченно число параметров
  - $k \times n + 1$
  - Невозможно моделировать сложные функции
  - $x1 * x2$
  - $y = W_1 W_2 W_3 x = Wx$

Rectified linear units (ReLU)

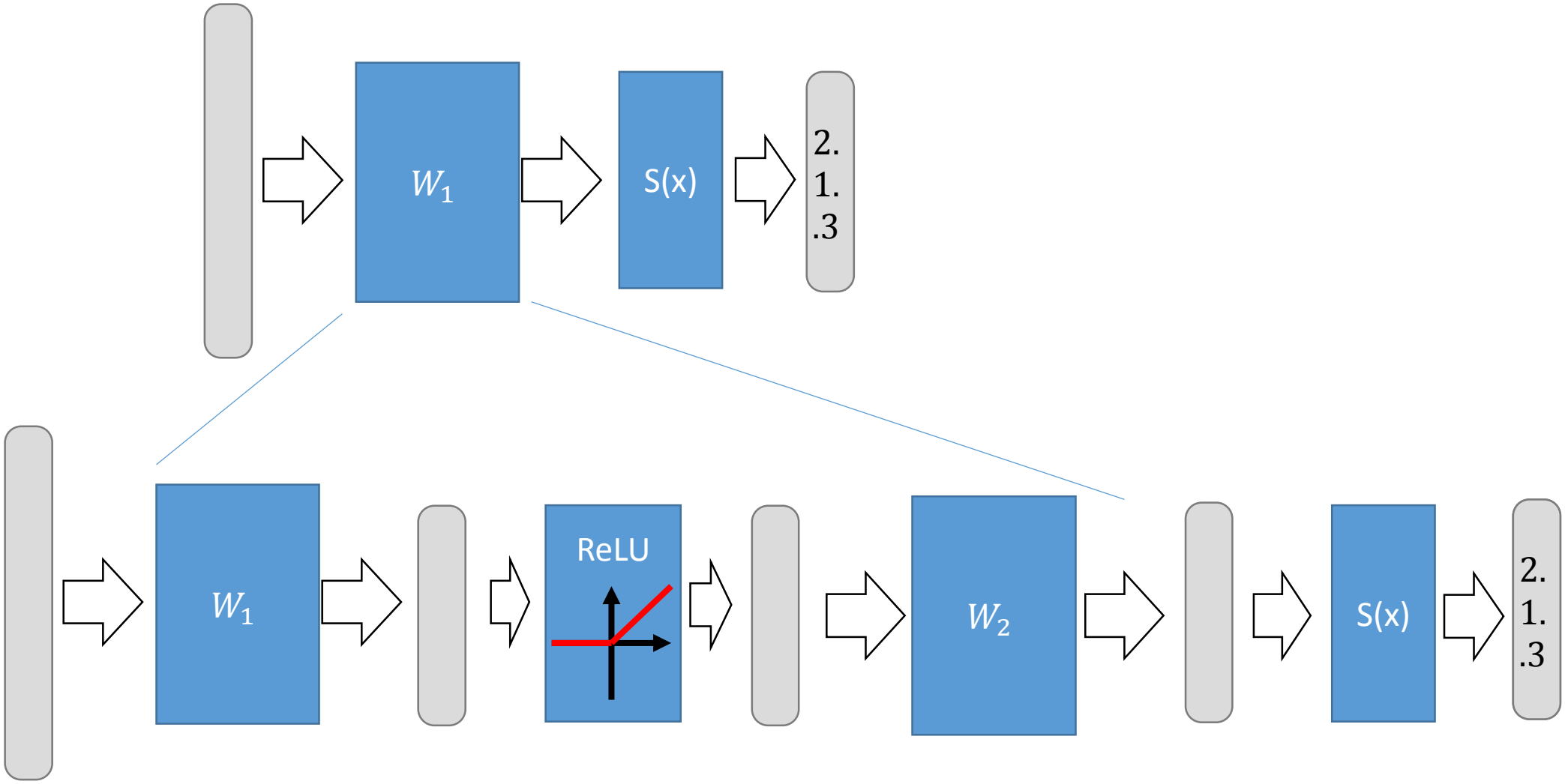
$$y = \max(x, 0)$$



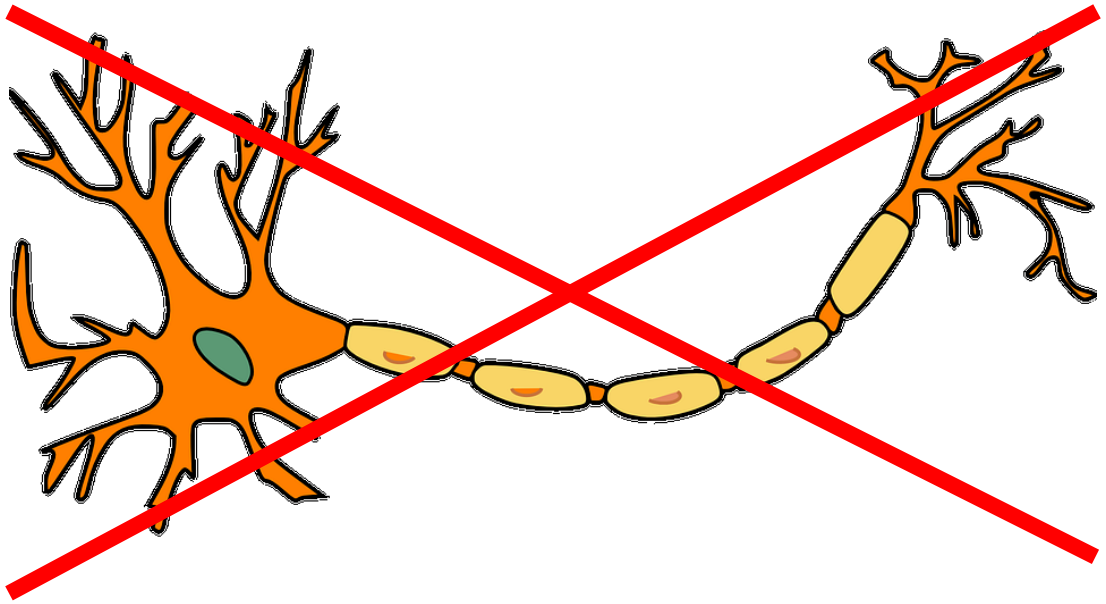
Производная?



Нейронная сеть



Биологический нейрон



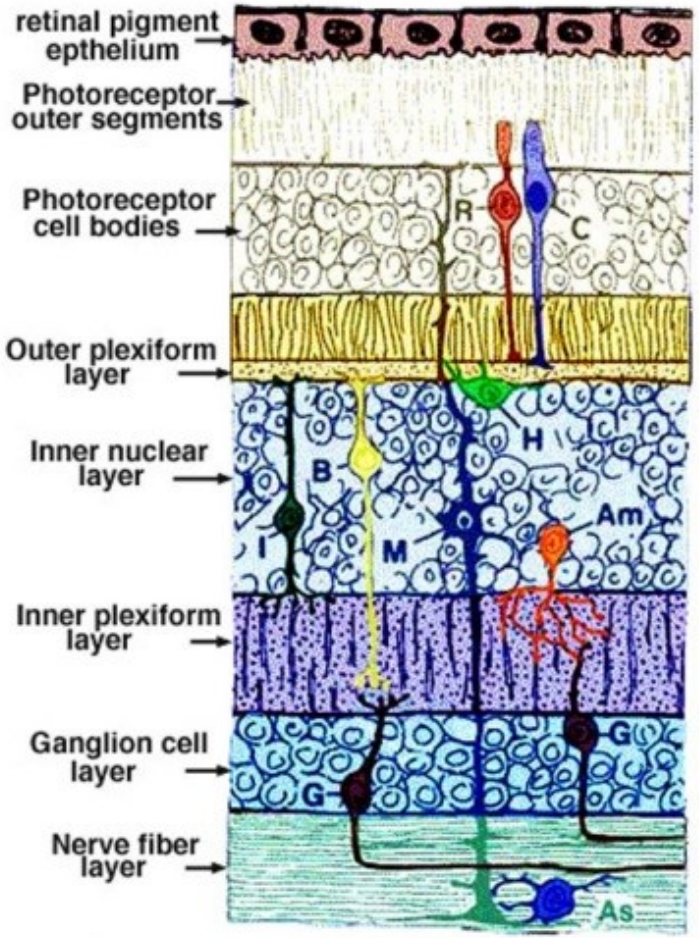
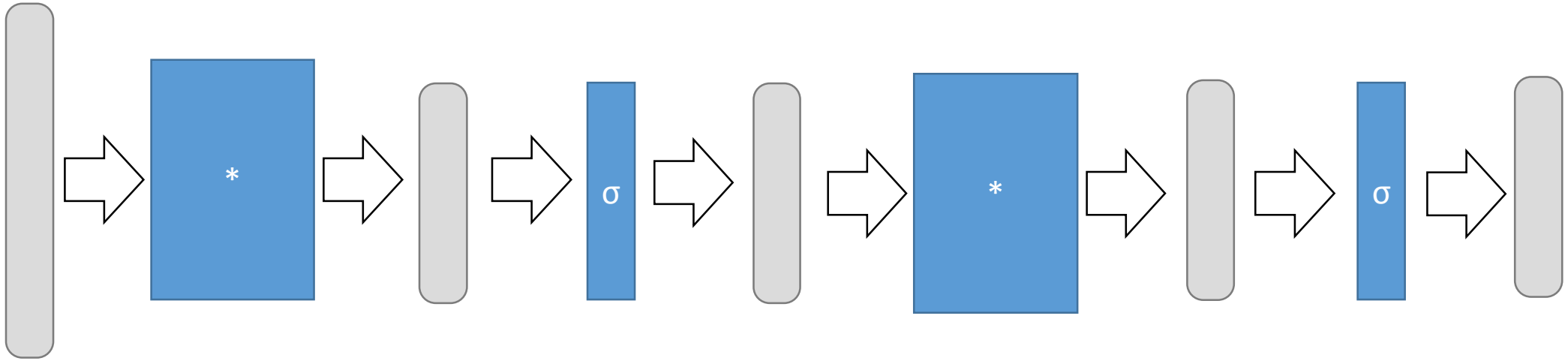


