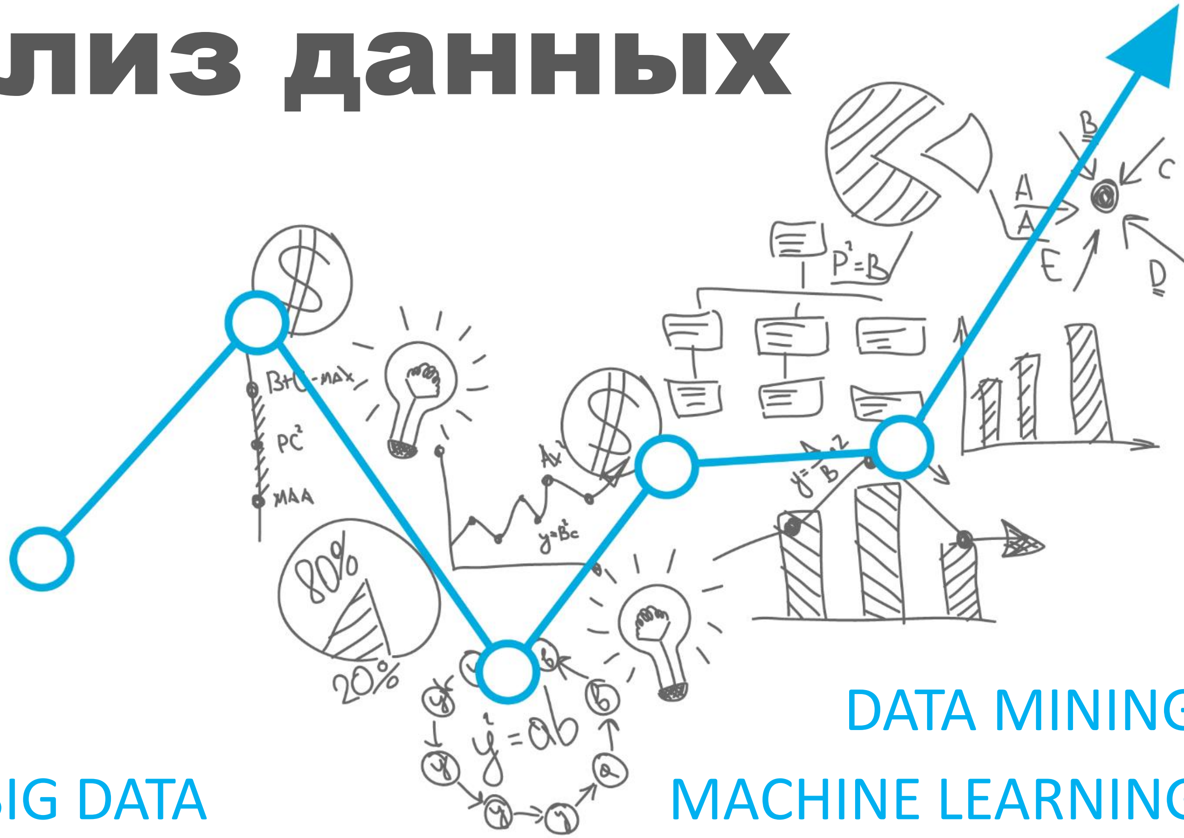
Интеллектуальный анализ данных



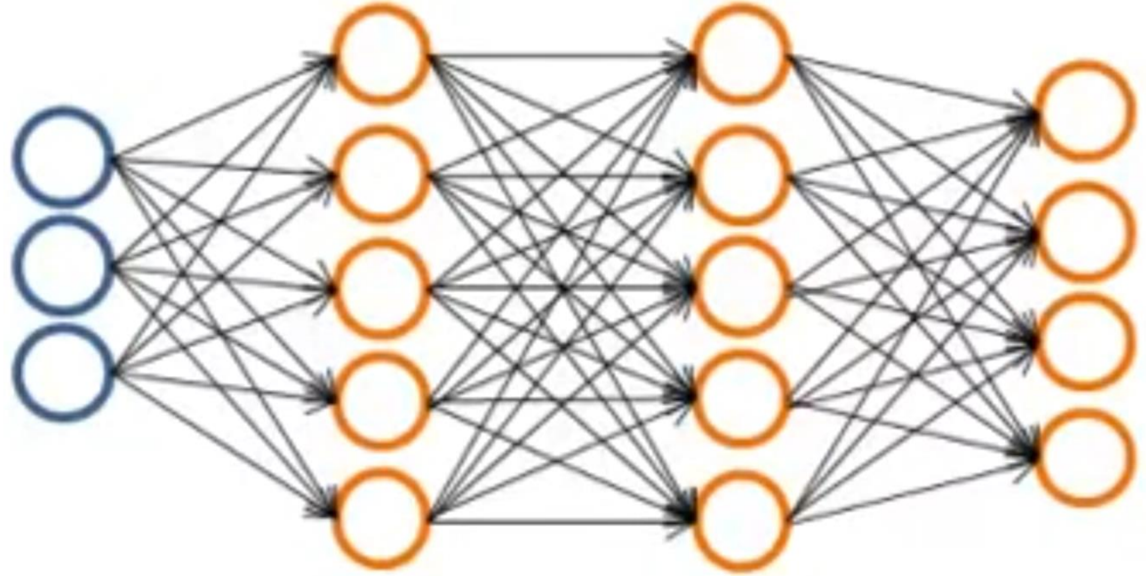
DATA MINING

BIG DATA

MACHINE LEARNING

Лекция 6. Обучение нейронных сетей

Нейронная сеть



Слой 1 Слой 2 Слой 3 Слой 4

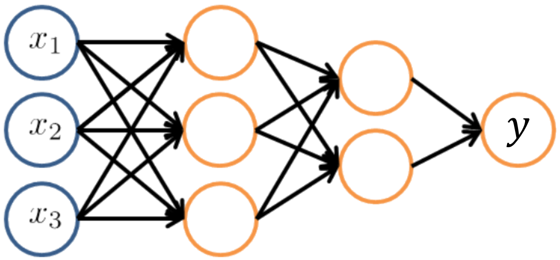
$s_1 = 3$ $s_2 = 5$ $s_3 = 5$ $s_4 = 4$

$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})\}$$

L – количество слоев в сети ($L = 4$)

s_l – количество нейронов в слое l ($1 \leq l \leq L$)
(фиктивный +1 не учитывается)

Задача классификации



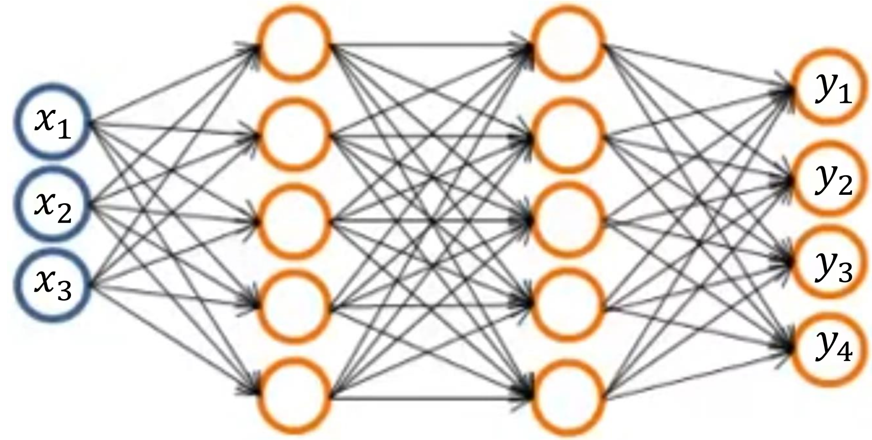
Бинарная классификация

$y \in \{0,1\}$ $\qquad y = 0$

$y = 1$

Один выходной нейрон

$h_\Theta(x) \in \mathbb{R}$ $\qquad s_L = 1$ $\qquad K = 1$



Множественная классификация
(на K классов)

$y \in \mathbb{R}^K$

$$y = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad y = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

K выходных нейронов

$h_\Theta(x) \in \mathbb{R}^K$ $\qquad s_L = K$ $\qquad K \geq 3$

Функция стоимости

Логистическая регрессия с регуляризацией:

$$J(\theta) = \boxed{-\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right) + (1-y^{(i)})\log\left(1-h_\theta\left(x^{(i)}\right)\right)\right]} + \boxed{\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2}$$

Нейронная сеть:

$$J(\Theta) = \boxed{-\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[y_k^{(i)}\log\left(\left(h_\Theta\left(x^{(i)}\right)\right)_k\right) + \left(1-y_k^{(i)}\right)\log\left(1-\left(h_\Theta\left(x^{(i)}\right)\right)_k\right)\right]} +$$

$$+\boxed{\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(\Theta_{i,j}^{(l)}\right)^2}$$

$\left(h_\Theta\left(x^{(i)}\right)\right)_k$ – отклик k-го нейрона выходного слоя

Обучение нейронной сети

Нам необходимо минимизировать функцию $J(\Theta)$ по Θ с использованием одного из продвинутых методов оптимизации (сопряженных градиентов, стохастический градиентный спуск, БФГШ и т.д.).

