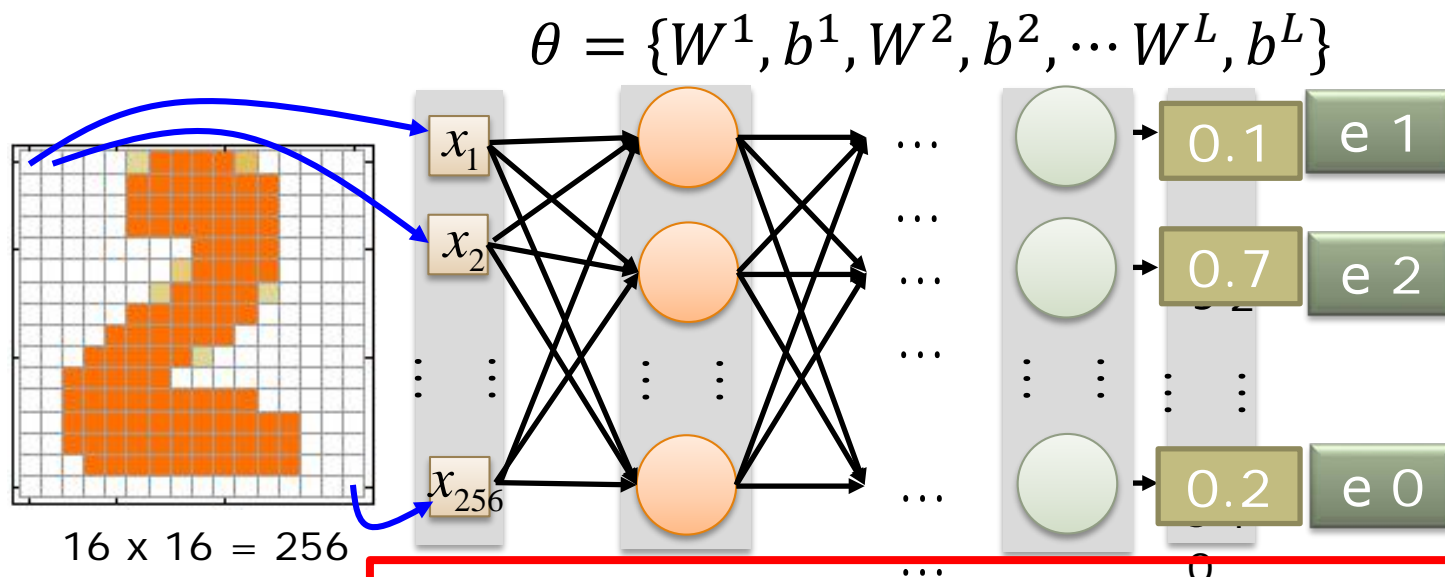


---

**Алгоритми за обучение и настройка на дълбоки невронни мрежи, инициализация. Градиентен подход. Многокритериална оптимизация, нормиране на данните, подготовка и формиране на извадки от данни за тестване и обучение на дълбоки невронни мрежи.**

---

## Параметри на дълбоките невронни мрежи



Задаване на параметрите на НМ  $\theta$ , при които....

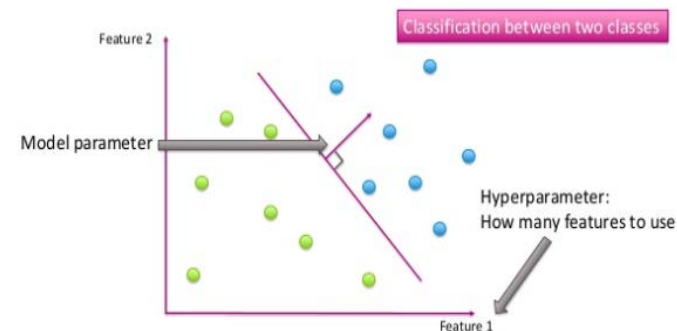
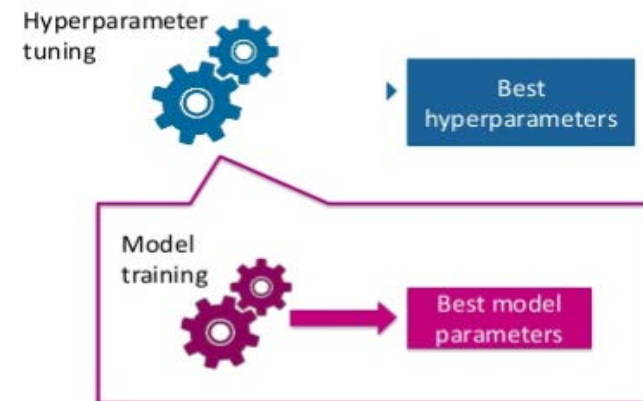
Вход:  →  $y_1$  има максимална стойност

Вход:  →  $y_2$  има максимална стойност

Как невронната  
мрежа постига това

# Моделни параметри и хиперпараметри

- Моделните параметри са променливи на избрания модел, които може да бъде оценени чрез напасване на дадените данни към модела.
- Хиперпараметър е параметър от предишно разпределение, преди данните да бъдат наблюдавани.
  - Това са параметрите, които контролират параметрите на модела
  - Във всеки алгоритъм за машинно обучение тези параметри трябва да бъдат инициализирани преди обучението на модела.



# Дълбока невронна мрежа: параметри срещу хиперпараметри

---

- **Параметри:**

- $W^1, b^1, W^2, b^2, \dots, W^L, b^L$

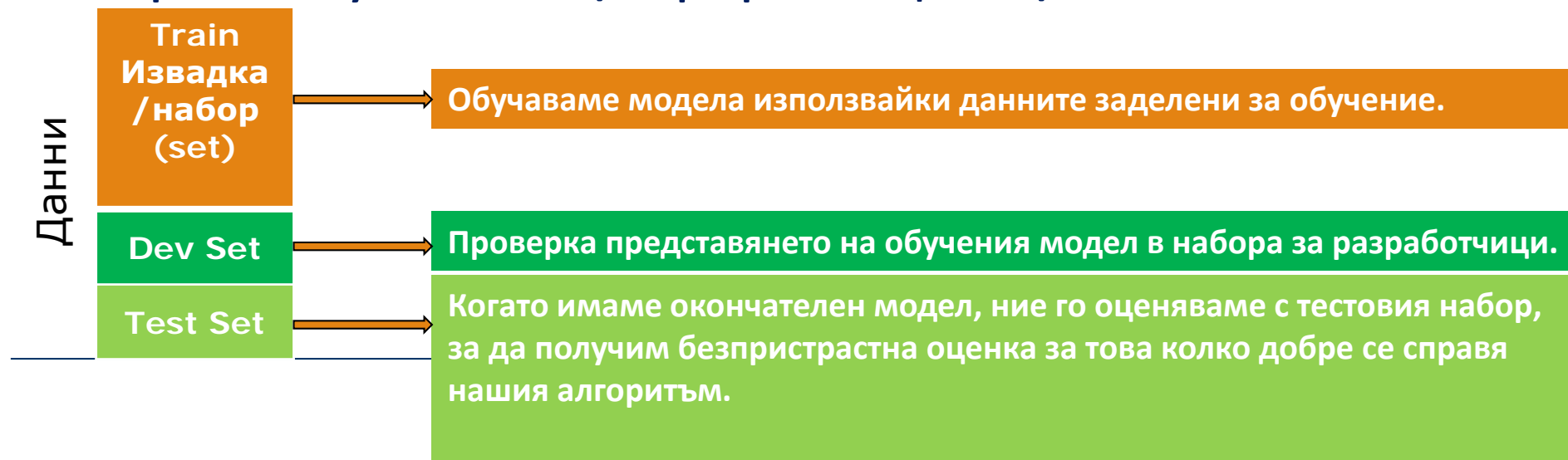
- **Хиперпараметри:**

- Скорост на сходимост ( $\alpha$ ) в посоката на намаляване на градиента
- Броят итерации в намаляването на градиента
- Бройката слоеве в Невронната Мрежа
- Брой неврони в слоевете на Невронната Мрежа
- Вида на активационните функции
- Размер на мини партида
- Регуларизационни параметри

## Train / Dev / Test извадки/набори (set)

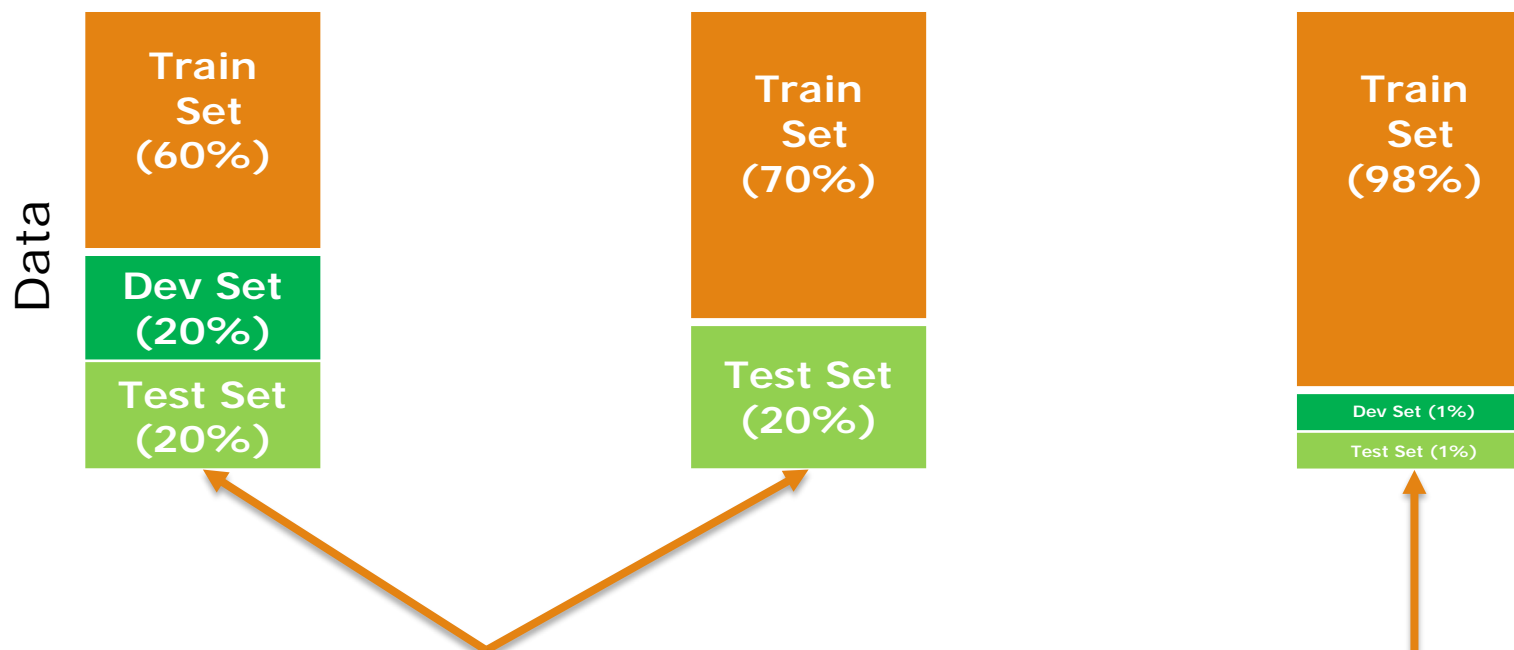
---

- Настройката на хиперпараметри е силно итеративен процес, при който:
  - Започваме с начална идея, т.е. започваме с определен брой скрити слоеве, определена скорост на обучение и т.н.
  - Опитайте идеята, като я приложите/реализирате
  - Експериментирайте с това, до колко добре е сработила идеята
  - Усъвършенствайте идеята и повторете този процес
- По какъв начин да определим дали идеята работи? Тук влизат в помощ наборите за обучение Train / за разработчици Dev / тестове Test.