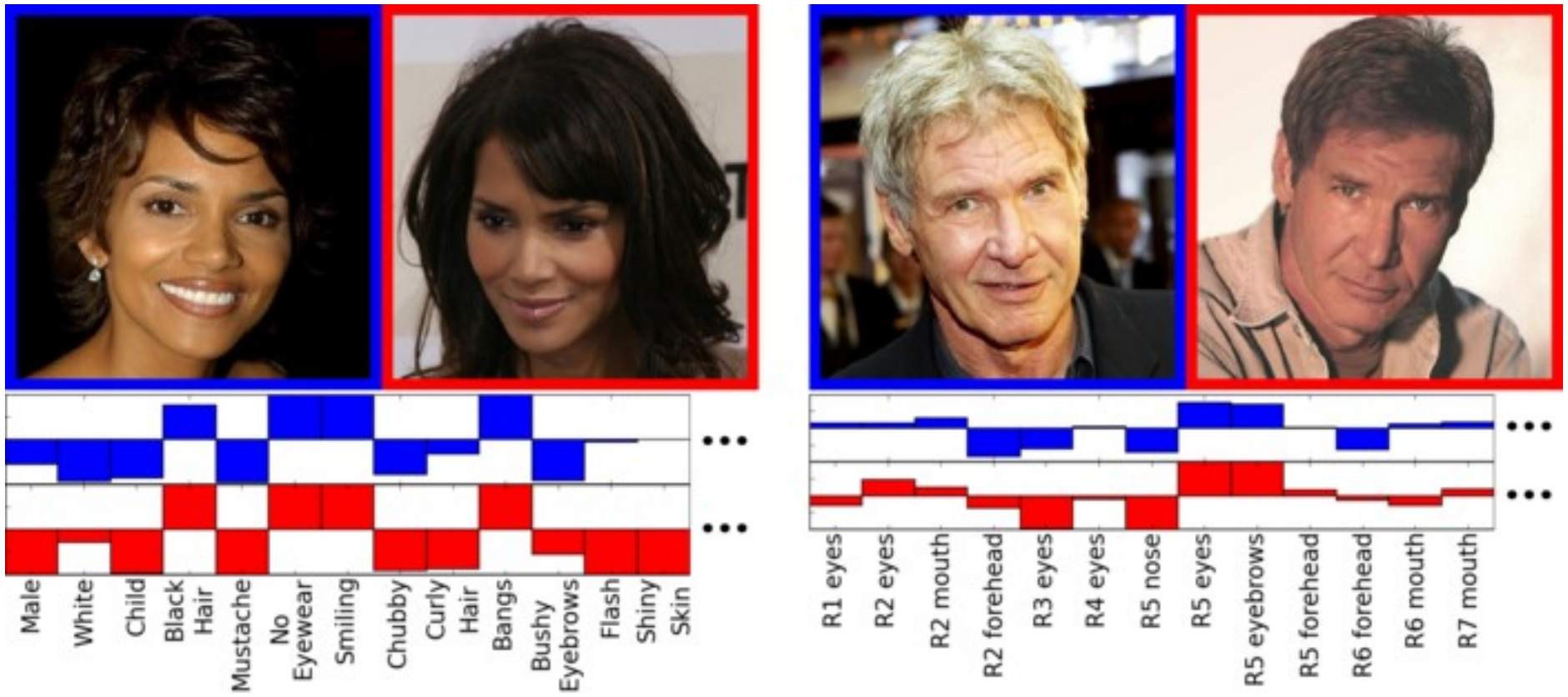
Fig. 5. Scheme of the layers of the developing retina around 5 months' gestation (Modified from Odgen,1989).

Многоуровневый вычислительный граф



- Первый уровень: параллельная логистическая регрессия
- Результат первого уровня используется в качестве признаков для второго уровня
- Второй уровень:

Выход классификатора как вектор признаков



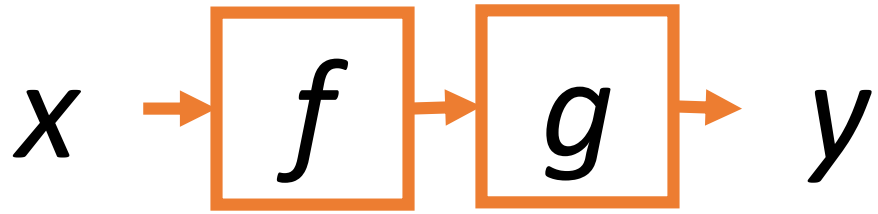
[Kumar et al. **Attribute and Simile Classifiers for Face Verification. ICCV 2009]**

Производная сложной функции (chain rule)

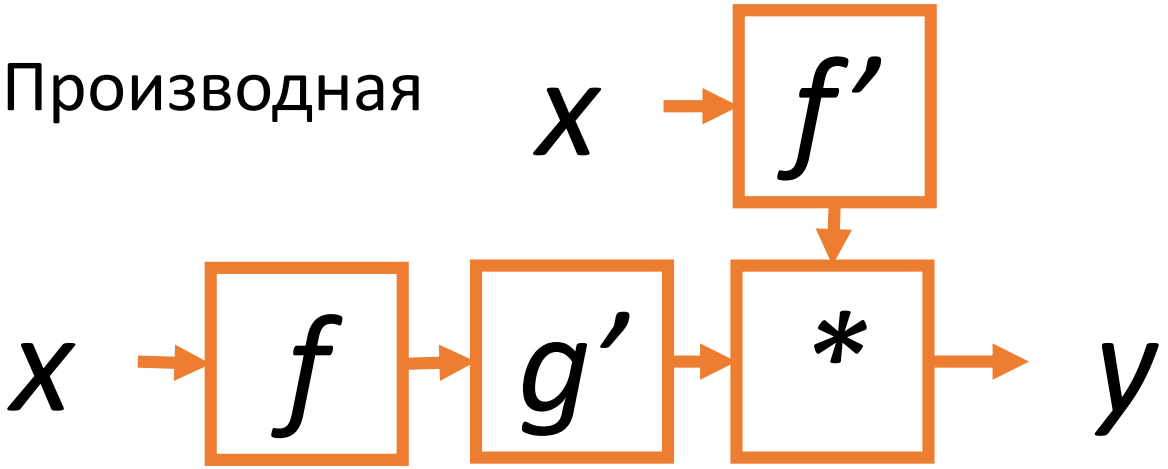
- $\frac{d(xy)}{dx} = y$
- $\frac{d(x+y)}{dx} = 1$
- $\frac{df(g(x))}{dx} = \frac{df}{dg}\frac{dg}{dx}$
- $\frac{df(g(x),m(x))}{dx} = \frac{df}{dg}\frac{dg}{dx} + \frac{df}{dm}\frac{dm}{dx}$

Производная сложной функции (chain rule)