Какую процедуру (или несколько) мы должны запрограммировать и передать функции оптимизации?

- ☒ Для вычисления Θ
- ☒ Для вычисления $J(\Theta)$
- ☒ Для вычисления частных производных $\frac{\partial}{\partial \Theta_{i,j}^{(l)}}$ для всех i, j, l
- ☑ Для вычисления $J(\Theta)$ и $\frac{\partial}{\partial \Theta_{i,j}^{(l)}}$ для всех i, j, l

Обучение нейронной сети

$$J(\Theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[y_k^{(i)}\log\left(\left(h_\Theta\left(x^{(i)}\right)\right)_k\right) + \left(1 - y_k^{(i)}\right)\log\left(1 - \left(h_\Theta\left(x^{(i)}\right)\right)_k\right)\right] +$$

$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(\Theta_{i,j}^{(l)}\right)^2$$

Обучение нейронной сети: $\min_{\Theta} J(\Theta)$ Нахождение таких весов Θ, при которых ошибка минимальна

Требуется вычислить: $J(\Theta)$ – значение функции стоимости

$\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta)$ – значения частных производных функции стоимости

Алгоритм прямого распространения

Алгоритм прямого распространения нужен для вычисления $h_\Theta(x)$:

Рассмотрим на примере одного обучающего примера $x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$

$$a^{(1)} = x$$

$$z^{(2)} = \Theta^{(1)} a^{(1)} \qquad a_0^{(1)} = x_0 = 1$$

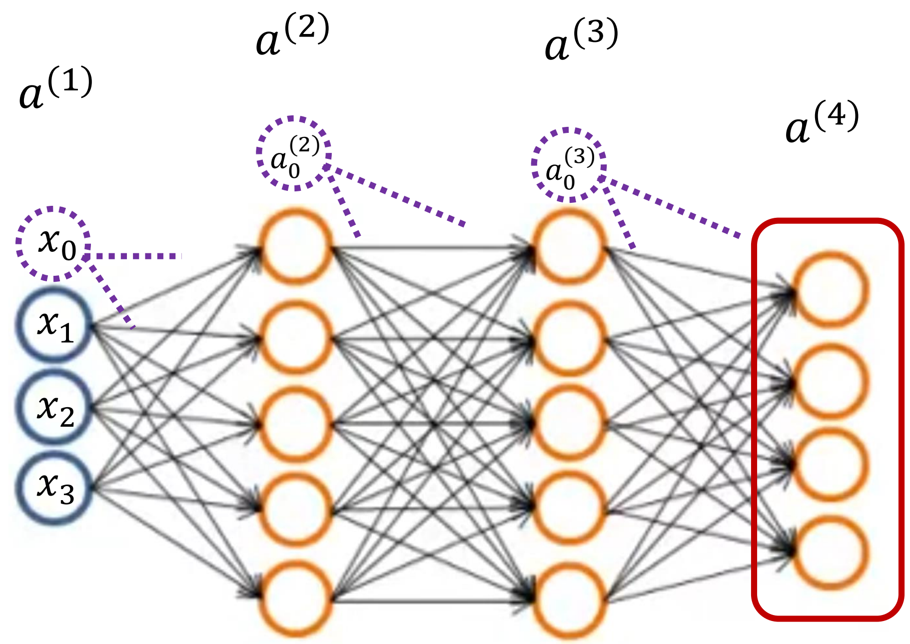
$$a^{(2)} = g(z^{(2)})$$

$$z^{(3)} = \Theta^{(2)} a^{(2)} \qquad a_0^{(2)} = 1$$

$$a^{(3)} = g(z^{(3)})$$

$$z^{(4)} = \Theta^{(3)} a^{(3)} \qquad a_0^{(3)} = 1$$

$$a^{(4)} = \boxed{h_\Theta(x)} = g(z^{(4)})$$



Алгоритм обратного распространения

Алгоритм обратного распространения (ошибки) нужен для вычисления $\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta)$

$a_j^{(l)}$ – активация (результат вычисления) j-го нейрона слоя l

$\delta_j^{(l)}$ – «ошибка» j-го нейрона слоя l

$$g(z) = \frac{1}{1+e^{-z}}$$

$$g'(z) = g(z)(1-g(z))$$

Для выходного слоя ($L = 4$):