## Train / Dev / Test извадки/набори (set)

---

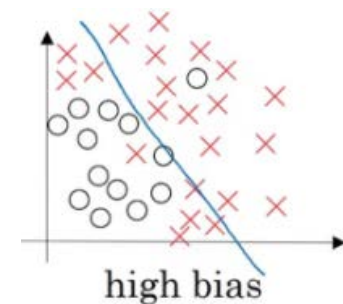
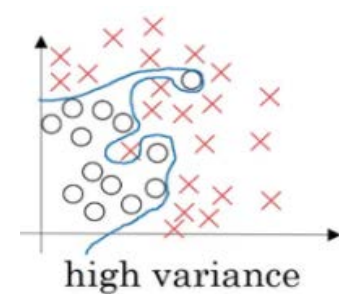
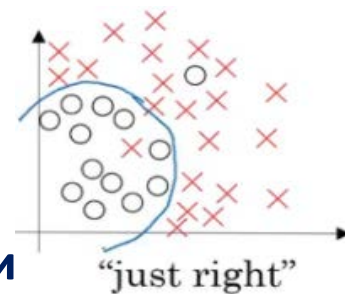
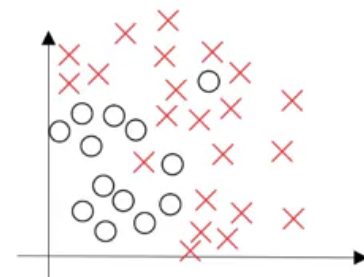


В предходни години, когато имаше на разположение само малки набори от данни, най-често разпределението на различни набори бе следното

Тъй като през последните години наличността на данни се е увеличила, можем да използваме по-голяма (огромна) част от тях за обучение на модела

## Отместване / Вариация и компромиси

- Необходимо е разпределението на dev/test извадката да е същото като обучителния набор
  - \*Разделете наборите за обучение, разработка и тестове така, че да са със сходни разпределения
  - \*Пропуснете тестовия набор и валидирайте модела, като използвате само набора за разработчици
- Искаме нашият модел да бъде коректния, което означава да имаме малко отместване и ниска вариация.
- **Overfitting/пrenaпасване:** ако грешката при набора за разработчици е много по-голяма от тази при тестовата извадка, моделът е пренапаснат и с голяма вариация
- **Underfitting:** когато грешката е голяма едновременно и при train и dev извадки, моделът е недонапаснат и с голямо отместване



## Overfitting Пренапасване в дълбоките невронни мрежи

---

- Дълбоките невронни мрежи съдържат множество нелинейни скрити слоеве
  - Това ги прави едни много изразителни/описателни модели, които могат да изучат и научават много сложни връзки между техните входи и изходи.
  - С други думи, моделът научава и апроксимира дори най-малките детайли, които присъстват в данните.
- С ограничени ресурси и респективно, ограничени данни за обучение, много от тези сложни взаимоотношения ще са резултат на шум или греша в процеса на семплиране
  - Така че, те ще съществуват в набора за обучение, но не и в реалните тестови данни, дори ако са извлечени от същото разпределение.
  - Така че, след като научи всички възможни модели, които може да намери, моделът има тенденция да се представя изключително добре в обучителния набор, но не успява да даде добри резултати в наборите за разработка и тестове.



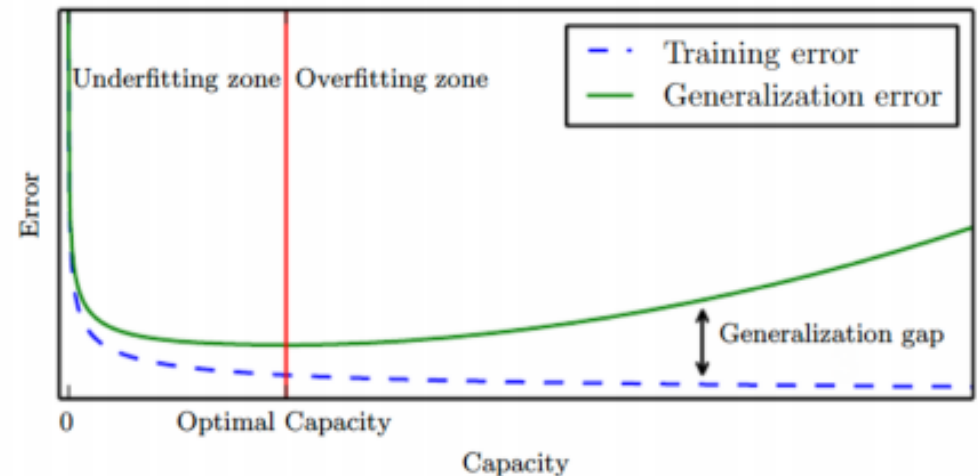
# Регуляризация

---

- Регуляризацията е:

- “всяка модификация на алгоритъма за обучение, направена за да се намали неговата грешка при обобщаване, но не и грешката при обучение”
- Намаляване на грешката при обобщаване дори за сметка на увеличаване на грешката при обучение

- Например, ограничаването на капацитета на модела е метод за регуляризация



## Наказателни нормиращи коефициенти

---

- Най-традиционния вид регуляризация прилагана в дълбокото обучение е тази с прилагане на **наказателни нормиращи коефициенти**.
- С този подход се ограничава капацитета на модела с въвеждането на наказанието  $\Omega(\theta)$  в целевата функция довеждащо до:

$$\min_{\theta} J = \ell(\theta) + \lambda \Omega(\theta)$$

- $\lambda \in [0, \infty)$  е хуперпараметър, който оценява и претегля относителния принос на наказанието от нормата към стойността на целевата функция.

## L2 Норма за регуляризация на параметри

---

- С използване на L2 норма, се въвеждат ограничения към оригиналната функция на загубите, така, че теглата на невронната мрежа да не стават прекалено големи.

$$\Omega(\theta) = ||\theta||_2^2$$

- Ако приемем, че нямаме отмествания, а само тегла

$$\Omega(w) = ||w||_2^2 = w_{11}^2 + w_{12}^2 + \dots$$

- С добавяне на регуляризационни коефициенти се подвежда модела, по начин да не води грешката от обучение към нулата, с което на свой ред се намалява сложността на модела.

## L1 Норма за регуляризация на параметри

---

- L1 нормата представлява друга опция, която може да се използва за да се ,накажат' размерностите на моделните параметри.
- L1 регуляризация на тегловите коефициенти  $w$  е:

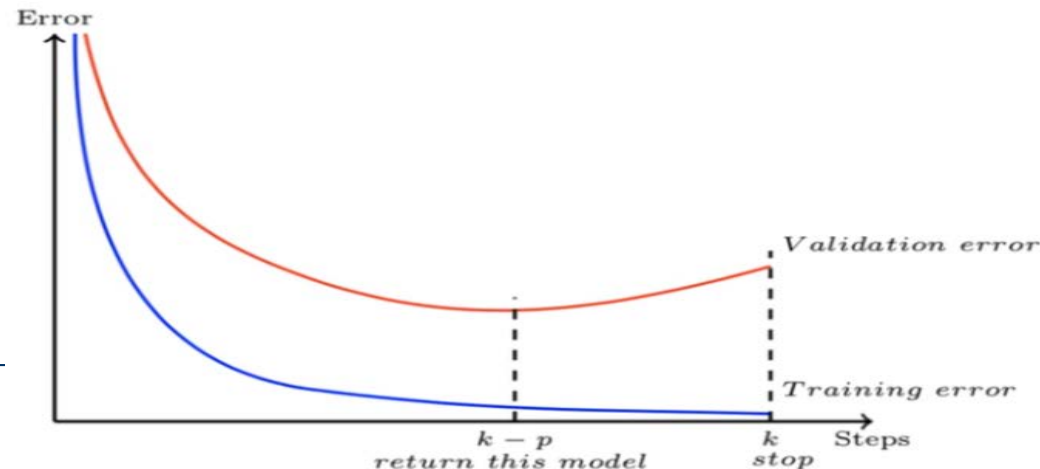
$$\Omega(w) = ||w||_1 = \sum_i |w_i|$$

- L2 норма наказателните коефициенти затихват тези компоненти от вектора на  $w$ , които не допринасят значително за намаляването на целевата функция.
- От друга страна, наказанието за норма L1 предоставя решения, които са **оскъдни /sparse/**.
- Това качество за **оскъдност /sparse/** може да се възприеме и като **механизъм за избор на информативни признаци/feature selection/**.

## Ранно спиране на обучението

---

- Когато обучаваме модели с достатъчен представителен капацитет, за да отговарят на задачата, често наблюдаваме, че грешката в обучението намалява постоянно с времето, докато грешката в набора за валидиране започва да нараства отново или остава същата за определени итерации, като тогава няма смисъл да обучаваме модела по-нататък.
- Това означава, че можем да получим модел с по-добра грешка в набора за валидиране (и по този начин, да се надяваме на по-добра грешка в набора за тестове), като се върнем към настройката на параметрите в момента с най-ниската грешка в набора за валидиране



## Обвързване на параметри

---

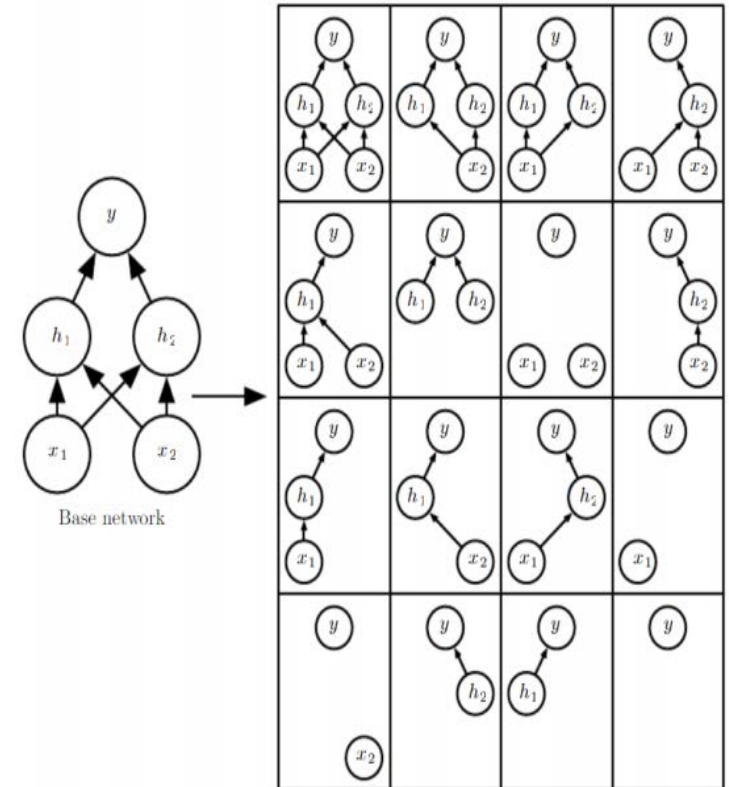
- Понякога може да не знаем в кой регион ще лежат параметрите, а по-скоро знаем, че има някои зависимости между тях.
- **Обвързването на параметрите** се отнася до изрично принудително принудяване на параметрите на два модела да бъдат близо един до друг, чрез използване на наказателни коефициенти за норма.

$$||W^{(A)} - W^{(B)}||$$

- Тука,  $W^{(A)}$  се отнася към тегловите коефициенти на първият модел, докато  $W^{(B)}$  се отнася към тегловите коефициенти на втория модел.

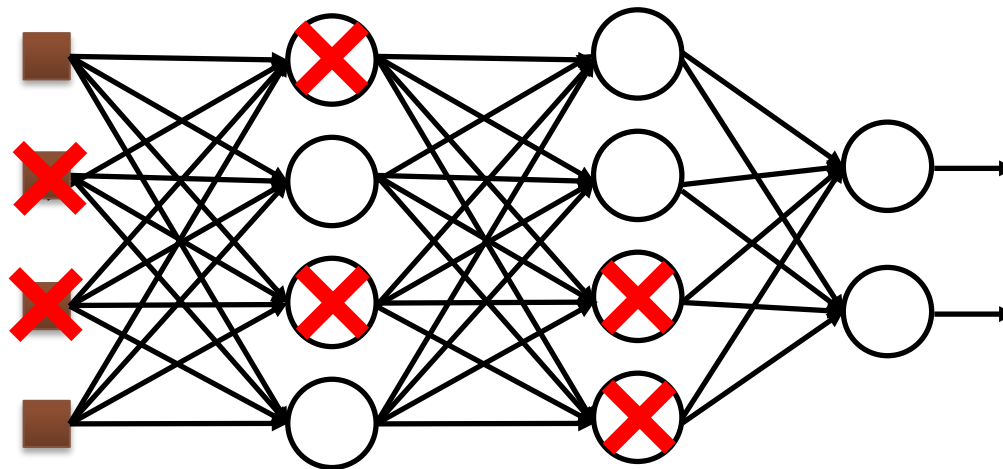
## Отпадания

- Отпаданията са вид метод за пакетирания
  - Пакетирането е методика за усредняване на няколко модела с цел подобряване на генерализацията
    - Не е практично да се обучават множество невронни мрежи, т.к. това е ,скъп‘ по време и използвана оперативна памет процес
  - Представява метод за пакетиране приложен към невронните мрежи
- Отпадането е лесен за реализация и с ниско ресурсно натоварване метод за регуляризация на големи групи от модели
- По-конкретно, с отпадането се обучава ансамбъла, състоящ се от всички подмрежи, които могат да бъдат образувани чрез премахване на неизходни единици от основната базова мрежа.