Функция



Производная

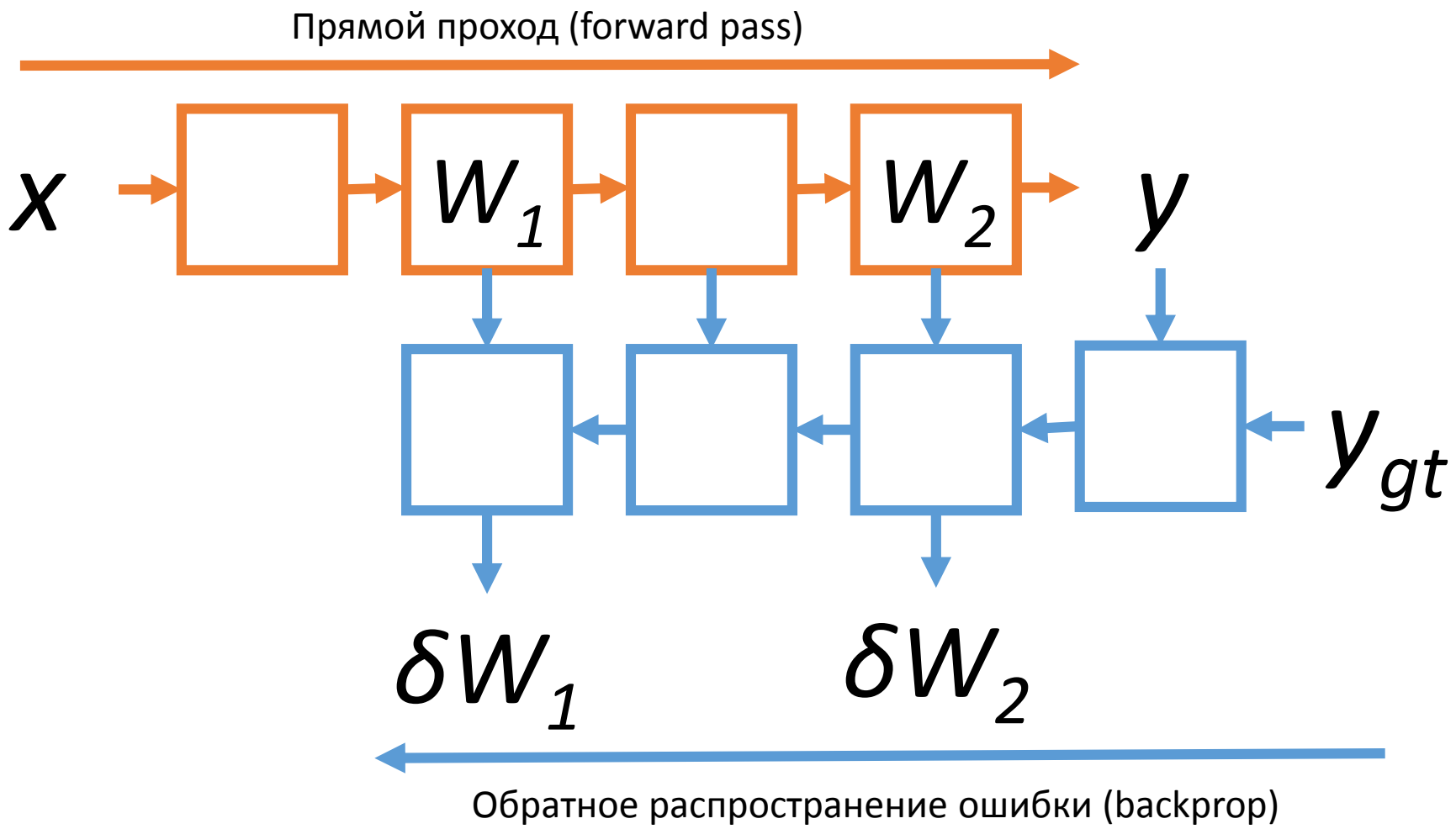


- Автоматически строится большинством библиотек машинного обучения
- Эффективное использование ресурсов
- Много повторных использований

Градиенты абстрактного слоя

- Прямой проход:
- $y = f(x, w)$
- Обратный проход:
- $z(x) = z\big(f(x, w)\big)$
- $\frac{dz}{dx} = \frac{dy^T}{dx}\frac{dz}{dy}$
- $\frac{dz}{dw} = \frac{dy^T}{dw}\frac{dz}{dy}$

Обратное распространение ошибки



Градиенты мультипликативного слоя

$$z(x) = z\big(f(x, w)\big) = z(Wx)$$

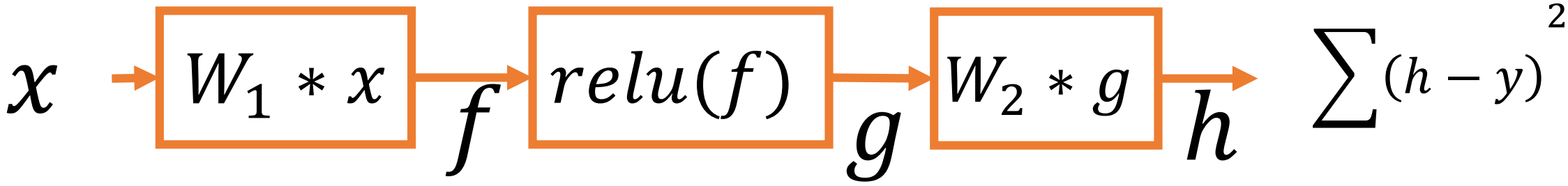
$$\frac{dz}{dx} = \frac{dy^T}{dx}\frac{dz}{dy}$$

- $y = Wx$
- $\frac{dy^T}{dx} = W^T$
- $\frac{dz}{dx} = W^T \frac{dz}{dy}$

$$\frac{dz}{dw} = \frac{dy^T}{dw}\frac{dz}{dy}$$

- $\frac{dz}{dw_{ij}} = \left(\frac{dy}{dw_{ij}}\right)^T \frac{dz}{dy} = x_j \frac{dz}{dy_i}$
- $\frac{dz}{dw} = \frac{dz}{dy} x^T$

Пример для нейронной сети



- $L\left(W_2\left(ReLU(W_1 x)\right)\right) = L(h\left(g\left(f(x)\right)\right))$
- $\frac{dL}{dW_2} = \frac{dL}{dh}\frac{dh}{dW_2}$
- $\frac{dL}{dW_1} = \frac{dL}{dh}\frac{dh}{dg}\frac{dg}{df}\frac{df}{dW_1}$

- X = [64,1000]
- W1 = [1000,100]
- W2 = [100,10]
- Y = [64,10]

Пример для нейронной сети

- $\frac{dL}{dW_2} = \frac{dL}{dh}\frac{dh}{dg}\frac{dg}{df}\frac{df}{dW_1}$
- $\frac{dL}{dh} = 2(h - y)$
- [64,10]
- $\frac{dh}{dg} = W_2$
- [100,10]
- $\frac{dL}{dh}\frac{dh}{dg} = 2(h - y) * W_2^T$
- [64,100]

- X = [64,1000]
- W1 = [1000,100]
- W2 = [100,10]
- Y = [64,10]

Пример для нейронной сети

- $\frac{dL}{dW_1} = \frac{dL}{dh}\frac{dh}{dW_2}$
- $\frac{dL}{dh} = 2(h - y)$
- [64,10]
- $\frac{dh}{dW_2} = g$
- [64,100]
- $\frac{dL}{dW_2} = g^T 2(h - y)$
- [100,10]

- X = [64,100]
- W1 = [100,1000]
- W2 = [100,10]
- Y = [64,10]