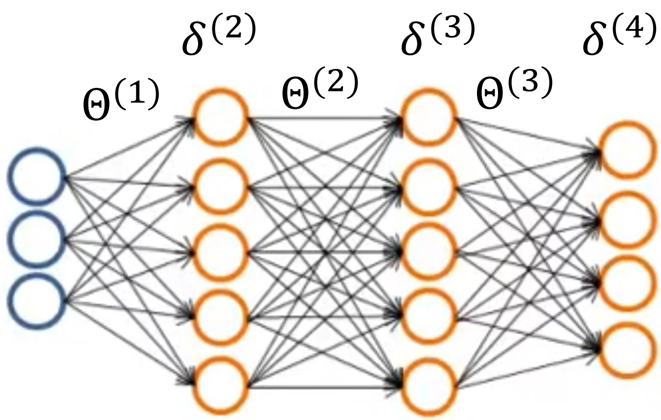
$$\delta^{(4)} = a^{(4)} - y \qquad \delta_j^{(4)} = a_j^{(4)} - y_j$$

Для $l = 3$:

$$\delta^{(3)} = \left(\Theta^{(3)}\right)^T \delta^{(4)} \cdot g'\left(z^{(3)}\right)$$

Для $l = 2$:

$$\delta^{(2)} = \left(\Theta^{(2)}\right)^T \delta^{(3)} \cdot g'\left(z^{(2)}\right)$$



Для $l = 1$ (входной слой) :

Ошибка не вычисляется

$$\boxed{\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta) = a_j^{(l)} \delta_i^{(l+1)}}$$

$$\frac{\partial}{\partial \Theta^{(l)}} J(\Theta) = \delta^{(l+1)} \left(a^{(l)}\right)^T$$

$$\begin{bmatrix} \delta \\ \delta \\ \delta \end{bmatrix} [a \quad a \quad a] = \begin{bmatrix} \theta & \theta & \theta \\ \theta & \theta & \theta \\ \theta & \theta & \theta \end{bmatrix}$$

На одном обучающем примере. Если их много?

Алгоритм обратного распространения

Вход: Обучающий набор: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})\}$

Веса сети: $\Theta = (\Theta^{(1)}, \dots, \Theta^{(L-1)})$; Параметр регуляризации: λ

Выход: Значения частных производных: $D^{(1)}, \dots, D^{(L-1)}$

Алгоритм:

$\Delta_{ij}^{(l)} = 0 \quad (\forall\, l, i, j)$

for $i = 1, \dots, m$

$$a^{(1)} = x^{(i)}$$

$$a^{(2)}, \dots, a^{(L)}, z^{(2)}, \dots, z^{(L)} = FP(x^{(i)}, \Theta)$$

$$\delta^{(L)} = a^{(L)} - y^{(i)}$$

$$\delta^{(l)} = \left(\Theta^{(l)}\right)^T \delta^{(l+1)} \cdot g'\left(z^{(l)}\right), \qquad l = L-1, \dots, 2$$

$$\Delta^{(l)} := \Delta^{(l)} + \delta^{(l+1)}\left(a^{(l)}\right)^T$$

end

$$D_{ij}^{(l)} = \begin{cases} \frac{1}{m}\Delta_{ij}^{(l)} + \lambda\Theta_{ij}^{(l)}, j \neq 0 \\ \frac{1}{m}\Delta_{ij}^{(l)}, \qquad j = 0 \end{cases}$$