## Отпадания – интуитивна причина

---



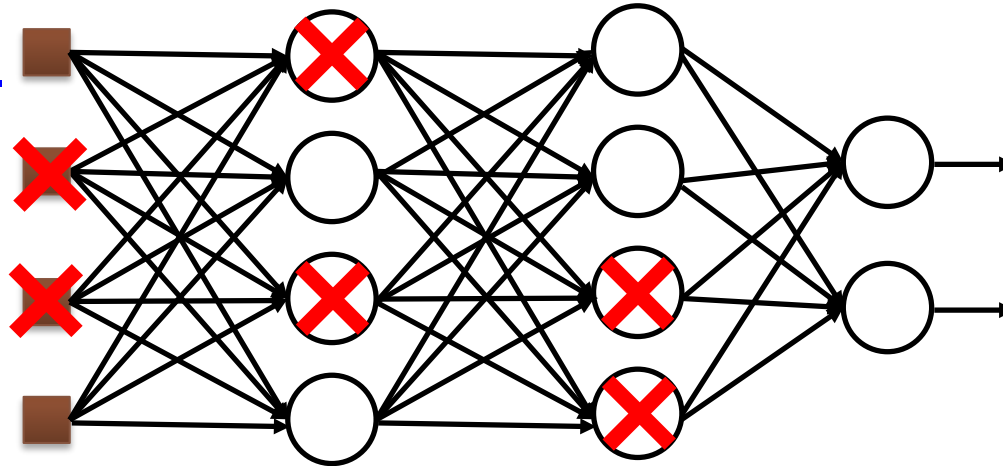
- Когато всички работят в екип, всички очакват останалите да свършат работата и накрая се стига да няма свършена работа
- Но!, ако знаете, че вашите партньори ще отпаднат, то вие ще си свършите по-добре работата.
- При тестването никой не отпада, така че евентуално някога ще се постигнат добри резултати.



## Отпадания

---

Обучение:

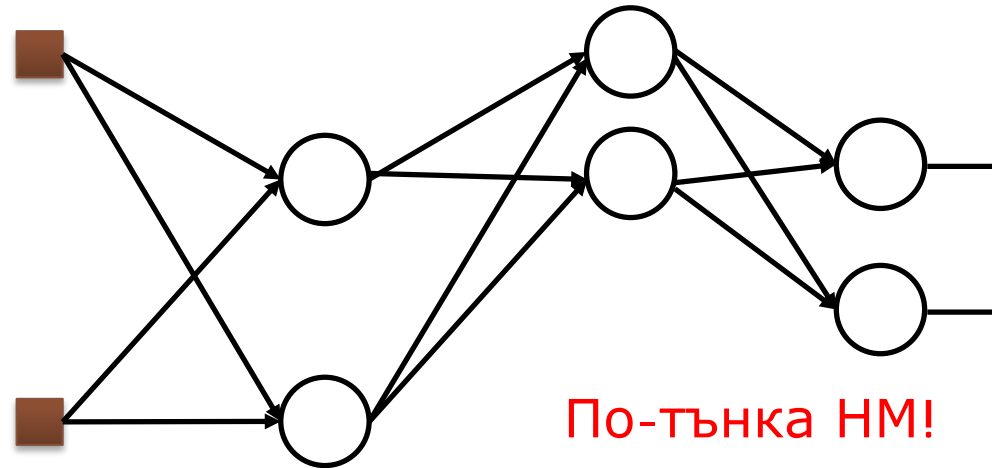


- **Всеки път преди изчислението на градиентите**
  - Всеки един неврон има  $p\%$  вероятност за отпадане

## Отпадания

---

Обучение:



➤ **Всеки път преди изчислението на градиентите**

- Всеки един неврон има  $p\%$  вероятност за отпадане



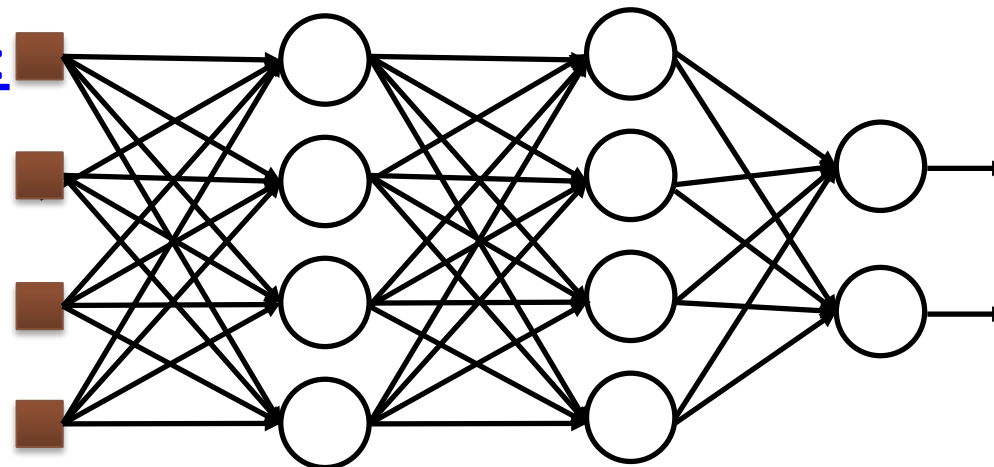
**Структурата на невронната мрежа бива променена.**

- Използване на новата НМ за обучение.

## Отпадания

---

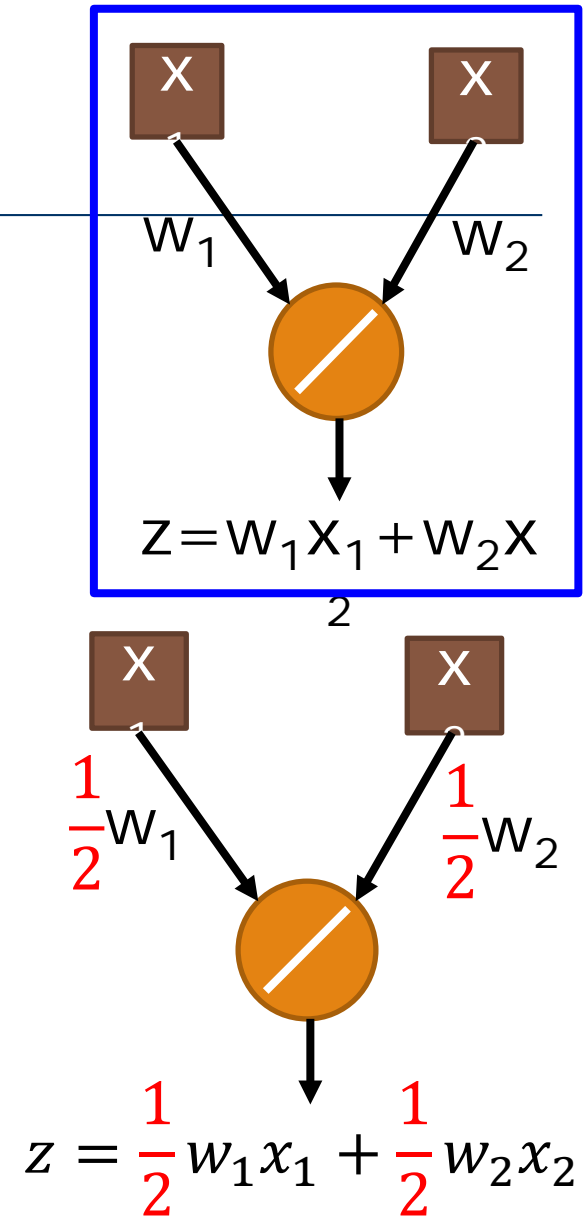
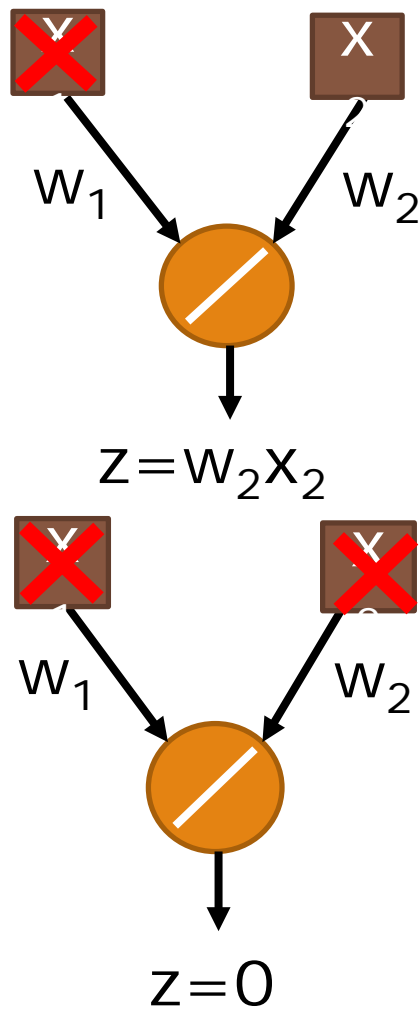
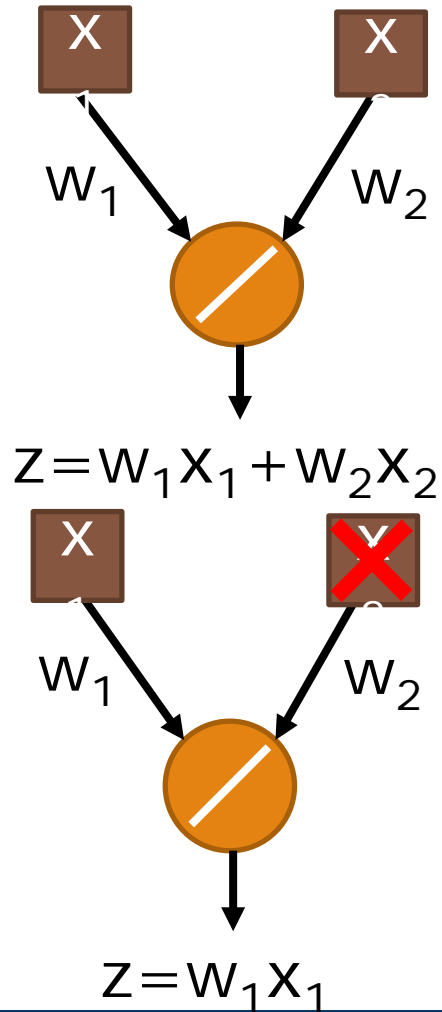
Обучение :



### ➤ Няма отпадания

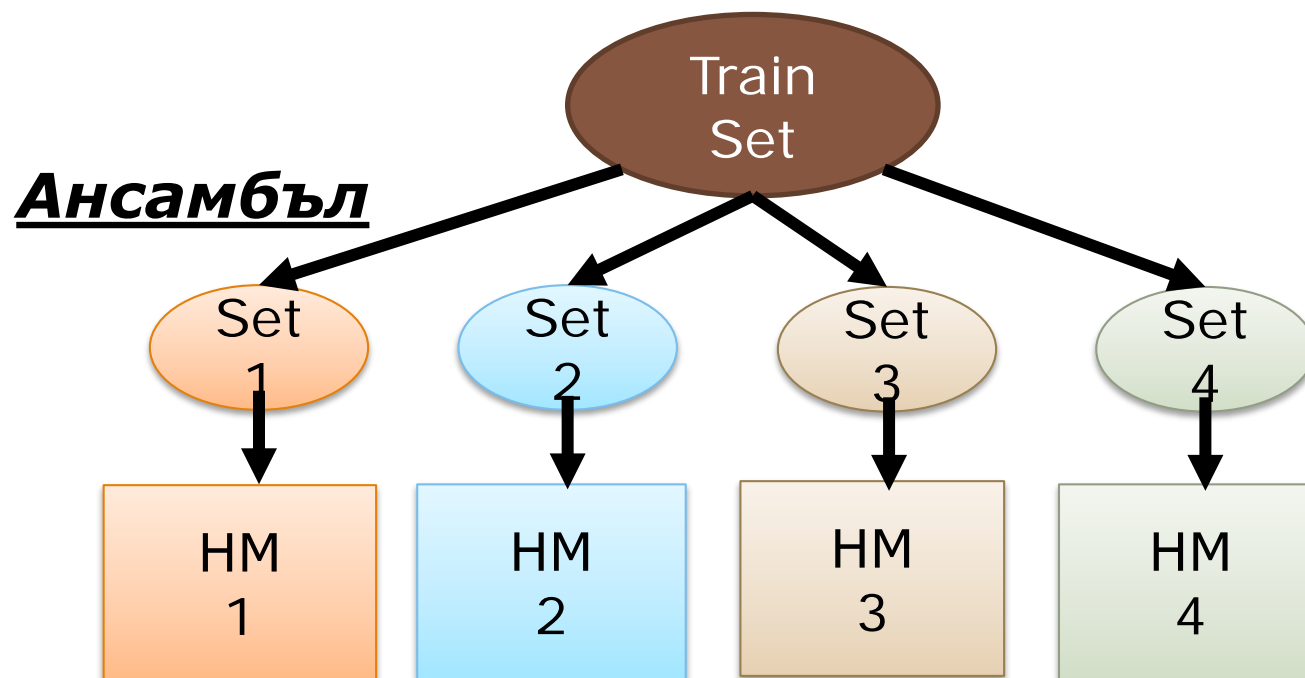
- Ако процентът на отпадане за един интервал е  $p\%$ , всичките тегла умножени по  $(1-p)\%$
- Да приемем, че процентът на отпадане е 50%.  
Ако едно тегло  $w = 1$  при обучението, да се зададе  $w = 0.5$  при тестването.