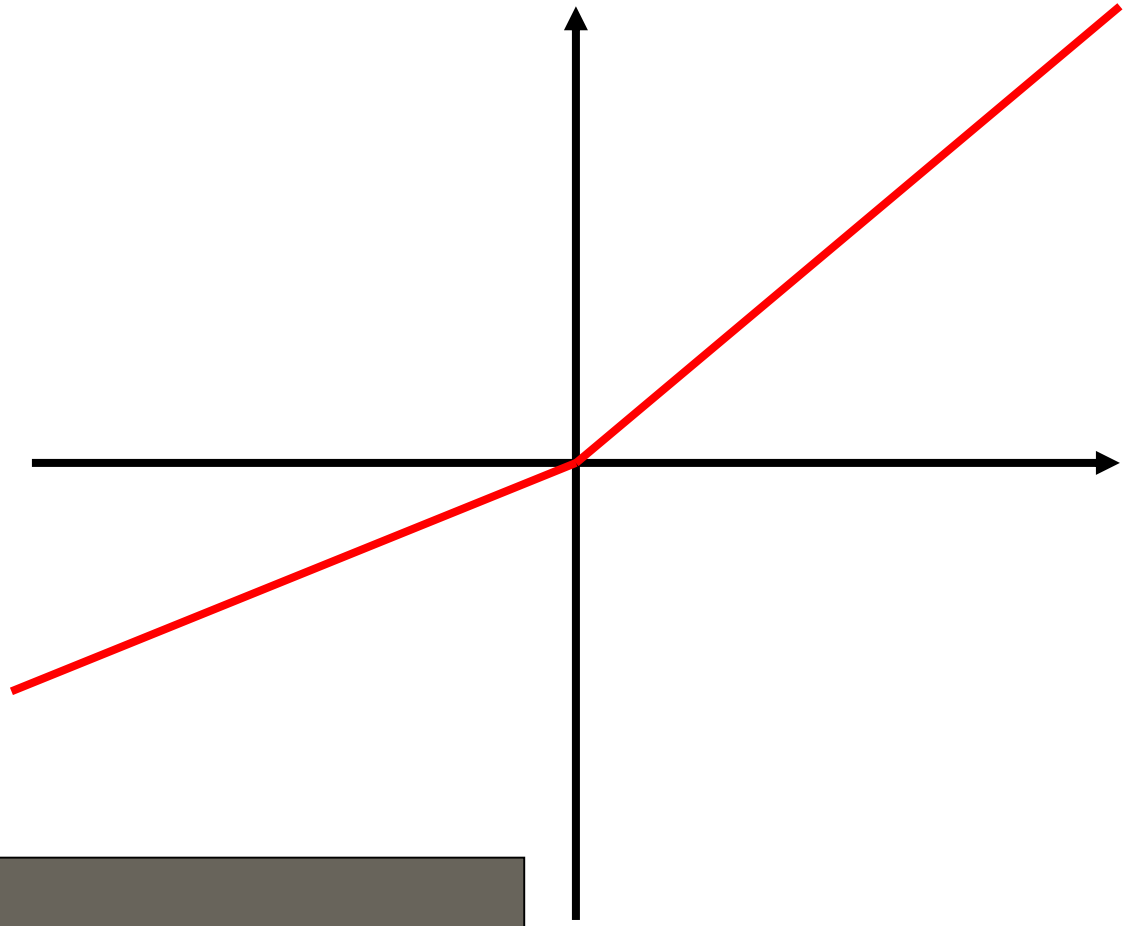
Программная реализация модуля

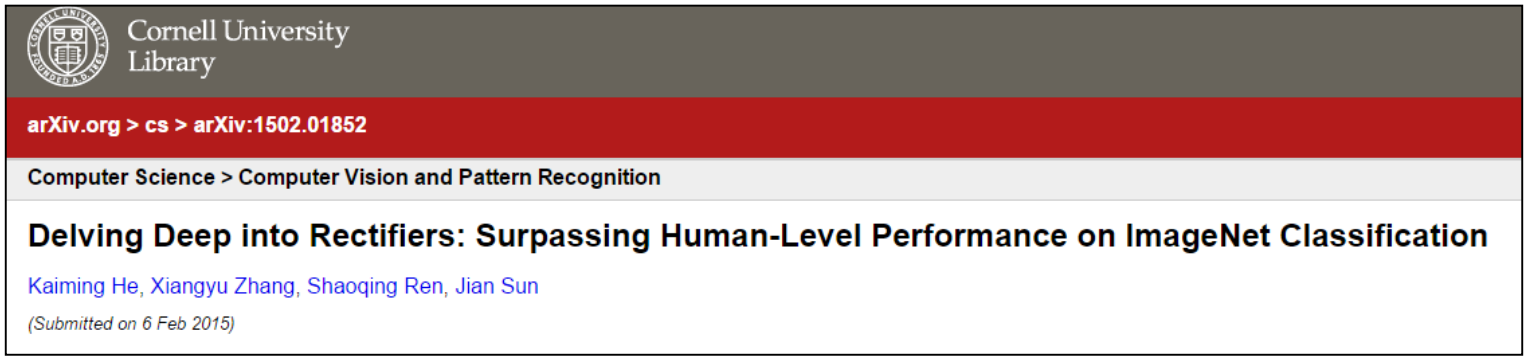
```
import numpy as np
class Layer:
    def __init__(self):
        pass

    def forward(self, x):
        return x

    def backward(self, x, grad_output):
        num_units = input.shape[1]
        iden = np.eye(num_units)
        return np.dot(grad_output, iden )
```

- Две функции
  - Прямой проход
  - Обратный проход
- Во время обратного прохода:
  - вычисляются производные от forward по x
  - скалярно умножаются на градиенты от следующей функции

Leaky ReLU

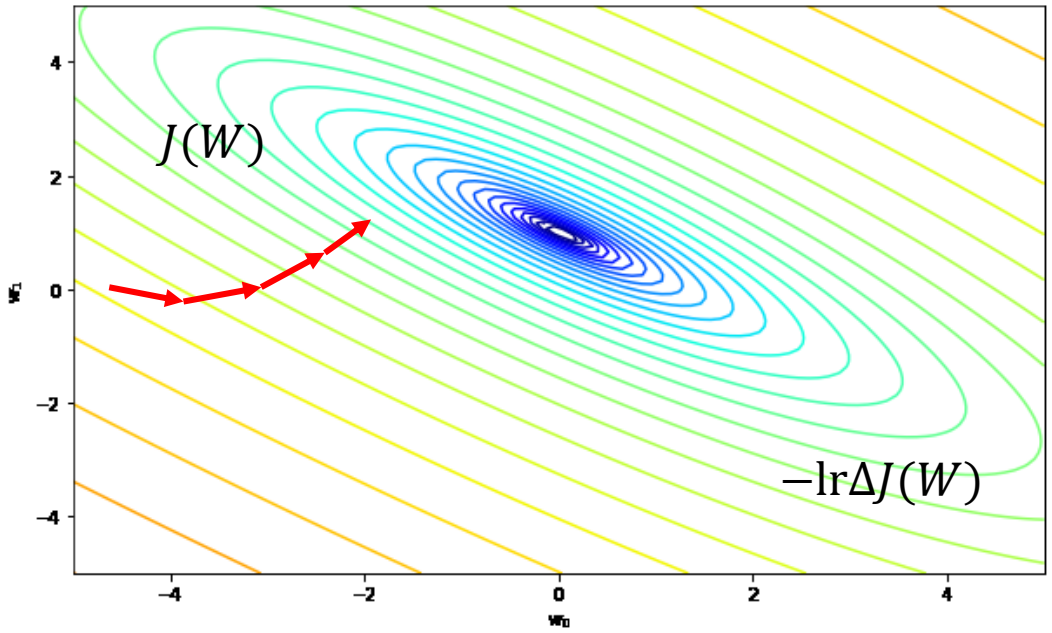
- $y = \max(x, \alpha x)$
- $\frac{dy}{dx} = ?$
- $\frac{dy}{d\alpha} = ?$





Стохастическая оптимизация

- Проблема: данных настолько много, что они не помещаются в оперативную память для градиентного спуска
  - Текущие базы данных изображений (Cityscapes, ImageNet, MS COCO) содержат тысячи изображений

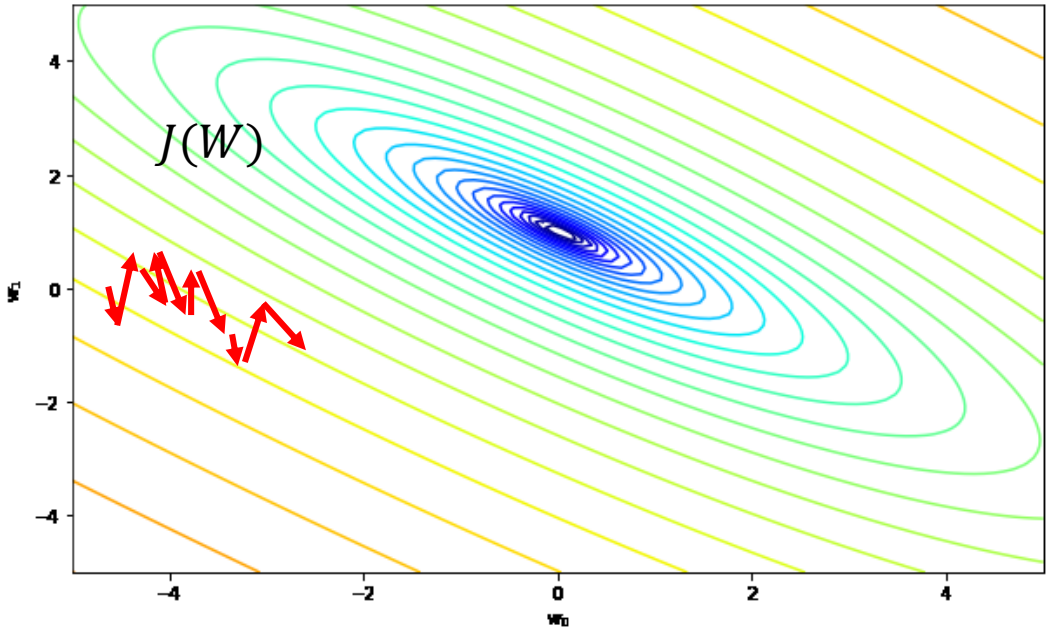


$J(W) = \sum D(x^{(i)})$
Сложность O(N)

$\Delta J(W)$
Сложность $\sim 3 \times O(N)$

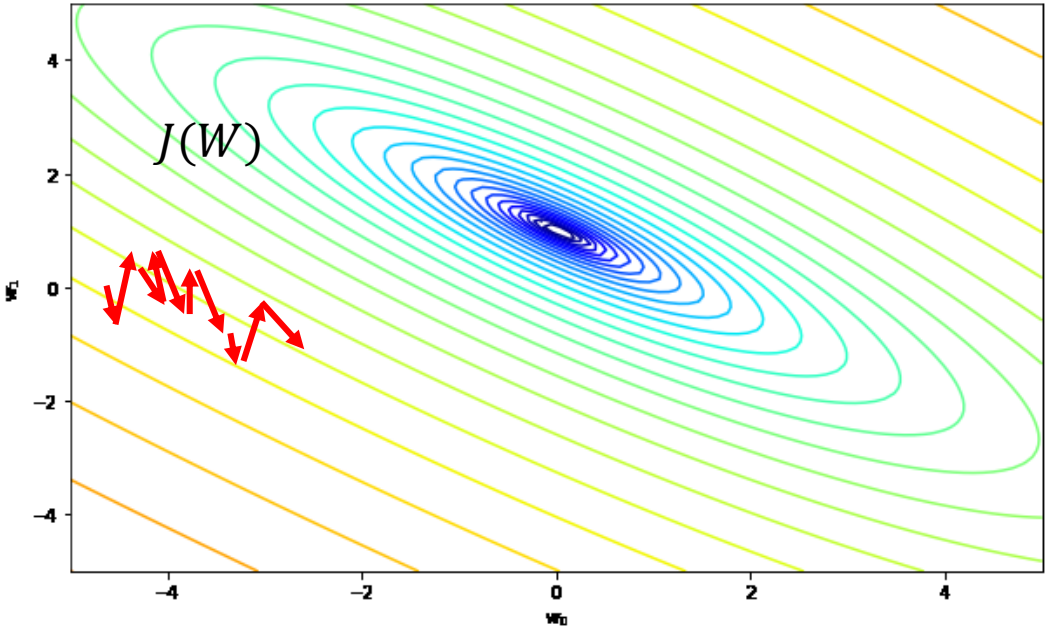
Стохастическая оптимизация (SGD)

- Вместо того, чтобы проходиться по всему обучающему множеству из N несколько раз с большим lr
- Выбираем случайное подмножество данных M<<N и делаем маленькую итерацию (lr - маленькое) на нем