$$\boxed{\frac{\partial}{\partial\Theta_{i,j}^{(l)}} J(\Theta) = D_{ij}^{(l)}}$$

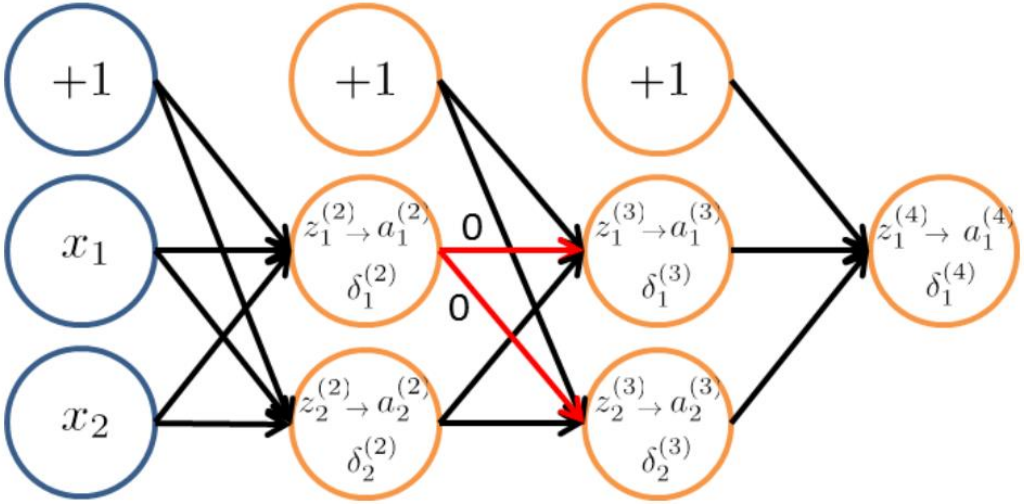
Алгоритм обратного распространения

Имеем два обучающих примера: $\left(x^{(1)}, y^{(1)}\right), \left(x^{(2)}, y^{(2)}\right)$. Какая из следующих последовательностей правильно вычисляет градиент?

(FP – прямое распространение, BP – обратное распространение)

- ☒ FP от $x^{(1)}$, FP от $x^{(2)}$, BP от $y^{(1)}$, BP от $y^{(2)}$
- ☒ FP от $x^{(1)}$, BP от $y^{(2)}$, FP от $x^{(2)}$, BP от $y^{(1)}$
- ☒ BP от $y^{(1)}$, FP от $x^{(1)}$, BP от $y^{(2)}$, FP от $x^{(2)}$
- ☑ FP от $x^{(1)}$, BP от $y^{(1)}$, FP от $x^{(2)}$, BP от $y^{(2)}$

Алгоритм обратного распространения

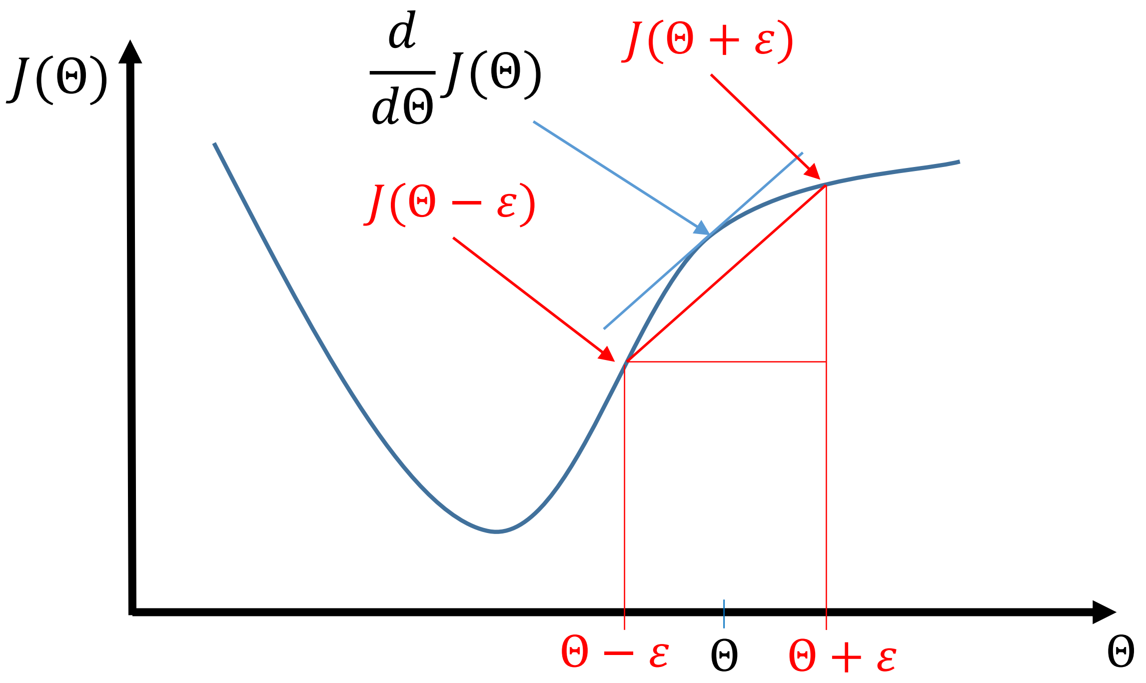


Пусть $\Theta_{11}^{(2)} = 0$ и $\Theta_{21}^{(2)} = 0$.