Защо теглата трябва да се умножават с (1-p)%  
(процент на отпадане) при тестване?



## Отпадането е един вид ансамбъл

---

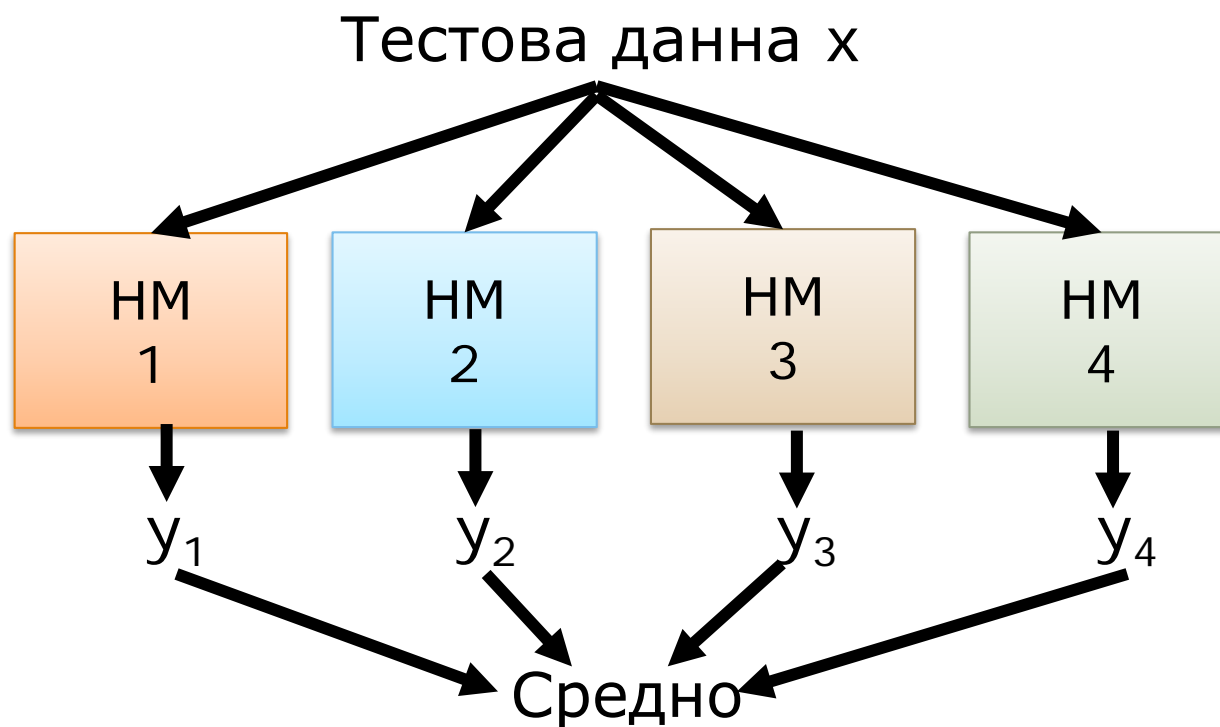


Обучава се група от невронни мрежи с различни структури

## Отпадането е един вид ансамбъл

---

### Ансамбъл



# Настройка на оптимизацията

## Нормализация на входните данни

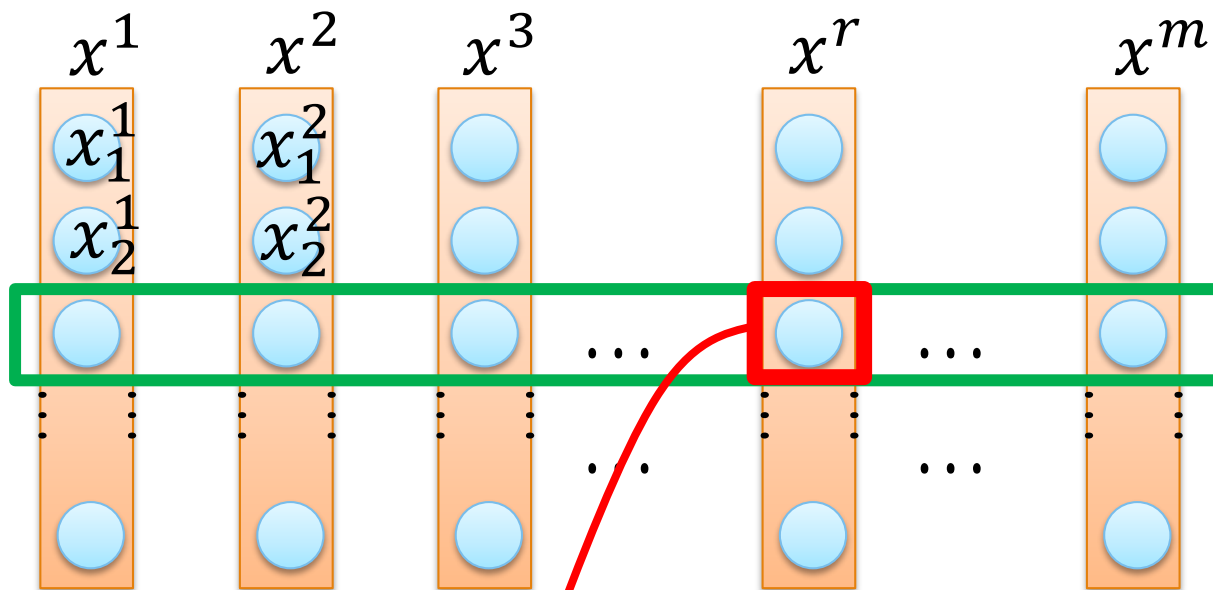
---

- Диапазона на стойностите на първичните входни данни между отделните входове в обучаващата извадка много често варира значително
  - Например: Вход/инф.признак „Има деца“ е  $\{0,1\}$
  - Стойност на кола: започва от 500лв и стига до 1500500лв.
- Ако една от характеристиките има широк диапазон от стойности, разстоянието ще се управлява от тази конкретна характеристика.
  - След нормализация, всеки един информативен признак допринася приблизително еднакво пропорционално към крайните разстояния между данните.
- По принцип, градиентните подходи имат много по-бърза сходимост ако има приложено скалиране на информ. признаци спрямо ако няма такова.

Представява Добра практика за числена стабилност при числени изчисления и за избягване на лоши условия при решаване на системи от уравнения

---

## Скалиране на информативните признаци



За всяка  
размерност  $i$ :  
средно:  $m_i$   
стандартно  
отклонение:  $\sigma_i$

$$x_i^r \leftarrow \frac{x_i^r - m_i}{\sigma_i}$$

Средните стойности за всички  
размерности са 0, а всички  
вариации са 1

В цялост намаляването на градиента е сходящо много  
по-бързо със скалиране на признаците, отколкото без.



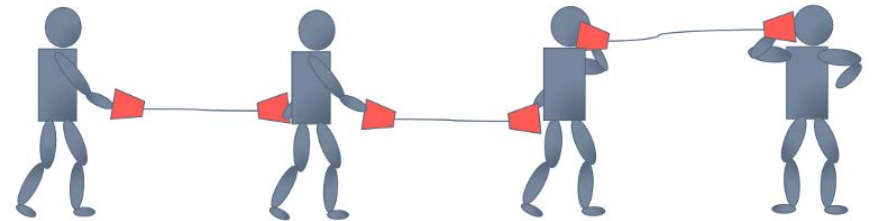
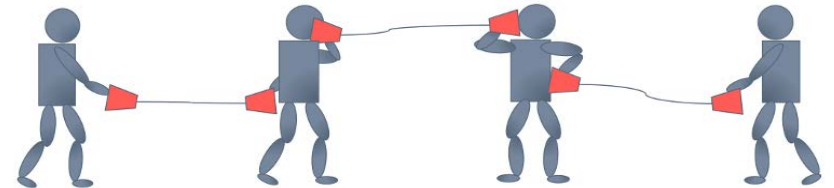
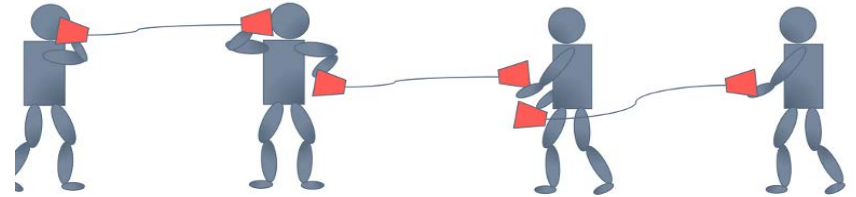
## Вътрешно ковариационно отместване

---

\* Първият човек казва на втория човек, „върви поливай растенията“, вторият казва на третия „имам вода в гащите ти“ и така нататък, докато последният не чуе „хвърчило, яжте маймуна с лице“ или нещо напълно погрешно.

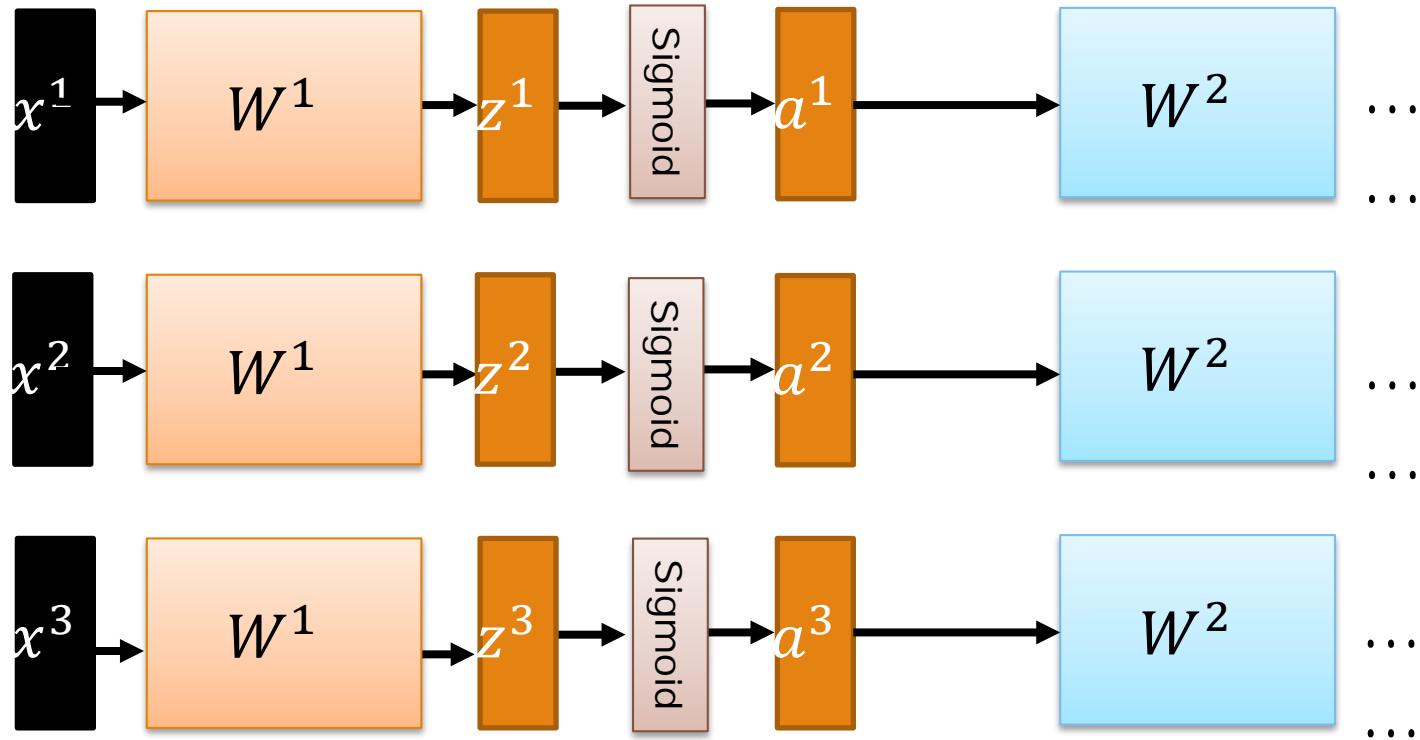
\* Да кажем, че проблемите са изцяло системни и се дължат изцяло на дефектни червени чаши. Тогава ситуацията е аналогична на фазата на разпространението на сигналите напред (feedforward).