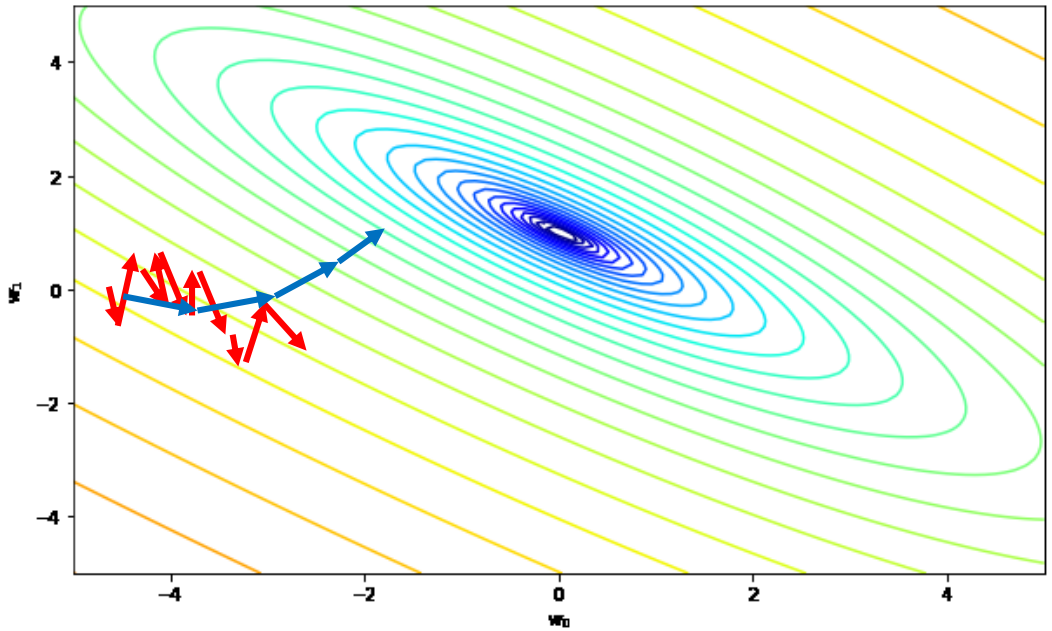
Проблемы SGD

- Сильно чувствительный к шуму
- Нужно:
  - Нормированный входной вектор (среднее 0, дисперсия 1)
  - Нормированные случайные веса W



Momentum

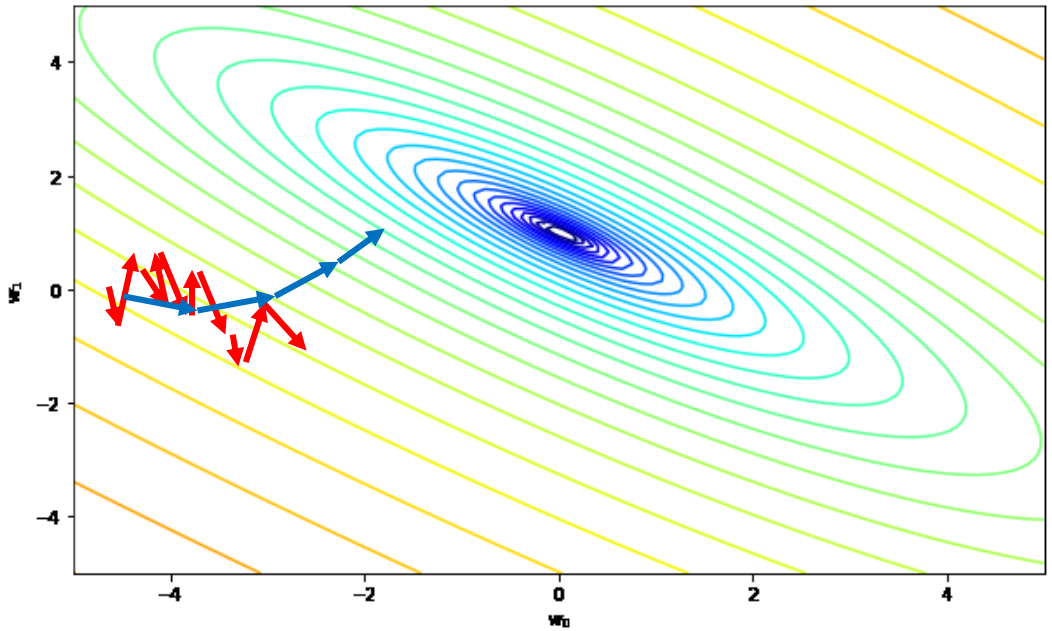
$$J(W) = \sum D(x^{(i)})$$



- ~~$W = W - lr \times \Delta J$~~
- Сглаживаем значения градиентов бегущим средним
- $\mu = 0.9\mu + \Delta J$
- $W = W - lr \times \mu$

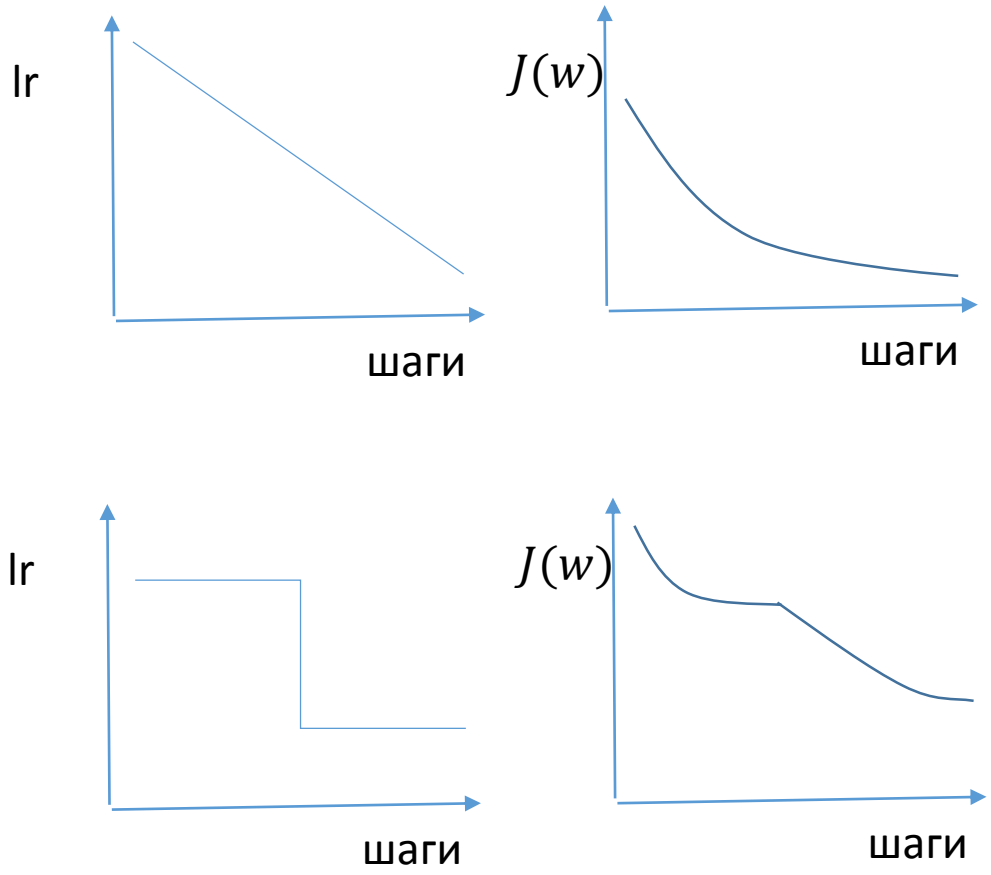
Momentum

$$J(W) = \sum D(x^{(i)})$$



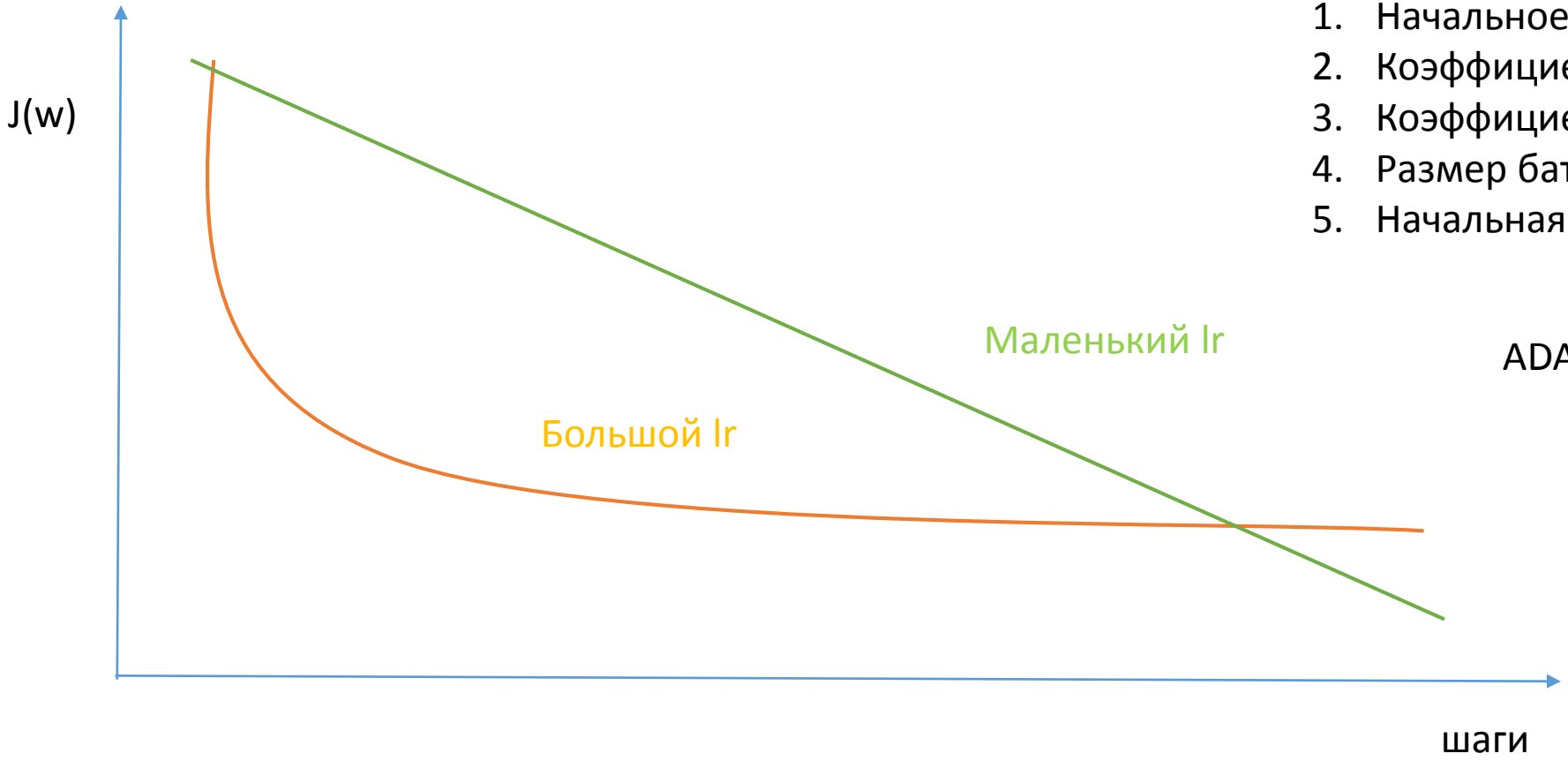
- ~~$W = W - lr \times \Delta J$~~
- Сглаживаем значения градиентов бегущим средним
- $\mu = 0.9\mu + \Delta J$
- $W = W - lr \times \mu$