При расчете обратного распространения, что можно сказать про $\delta_1^{(3)}$?

- ☒ $\delta_1^{(3)} > 0$
- ☒ $\delta_1^{(3)} = 0$, если $\delta_1^{(2)} = \delta_2^{(2)} = 0$
- ☒ $\delta_1^{(3)} \leq 0$
- ☑ Не достаточно информации

Численная оценка градиента



$$\frac{d}{d\Theta}J(\Theta) \approx \frac{J(\Theta+\varepsilon) - J(\Theta-\varepsilon)}{2\varepsilon}$$

$\varepsilon \to 0$ $\qquad\qquad$ $\varepsilon = 10^{-4} \dots 10^{-6}$

Численная оценка градиента

Пусть $J(\theta) = \theta^3$. В точке $\theta = 1$ значение $\frac{d}{d\theta}J(\theta) = 3$. Используя $\varepsilon = 0.01$, чему будет равно

$$\frac{J(\theta+\varepsilon) - J(\theta-\varepsilon)}{2\varepsilon} \quad ?$$

- ☒ 3.0000
- ☑ 3.0001
- ☒ 3.0301
- ☒ 6.0002

Проверка вычисления градиента

Чтобы проверить правильность работы алгоритма обратного распространения:

– Вычисляем $D_{ij}^{(l)}$
– Вычисляем $N_{ij}^{(l)} = \frac{J(\Theta+\varepsilon)-J(\Theta-\varepsilon)}{2\varepsilon}$
– Если $D_{ij}^{(l)} \approx N_{ij}^{(l)}$, значит алгоритм отрабатывает верно
– После проверки численный расчет отключают и не используют

Почему не использовать формулу

$$\frac{d}{d\Theta}J(\Theta) \approx \frac{J(\Theta+\varepsilon)-J(\Theta-\varepsilon)}{2\varepsilon}$$

сразу вместо алгоритма обратного распространения?

Численная оценка градиента работает очень медленно!

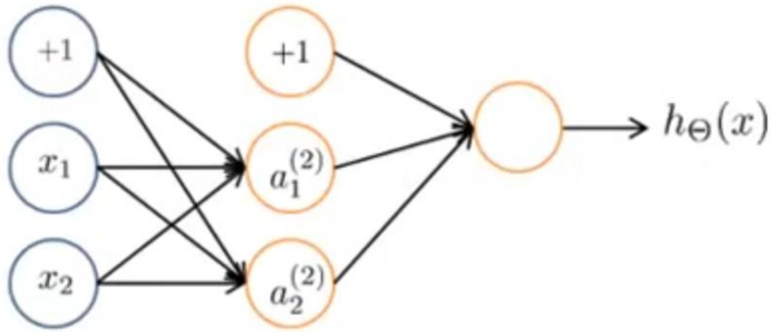
Какая основная причина использования для нахождения градиента алгоритма обратного распространения вместо численной оценки?

- ☒ Численная оценка градиента очень сложна в реализации
- ☑ Численная оценка градиента очень медленна
- ☒ Алгоритм обратного распространения не требует задавать параметр ε
- ☒ Ничего из перечисленного

Начальная инициализация

Какие значения задать $\Theta^{(1)}, \dots, \Theta^{(L-1)}$ до начала обучения нейронной сети ?

Нельзя использовать нулевые и любые одинаковые значения!



$$\Theta_{i,j}^{(l)} = 0$$

$$\forall\, l, i, j$$

$$a_1^{(2)} = a_2^{(2)} \Rightarrow \delta_1^{(2)} = \delta_2^{(2)} \Rightarrow \frac{\partial}{\partial \Theta_{11}^{(1)}} J(\Theta) = \frac{\partial}{\partial \Theta_{12}^{(1)}} J(\Theta) \Rightarrow \Theta_{11}^{(1)} = \Theta_{12}^{(1)}$$

«Проблема симметрии» – когда многие (все) нейроны имеют одинаковый отклик

Случайная инициализация:

$$\Theta_{i,j}^{(l)} = rand(-\tau, \tau)$$

τ – параметр

Начальная инициализация

Рассмотрим следующую процедуру начальной инициализации параметров нейронной сети:

Шаг 1. Вычислим число $r = rand(-\tau, \tau)$

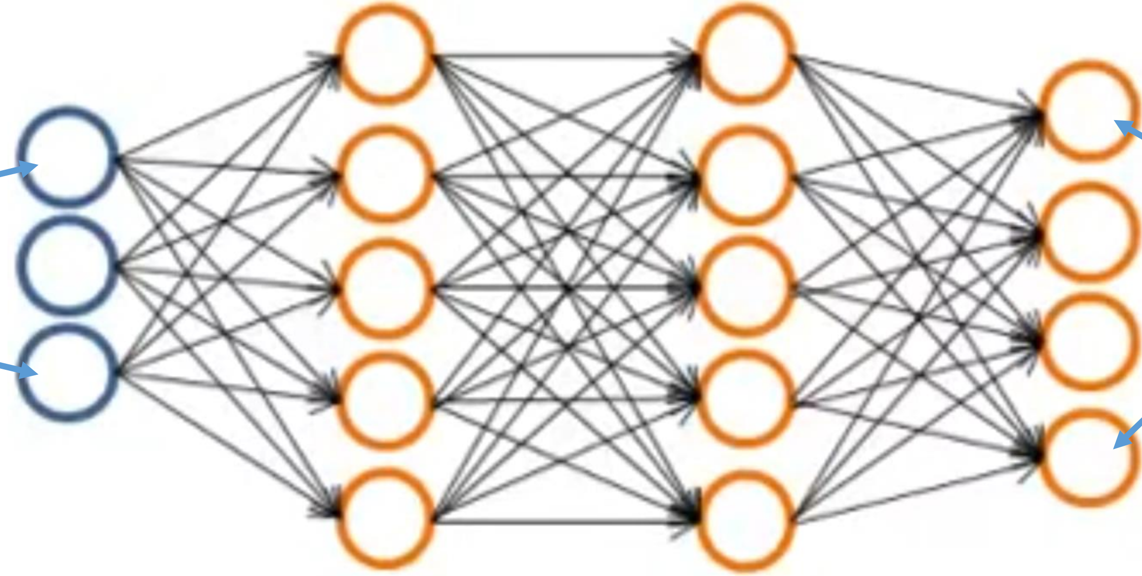
Шаг 2. $\Theta_{i,j}^{(l)} = r$ для $\forall\, l, i, j$

Будет ли такая начальная инициализация удачной?

- [ ] Да, параметры заданы случайно
- [ ] Да, если $r \neq 0$
- [x] Нет, это создает проблему симметрии
- [ ] Возможно, зависит от входных x

Обучение нейронной сети

$$x^{(i)} = \begin{bmatrix} x_1^{(i)} \\ \dots \\ x_{s_1}^{(i)} \end{bmatrix}$$



$$\begin{bmatrix} y_1^{(i)} \\ \dots \\ y_{s_L}^{(i)} \end{bmatrix} = y^{(i)}$$

$s_1 = 3$ $\quad s_2 = 5$ $\quad s_3 = 5$ $\quad s_L = 4$

Обучающая выборка: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})\}$

$$y^{(i)} = \begin{bmatrix} y_1^{(i)} \\ \dots \\ y_{s_L}^{(i)} \end{bmatrix} \qquad y^{(i)} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}? \qquad h_\Theta(x^{(i)}) = \begin{bmatrix} a_1^{(L)} \\ \dots \\ a_{s_L}^{(L)} \end{bmatrix} \qquad h_\Theta(x^{(i)}) = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.6 \\ 0.2 \end{bmatrix}?$$

$s_L =$ "количеству классов"

Обучение нейронной сети

1. Выполнить начальную инициализацию весов Θ случайными значениями

2. Реализовать алгоритм прямого распространения (FP) для вычисления $h_\Theta\left(x^{(i)}\right)$ для любого входного $x^{(i)}$