\* Ако можете да получите нови чаши, за да решите проблема чрез проба и грешка, би помогнало да имате последователен начин за предаване на съобщения по по-контролиран и стандартизиран („нормализиран“) начин. напр.: Същият обем, същия език и т.н

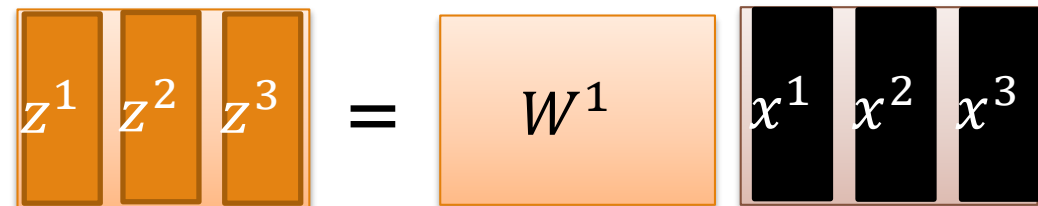


“Параметрите на първия слой се променят и така разпределението на входа към втория слой се променя”

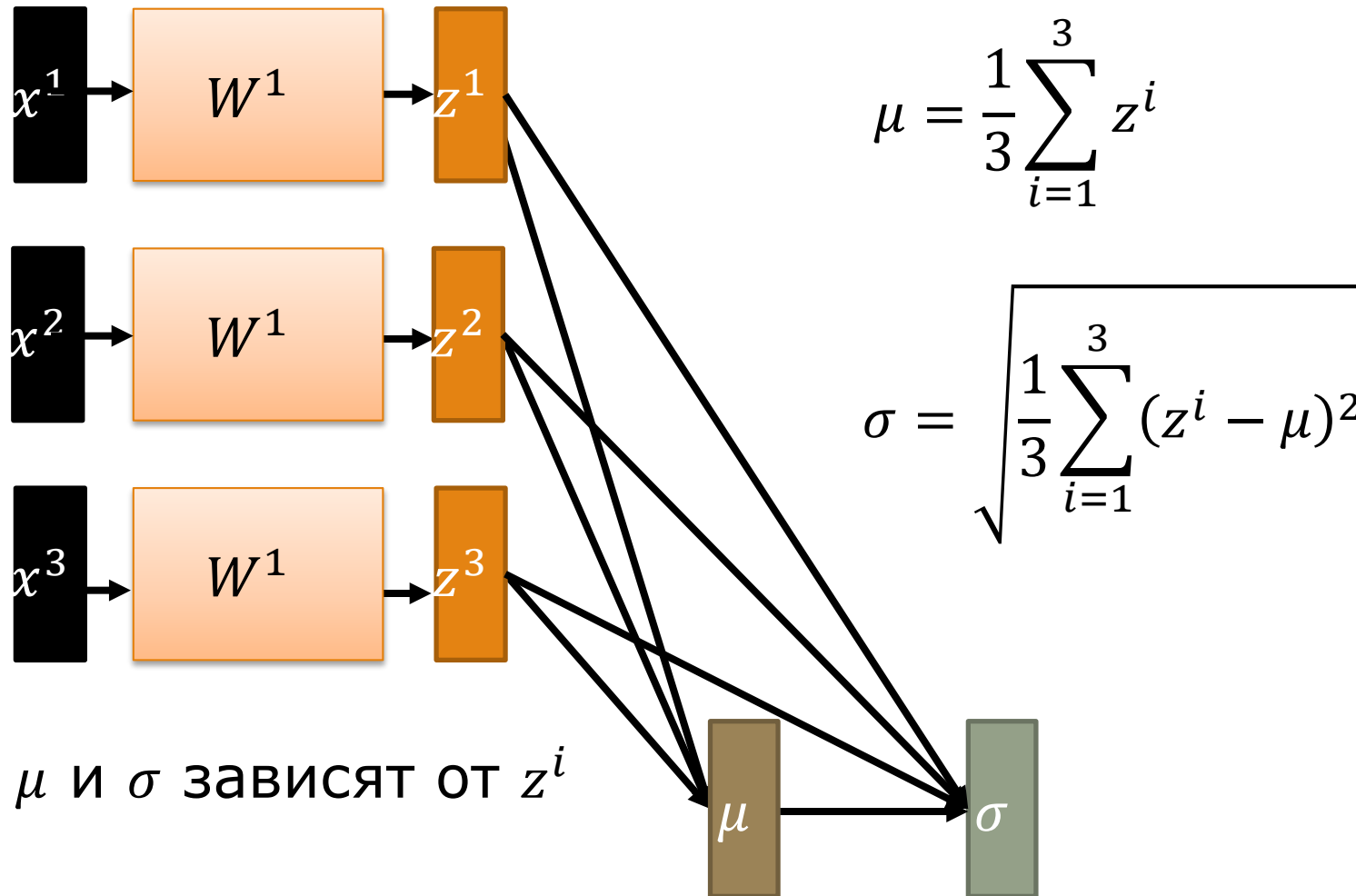
## Партида



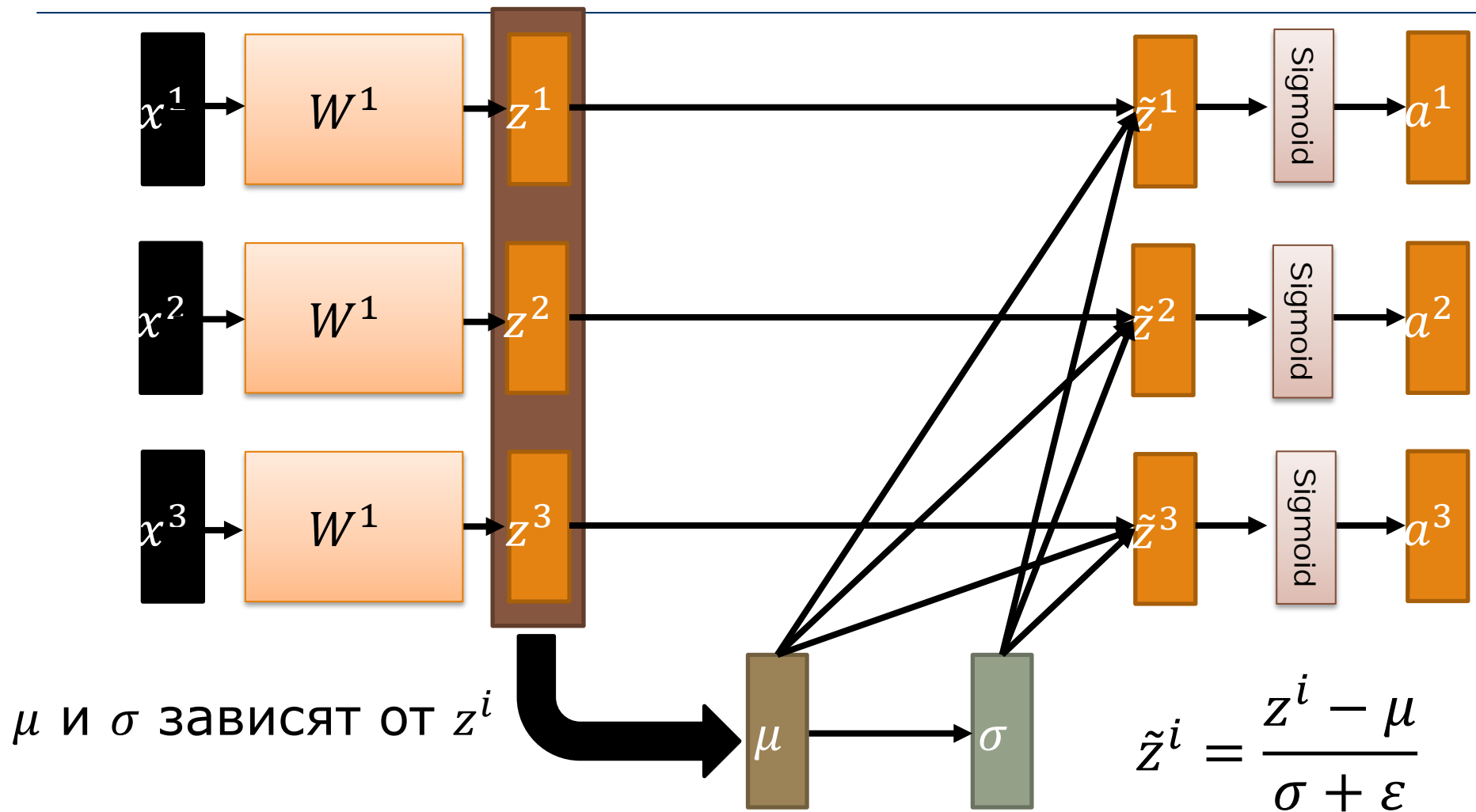
## Партида



## Нормализация на партида



## Нормализация на партида/пакет



Пакетните норми се случват между изчисления  $Z$  и  $A$ . Интуицията е, че вместо ненормализирана стойност  $Z$ , може да се използва нормализирана стойност  $Z$

## Нормализация на партида

---

- Задаване на средната стойност  $\mu = 0$  и  $\sigma = 1$  дава добри резултати при повечето приложения, но в конкретната реализация не желаем невроните от скритите слоеве винаги да имат средна стойност 0 и вариация 1

- $\tilde{z}^i = \frac{z^i - \mu}{\sigma + \varepsilon}$ , го заместваме със следния израз:

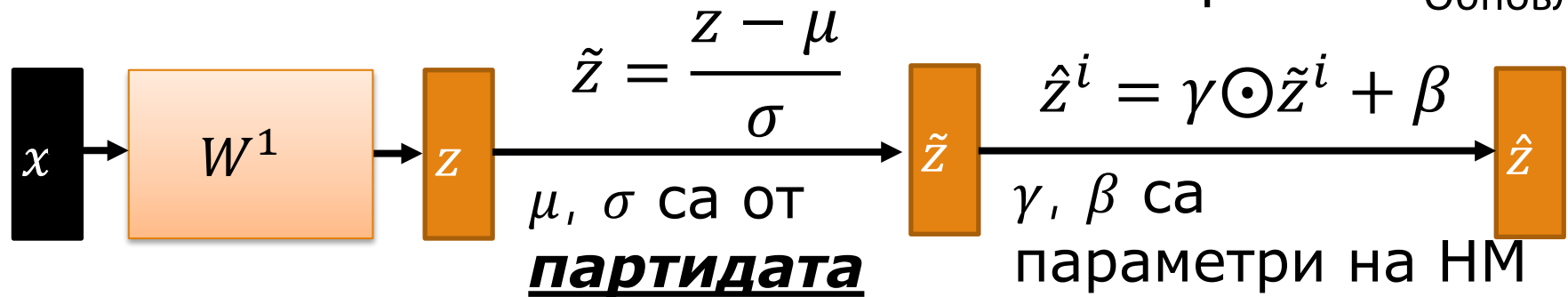
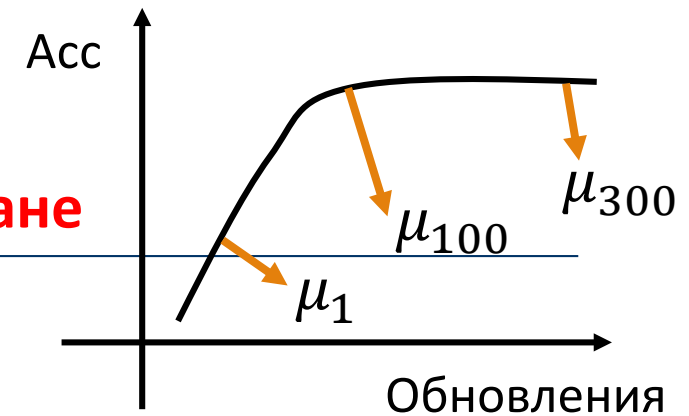
$$\tilde{z}_{norm}^i = \gamma \tilde{z}^i + \beta$$

където  $\gamma$  и  $\beta$  са параметри, които могат да се намерят в процес на обучение.