Понижение скорости обучения



- Стратегия 1:
  - Уменьшать несущественно скорость обучения после каждого шага
  - $lr = lr * 0.99998$
- Стратегия 2:
  - Уменьшать сильно при выходе функции потерь на плато
  - $lr = lr * 0.1$

Тюнинг скорости обучения



Гиперпараметры:

1. Начальное скорость обучения
2. Коэффициент уменьшения скорости
3. Коэффициент момента
4. Размер батча
5. Начальная инициализация весов

ADAM, ADAGRAD

Создание своей глубокой сети

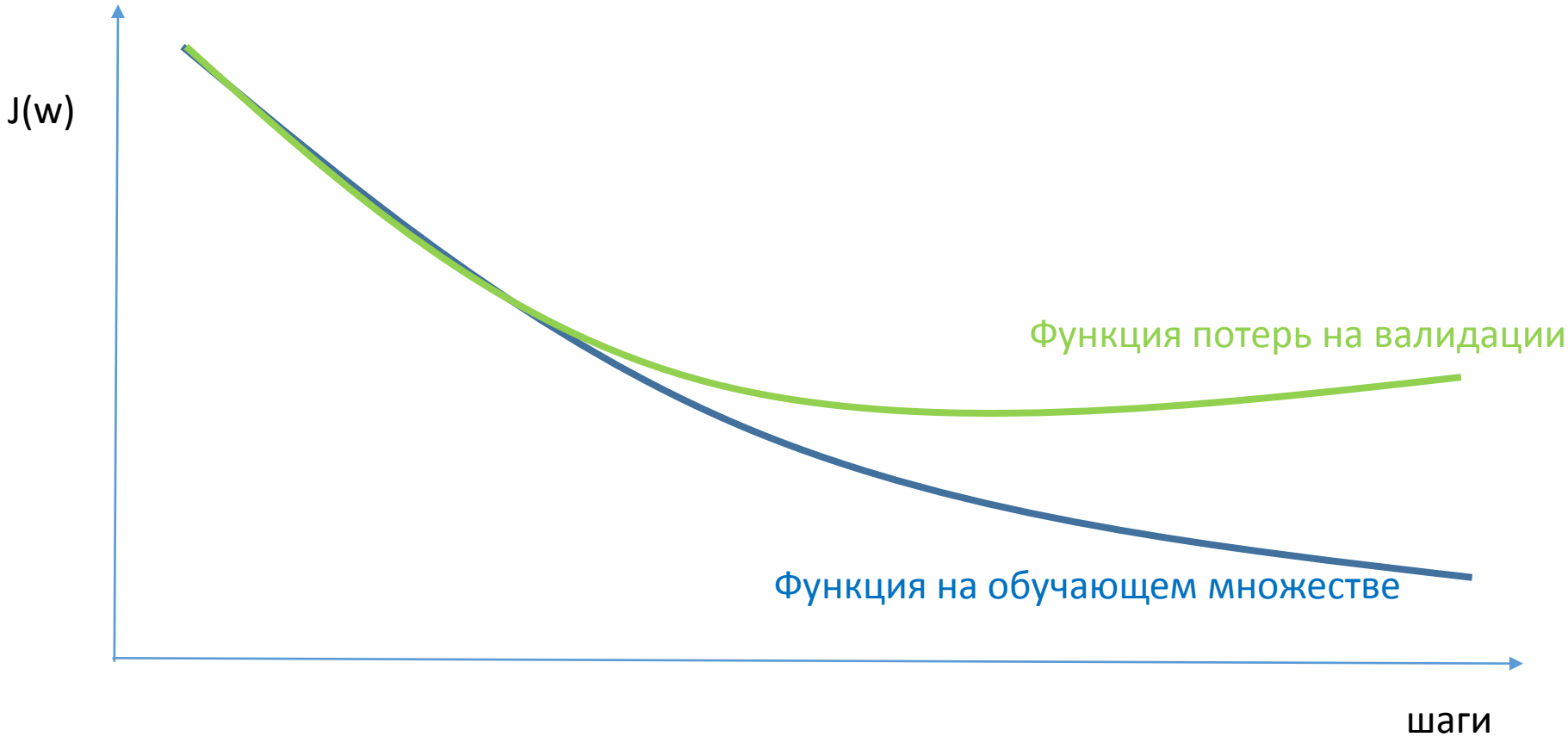
- Задать вычислительные слои
- Собрать слои в вычислительный граф
- Итерировать по случайным подмножествам выборки
- Для каждого подмножества вычисляем градиенты и обновляем параметры

Разбиение данных на несколько групп

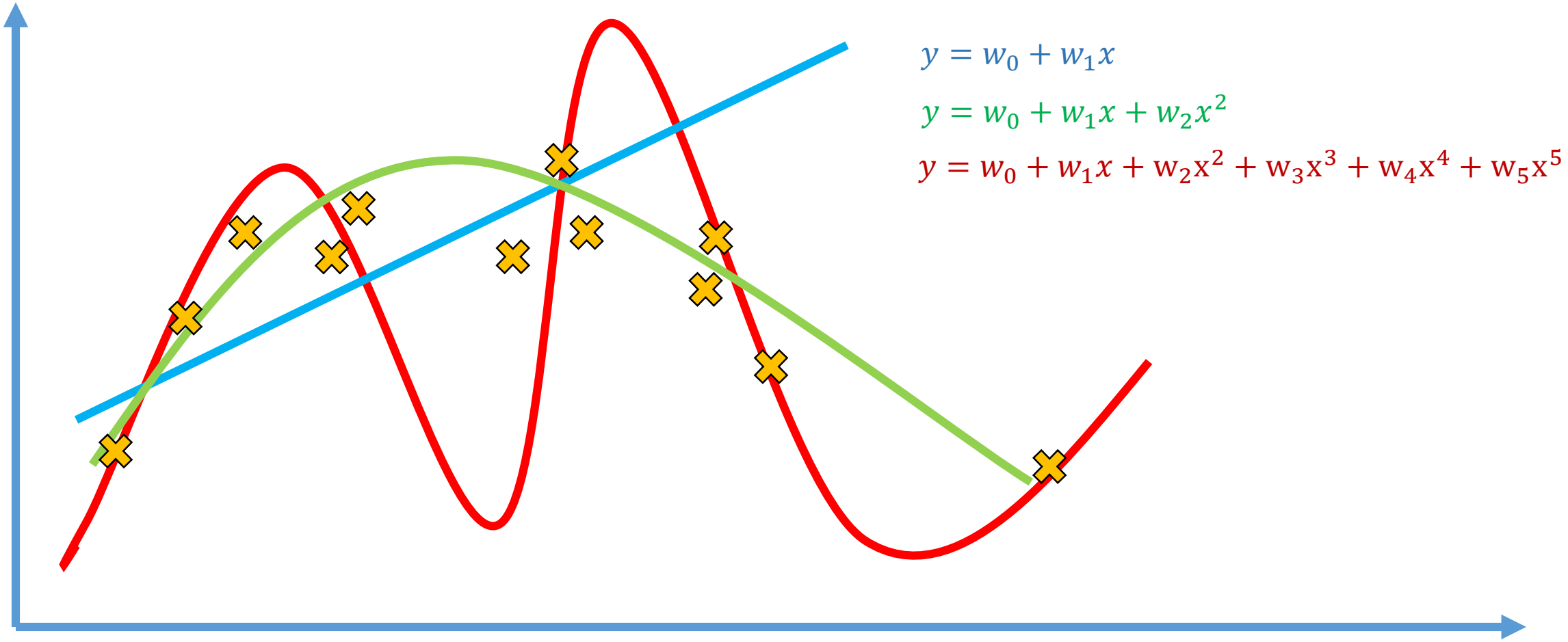
- 80% training
- 10% валидация
- 10% тест

- Используем train для обучения модели
- Валидацию для настройки гиперпараметров
- Тест – для оценки итоговой модели