3. Реализовать функцию вычисления $J(\Theta)$

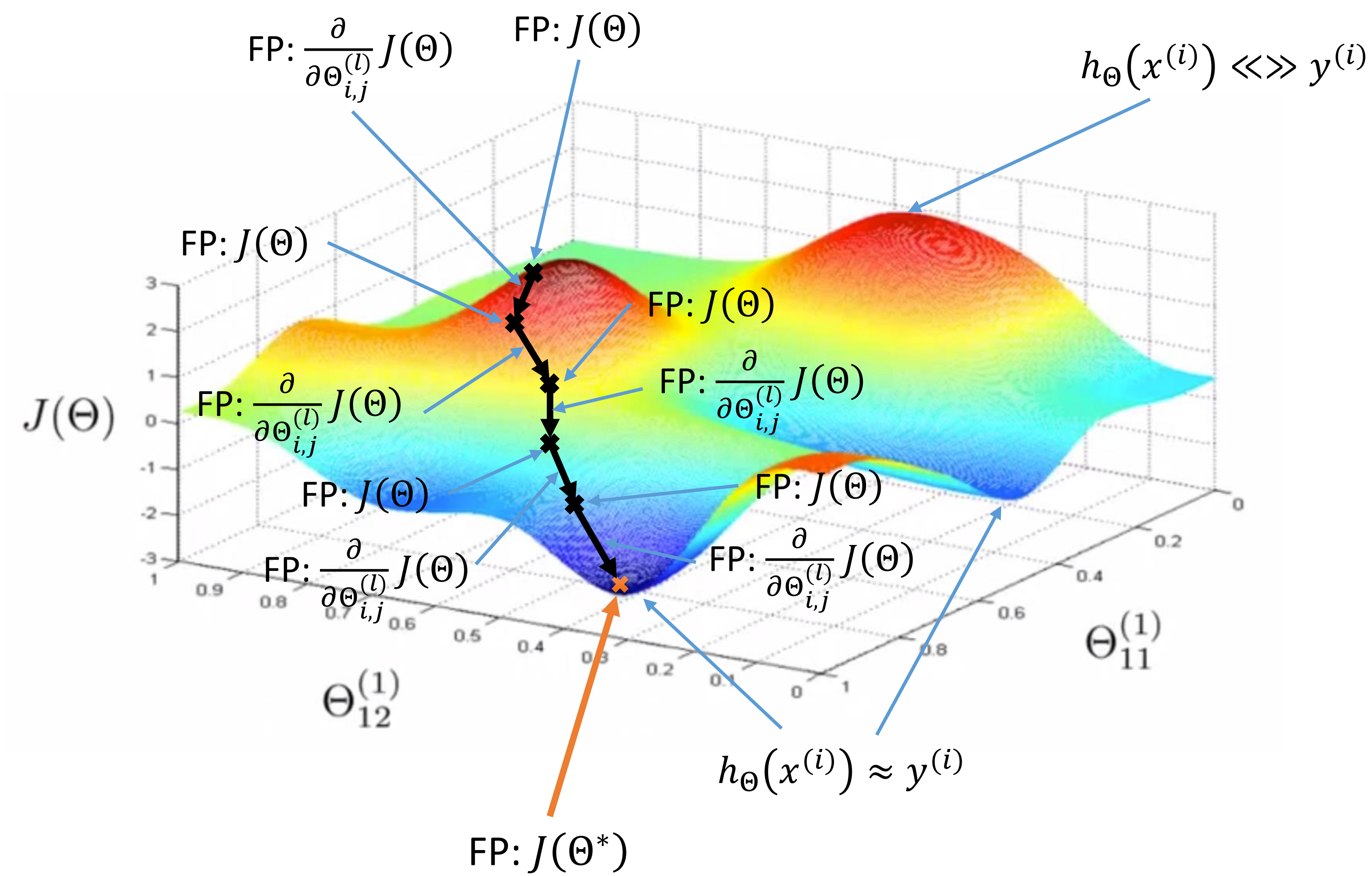
4. Реализовать алгоритм обратного распространения (BP) для вычисления частных производных $\frac{\partial}{\partial \Theta_{i,j}^{(l)}} J(\Theta)$

5. (Опционально) Реализовать численную оценку градиента и проверить правильность работы алгоритма BP

6. Использовать градиентный спуск или более продвинутый алгоритм оптимизации для нахождения оптимальных Θ

Обучение нейронной сети

Обучение нейронной сети



Функция $J(\Theta)$ не выпуклая, рекомендуется продвинутый алгоритм оптимизации (СГС, БФГШ и др.: Adam, RMSProp, Shampoo, …)

Обучение нейронной сети

Вы обучаете нейронную сеть с использованием градиентного спуска и алгоритмов прямого и обратного распространения для минимизации функции стоимости $J(\Theta)$ по параметрам Θ.
Каким способом вы можете убедиться, что процесс обучения проходит корректно?

- ☒ Отобразить график $J(\Theta)$ от Θ и убедиться, что градиентный спуск движется в нисходящем направлении
- ☒ Отобразить график $J(\Theta)$ от номера итерации и убедиться, что он увеличивается (не уменьшается) с каждой итерацией
- ☑ Отобразить график $J(\Theta)$ от номера итерации и убедиться, что он уменьшается (не увеличивается) с каждой итерацией
- ☒ Отобразить график $J(\Theta)$ от номера итерации и убедиться, что с каждым шагом качество классификации улучшается