- $z^i$  представлява специален подслучай на

$$\tilde{z}_{norm}^i = \gamma \tilde{z}^i + \beta \text{ при } \gamma = \sigma + \varepsilon \text{ и } \beta = \mu$$

## Нормализация на партида по време на тестване



Не разполагаме с **партидата** във фазата на тестване.

Идеално решение:

Изчисляват се  $\mu$  и  $\sigma$  използвайки цялата train извадка.

Практическото решение:

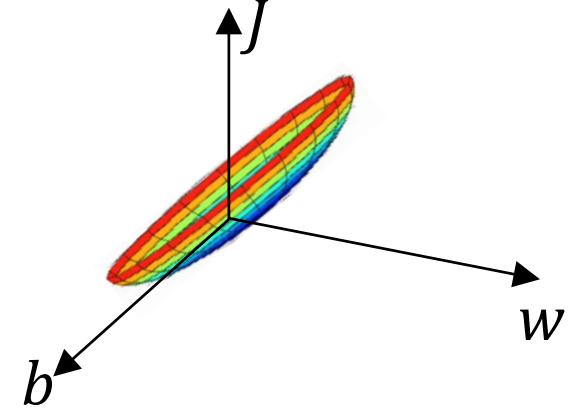
Изчисление на плъзгащата се средна стойност на  $\mu$  и  $\sigma$  от партидите по време на процеса на обучение.

## Защо с нормализацията на данните се ускорява алгоритъма?

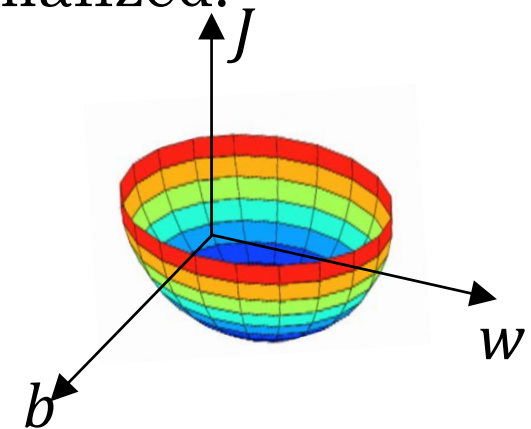
---

- В случай на ненормализирани данни, мащабът на характеристиките ще варира и следователно ще има вариация в параметрите, научени за всяка характеристика. Това ще направи функцията на разходите асиметрична.
- Когато имаме нормализирани данни, мащабът ще бъде същият и функцията на разходите също ще бъде симетрична.
- С това се улеснява алгоритъма за градиентно спускане да може да намира глобалните минимума по-бързо. А това от своя страна довежда до това алгоритъма да работи много по-бързо.

Unnormalized:

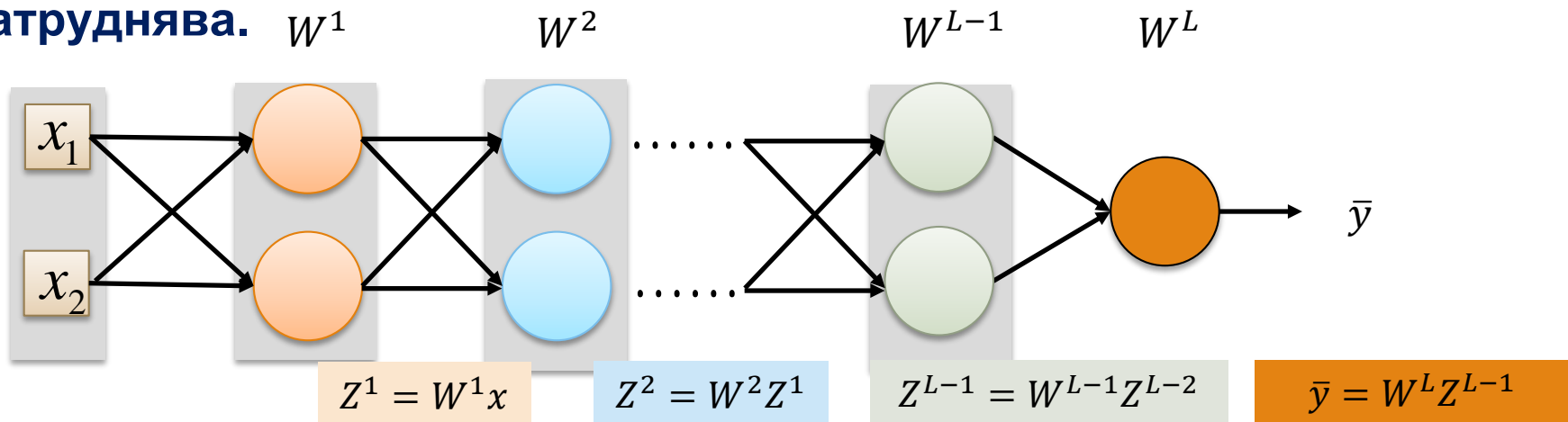


Normalized:



## Изчезващи градиенти

- При обучение на някоя много дълбока НМ, дериватите понякога могат да станат колосални и огромни, с което процеса на обучение се затруднява.



- За упростиране приемаме отместване ( $b = 0$ ) за всеки слой и линейна активационна ф-я
- Да приемем, че записите в матриците на теглата са в следния вид

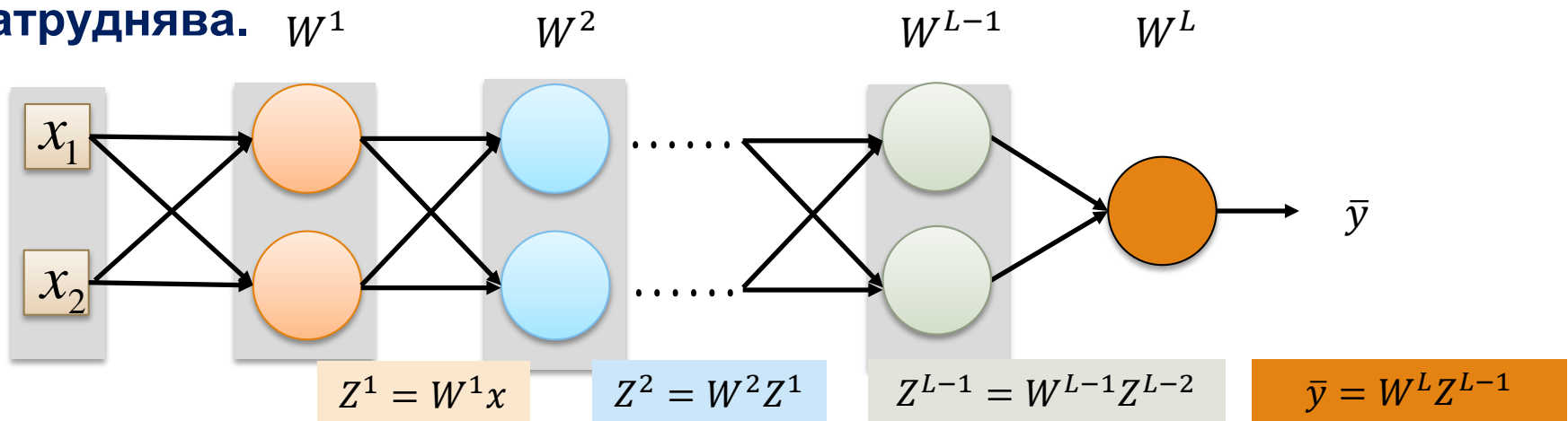
$$W^{L-1} = W^{L-2} = \dots W^2 = W^1 = \begin{bmatrix} p & 0 \\ 0 & p \end{bmatrix} \quad \text{тогава,}$$

$$\begin{aligned} \bar{y} &= W^L Z^{L-1} \\ \bar{y} &= W^L W^{L-1} W^{L-2} \dots W^2 W^1 x \\ \bar{y} &= W^L \times \begin{bmatrix} p & 0 \\ 0 & p \end{bmatrix}^{L-1} \times x \end{aligned}$$



## Изчезващи градиенти

- При обучение на някоя много дълбока НМ, дериватите понякога могат да станат колосални и огромни, с което процеса на обучение се затруднява.



- Ако  $p > 1$  и броя слоеве в НМ е голям, то стойността на  $\bar{y}$  ще изчезне.
- По подобие ако  $p < 1$ , стойността на  $\bar{y}$  ще бъде много малка. Следователно, намаляването на градиента ще се извърши с много малка стъпка.

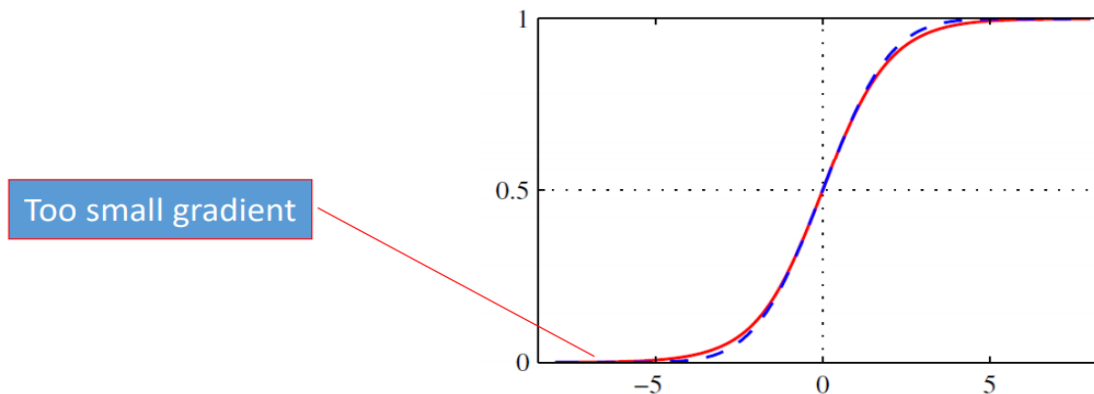
$$\bar{y} = W^L W^{L-1} W^{L-2} \dots W^2 W^1 x$$

$$\bar{y} = W^L \times \begin{bmatrix} p & 0 \\ 0 & p \end{bmatrix}^{L-1} \times x$$

## Решения: Изчезващи градиенти

---

- Да се използва добра и подходяща инициализация
  - Произволна инициализация
    - Главната причина за извикване на произволно избрани теглови коефициенти при инициализация е за да се разбие симетрията.
    - Ние искаме да сме сигурни, че отделните неврони в скритите слоеве научават и запомнят различни последователности.
- Да не се използват сигмоидални активационни функции в дълбоки невронни мрежи
  - Проблем: насищане