Переобучение модели



Переобучение полиномиальной регрессии

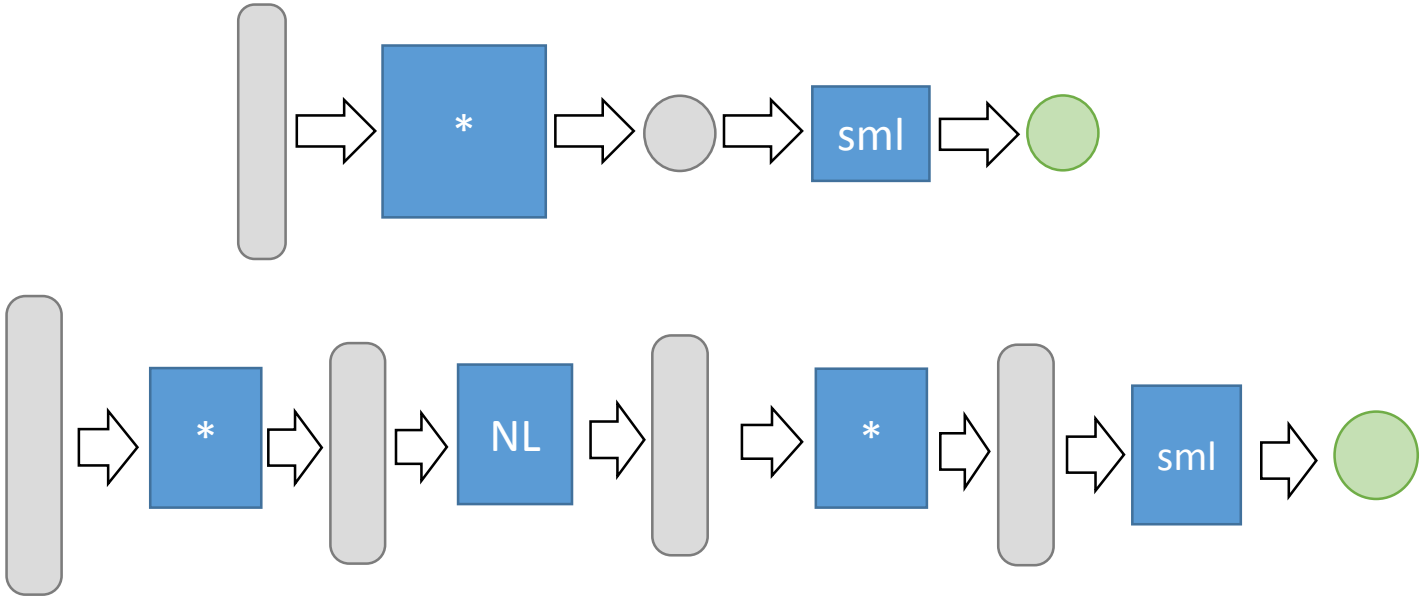


$$y = w_0 + w_1 x$$

$$y = w_0 + w_1 x + w_2 x^2$$

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4 + w_5 x^5$$

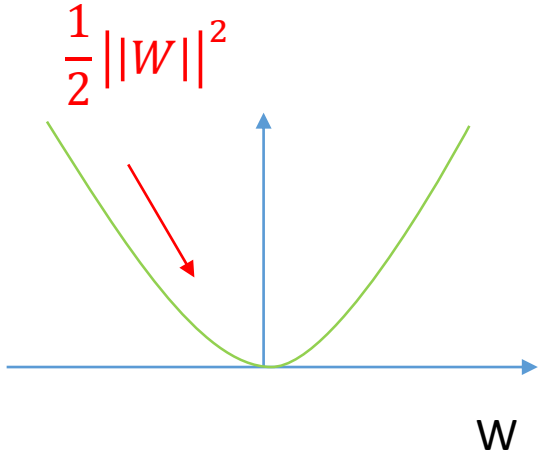
Переобучение модели



- Переобучение – очень ощутимо для глубоких моделей. Почему?
- Прогресс машинного обучения и глубинного обучения ждал появления большого количества данных
- Эффект переобучения: Параметров модели хватает, чтобы запомнить обучающую выборку

L2 регуляризация

- Регуляризация по Тихонову
- $J'(W) = J(W) + \frac{\beta}{2}||W||^2$
- Другие методы регуляризации:
- Взять меньше модель
- Ранняя остановка
- Использовать несколько моделей

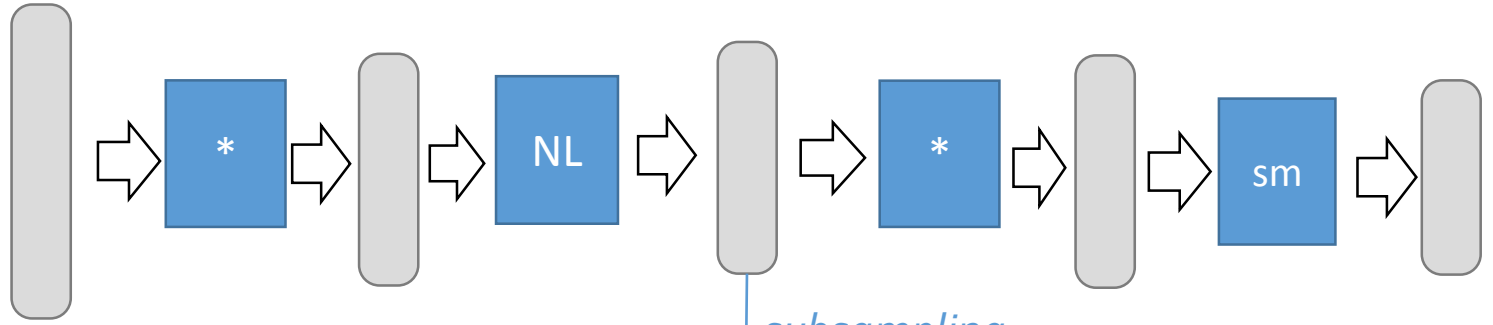


Использование нескольких моделей

- Различные локальные минимумы дают улучшения
- Использования различных архитектур – тоже
- Почти все соревнования по классификации были выиграны набором из нескольких глубоких моделей

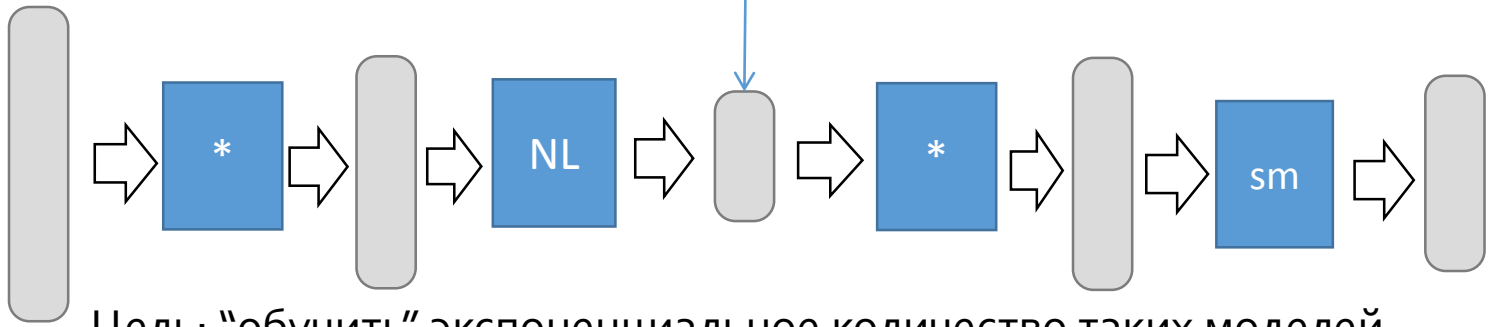
Dropout (выбросы)

Эмуляция работы ансамбля:



subsampling

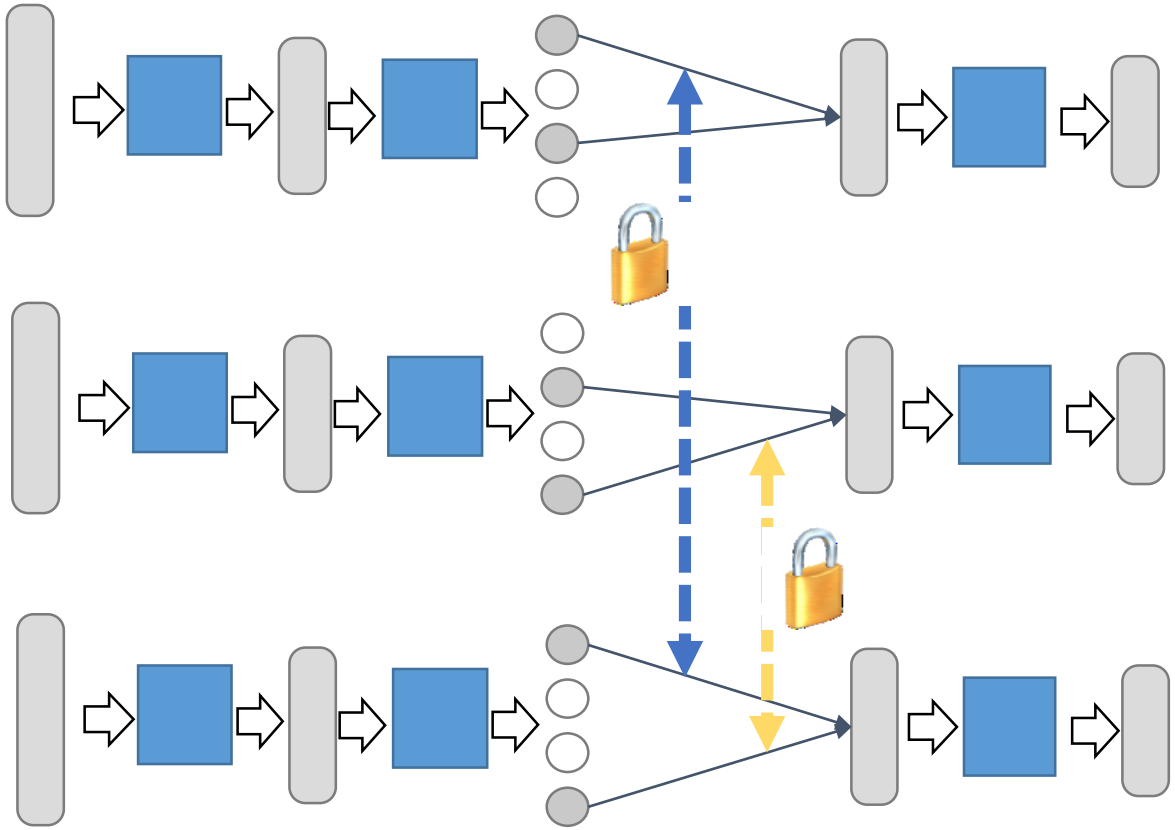
Результирующая модель:



Цель: “обучить” экспоненциальное количество таких моделей

[Srivastava et al. 2011]

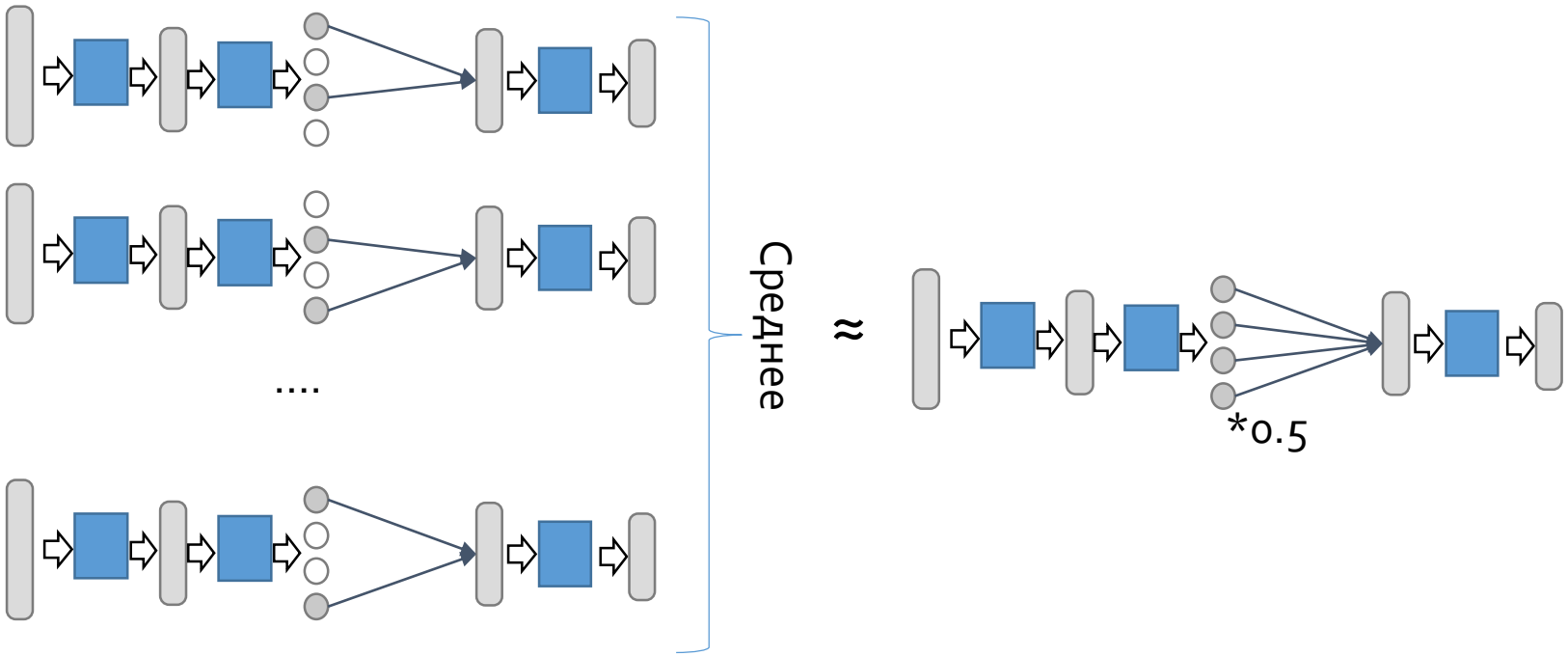
Обучение с Dropout



Обучаем очень большой набор таких моделей

[Srivastava et al. 2011]

Вывод с DropOut

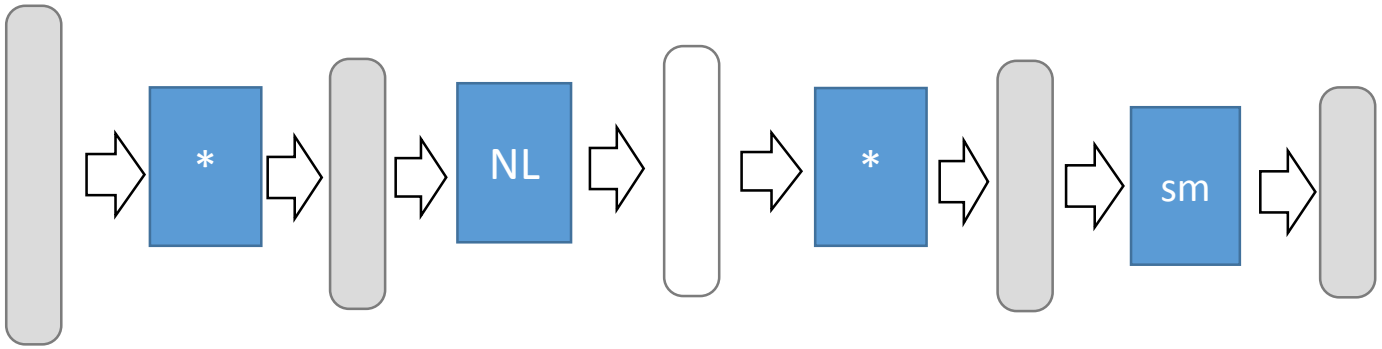


- Приближение не точное
- Но хорошо работает на практике

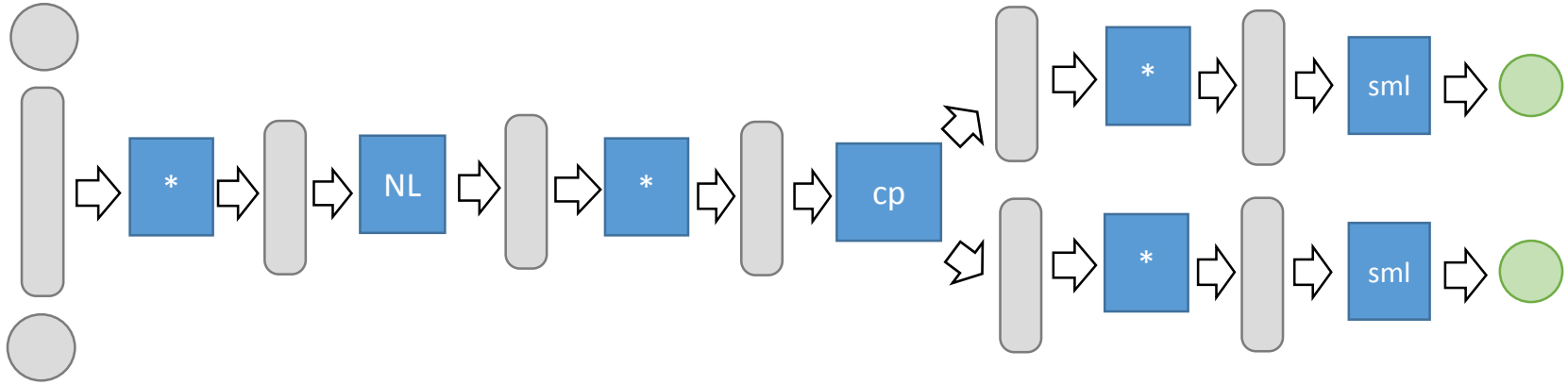
[Srivastava et al. 2011]

Реализация Dropout

- Во время тренировке с заданной вероятностью обнуляем случайные элементы данных
  - $n \sim Bernouli(p)$ $\quad\quad y = x * n$
  - $\frac{dz}{dx} = \frac{dz}{dy} * n$
- Во время вывода умножаем x на константу p, чтобы среднее значение активаций в модуле оставалось тем-же.



Многоцелевое обучение



- Две совместные задачи: пример (предсказание пола и возраста по фотографии)
- Есть много данных для близкой задачи
- Обучать совместно общую часть и отдельно для целевой задачи

Что такое «глубинное обучение»?

- Совместное обучение всех элементов модели
  - Модель – ориентированный граф из вычислительных блоков
  - Каждый блок дифференцируемый
  - Оптимизация градиентными методами
  - Градиенты вычисляются методом обратного распространения ошибки
- Революция глубинного обучения (2012?-now):
  - Инженерные решения соответствующие этим принципам