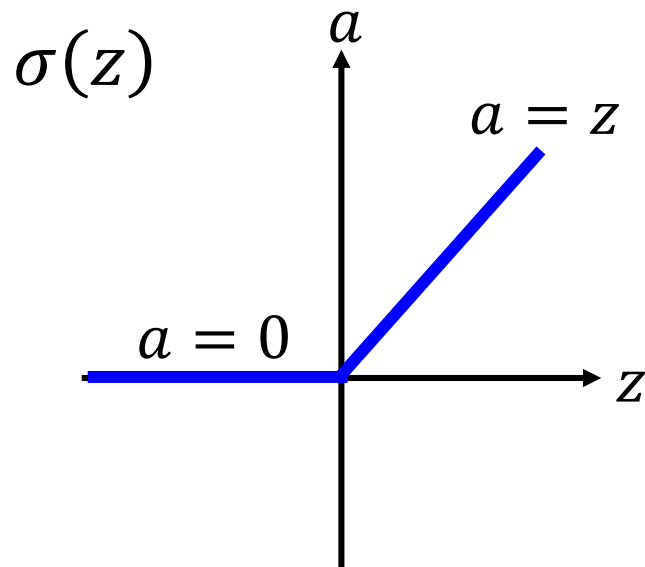
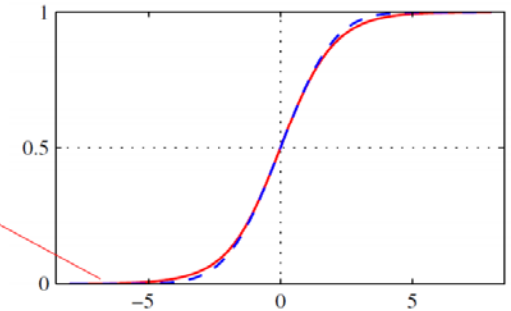


## ReLU

- Ректифициран линеен блок

### Rectified Linear Unit (ReLU)

Too small gradient

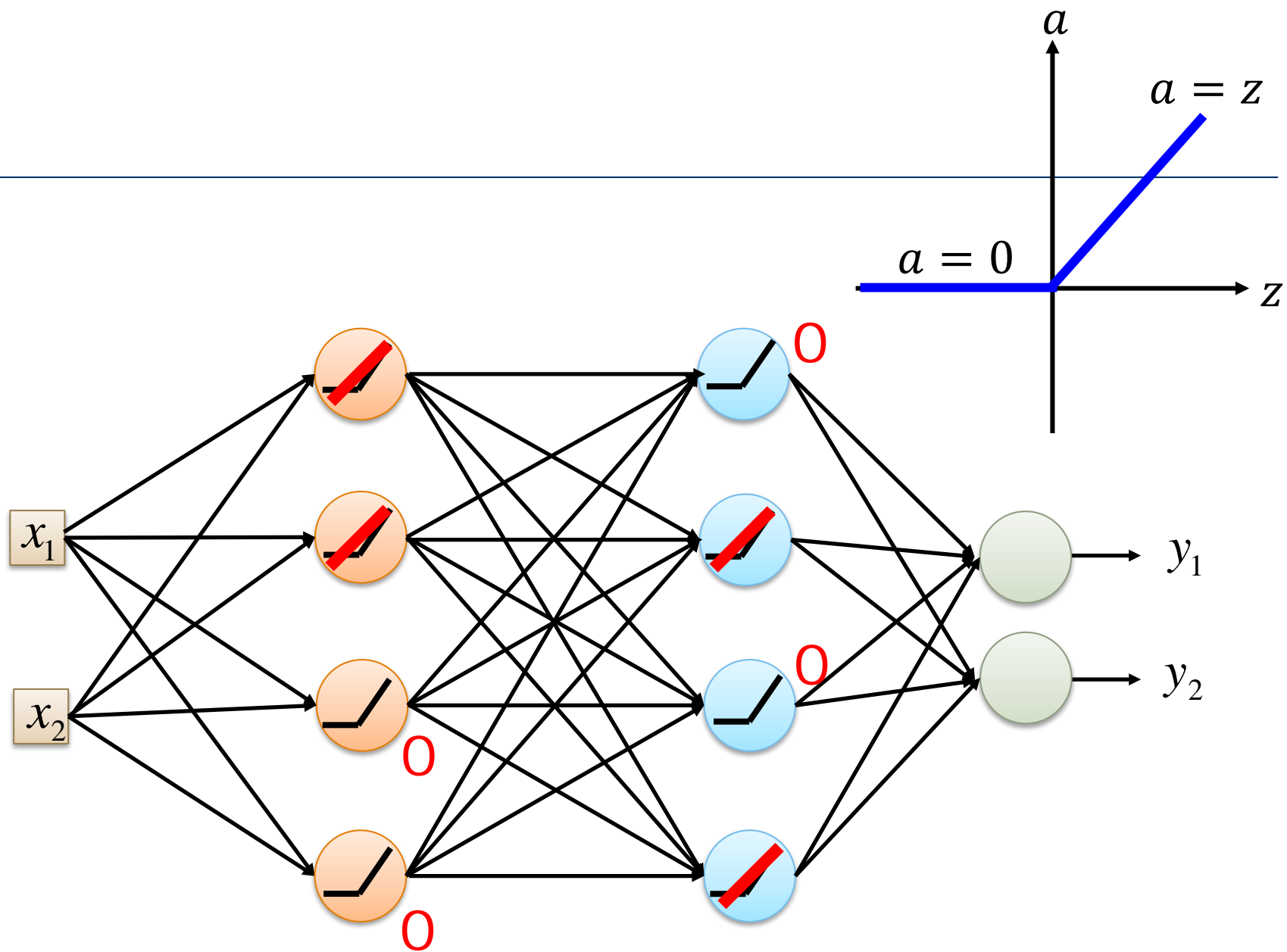


### Защо да се ползва:

1. Бърз за изчисления

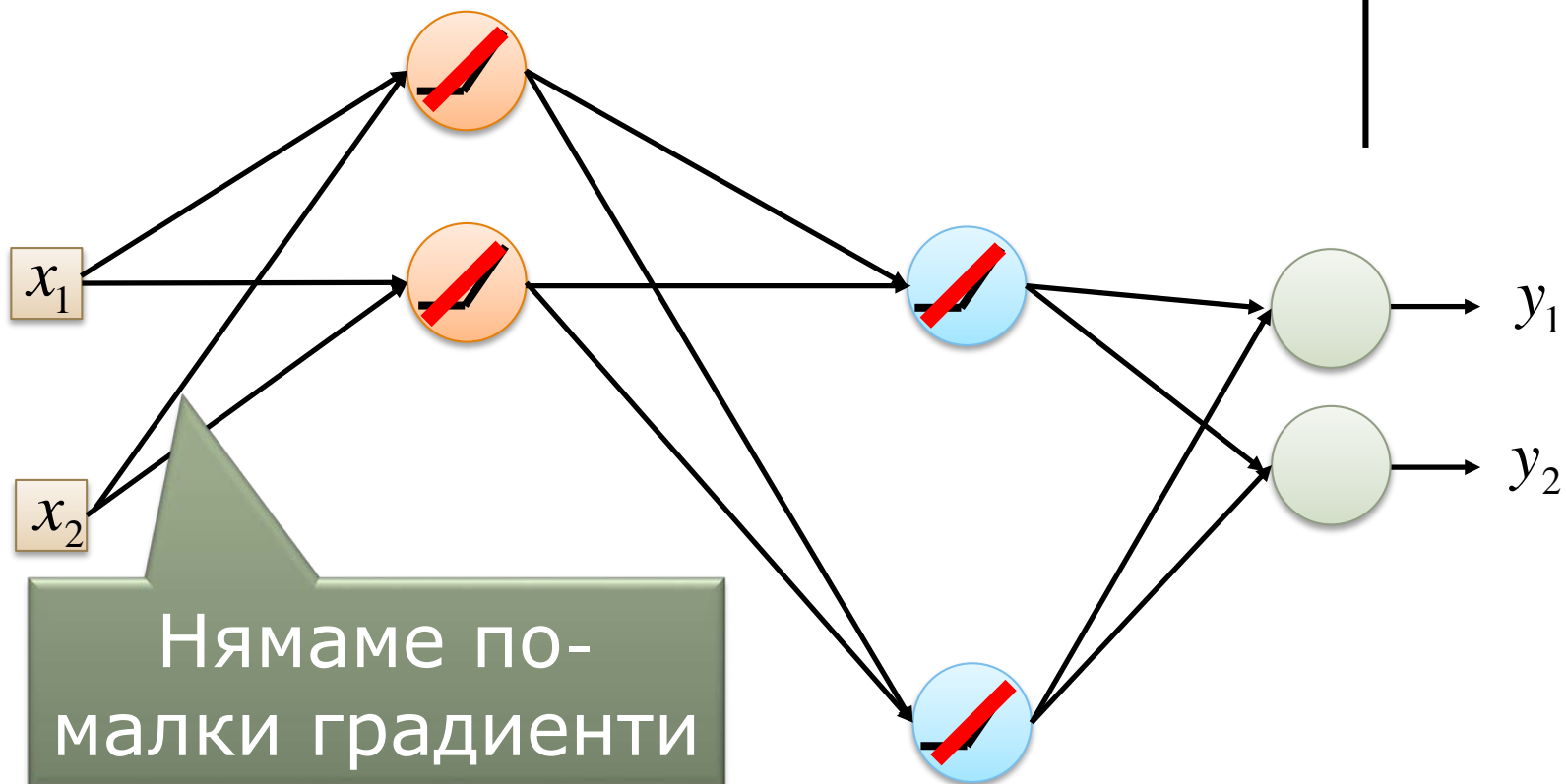
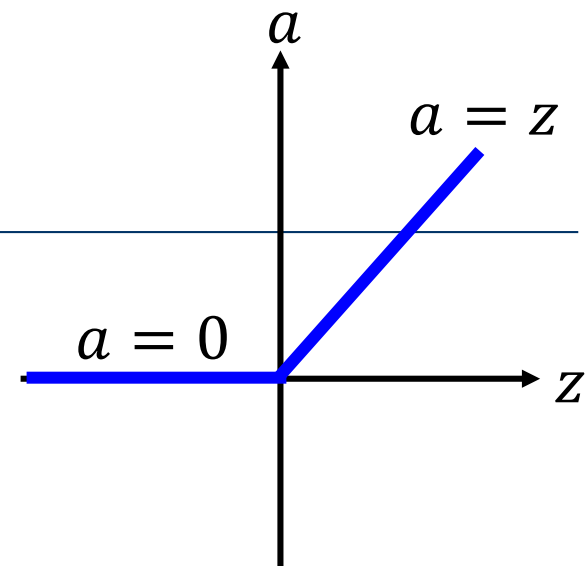
2. Проблема с изчезващия градиент

# ReLU



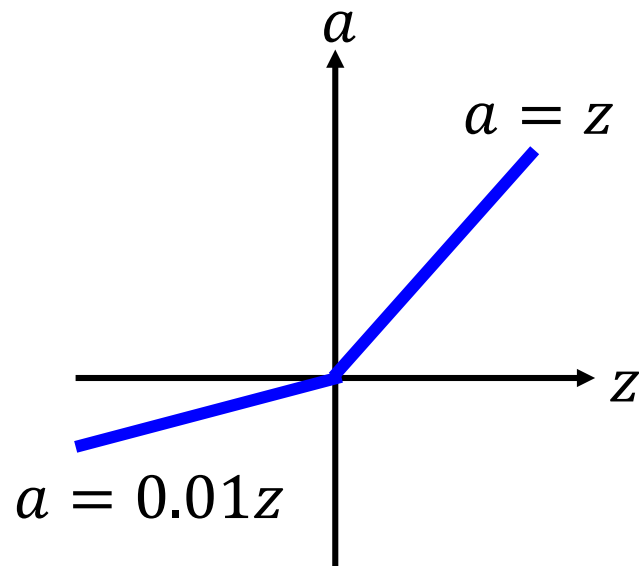
## ReLU

По-тясна линейна НМ

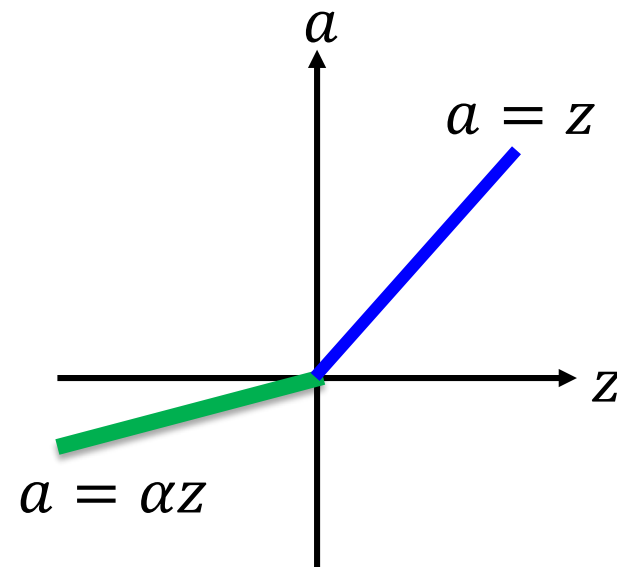


## ReLU - варианти

Изтичащ *ReLU*



Параметричен *ReLU*



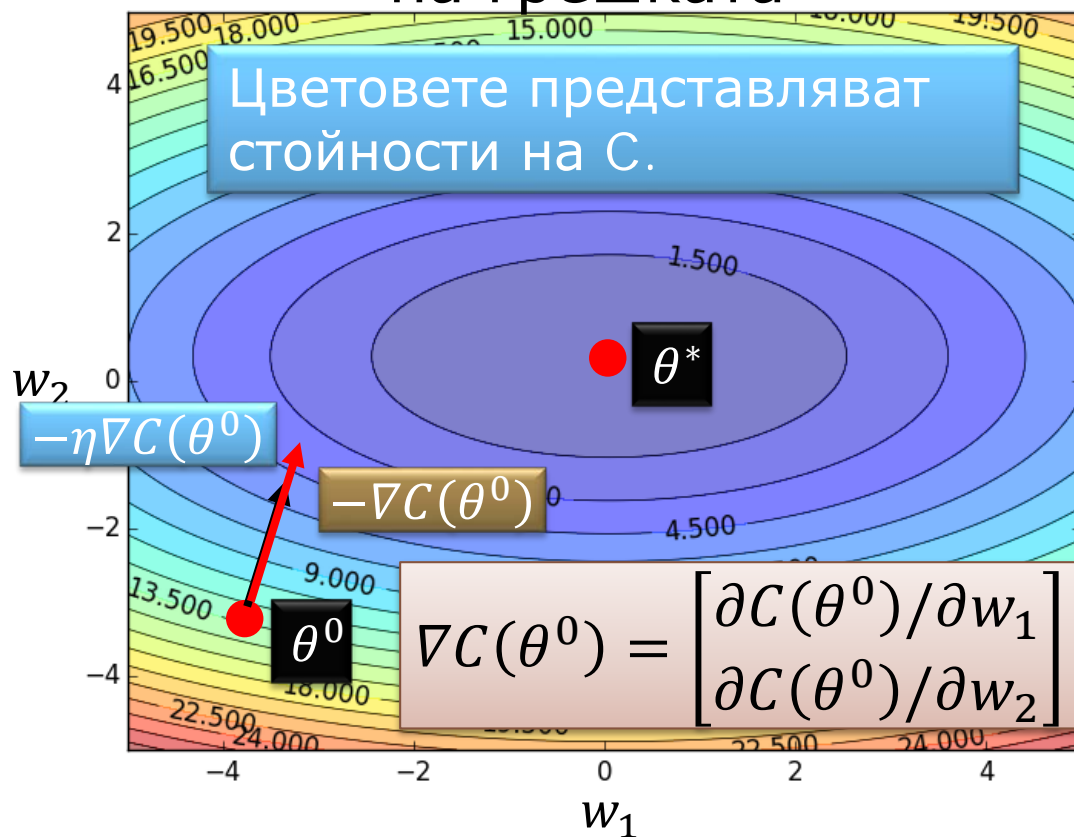
$\alpha$  се научава при намаляващ градиент

# Оптимизация на алгоритмите

## Намаляване на градиента/спускане

Пространство  
на грешката

Да приемем, че имаме само  
два параметъра  $w_1$  и  $w_2$  в  
една НМ.  $\theta = \{w_1, w_2\}$



По произволен начин  
да изберем началната  
точка  $\theta^0$

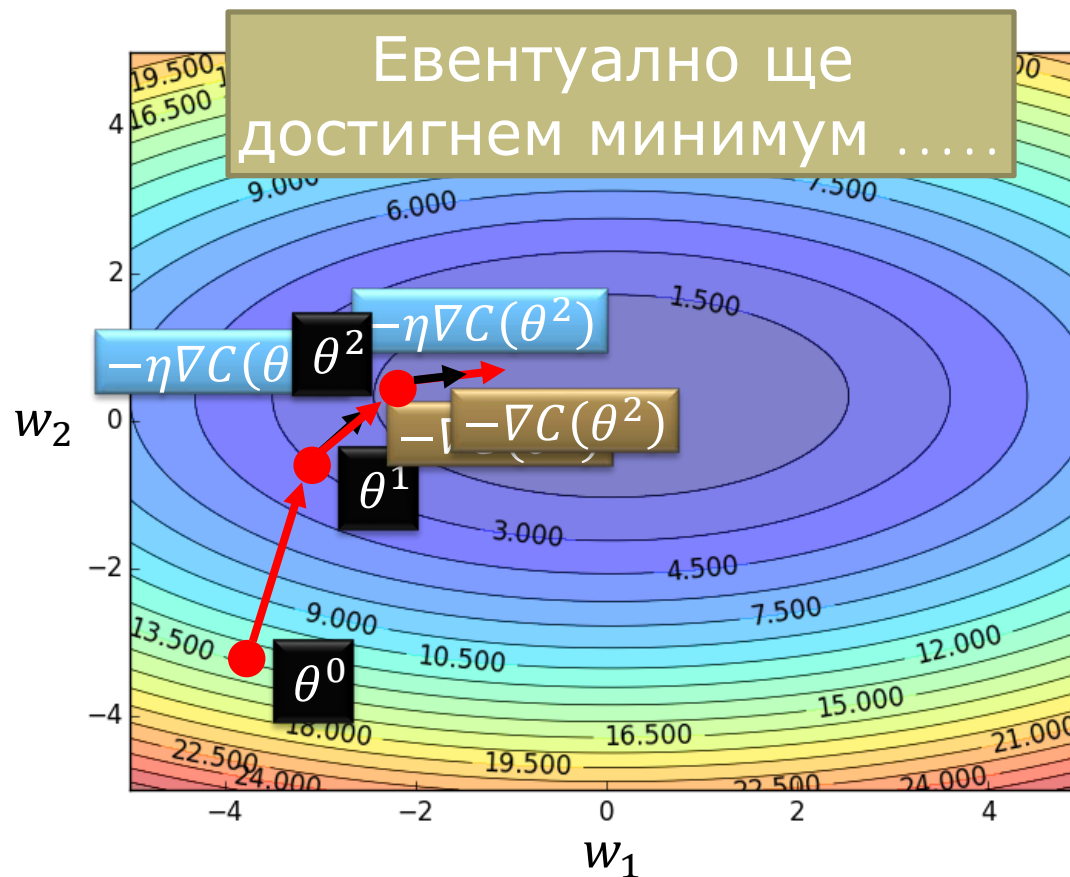
Да се изчисли  
отрицателният  
градиент в  $\theta^0$

➡  $-\nabla C(\theta^0)$

Умножен със скоростта  
на обучение  $\eta$

➡  $-\eta \nabla C(\theta^0)$

## Намаляване на градиента/спускане



По произволен начин  
да изберем началната  
точка  $\theta^0$

Да се изчисли  
отрицателният  
градиент в  $\theta^0$

➡  $-\nabla C(\theta^0)$

Умножен със скоростта  
на обучение  $\eta$

➡  $-\eta \nabla C(\theta^0)$



# Намаляване на градиента/спускане

## ■ Намаляване на градиента/спускане

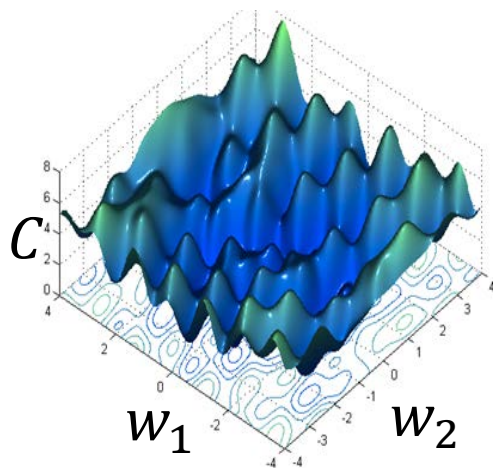
### – Предимства

Гарантирана сходимост към **глобален минимум** при изпъкнала повърхност на грешката  
Сходимост към **локален минимум** при неизпъкнала повърхност на грешката (nonconvex)

### – Недостатъци

Много бавна процедура

Неразрешим проблем за набор от данни, които не се побират в паметта (intractable)



При различни начала  $\theta^0$



Достига се до различни минимуми, респективно и резултати (non-convex)

## Намаляване на градиента/спускане: Практически проблеми

---

При оптимизационна процедура в големи мащаби

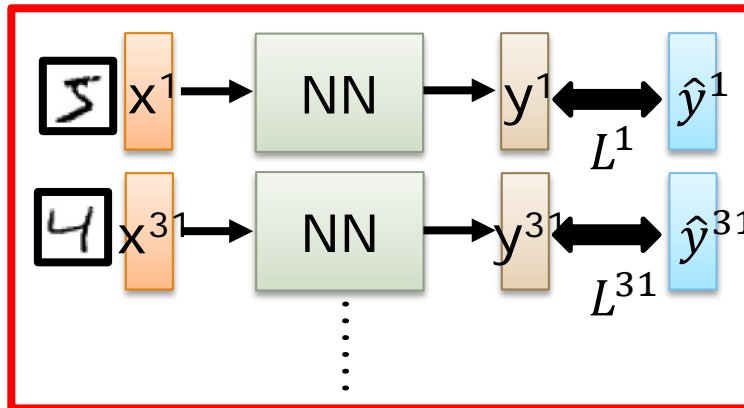
$$h(x) = \frac{1}{N} \sum_{i=1}^N f(x; y_i)$$

Изчисляването на градиента отнема  $O(N)$  време

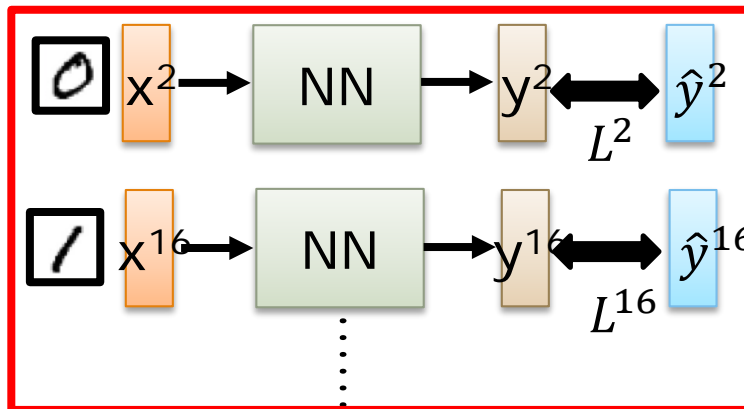
$$\nabla h(x) = \frac{1}{N} \sum_{i=1}^N \nabla f(x; y_i)$$

## Мини-партиди

Мини -  
партида



Мини -  
партида



- Произволно задаване на  $\theta^0$
- Избира се 1<sup>ва</sup> партида  
 $C = L^1 + L^{31} + \dots$   
 $\theta^1 \leftarrow \theta^0 - \eta \nabla C(\theta^0)$
- Избира се 2<sup>ра</sup> партида  
 $C = L^2 + L^{16} + \dots$   
 $\theta^2 \leftarrow \theta^1 - \eta \nabla C(\theta^1)$   
 $\vdots$

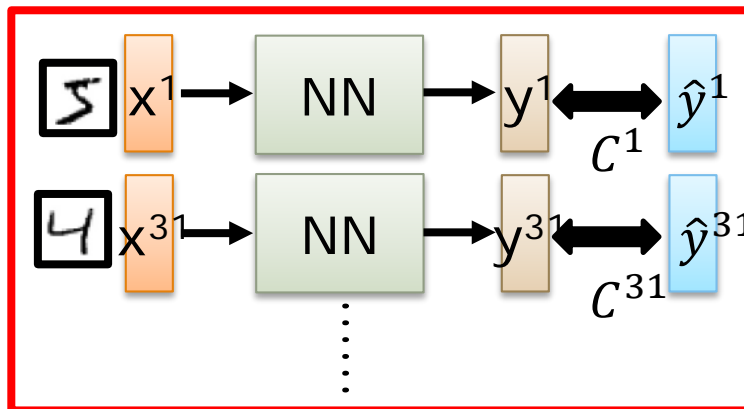
С е различно  
всеки път, когато  
обновяваме  
параметрите!

## Мини-партиди

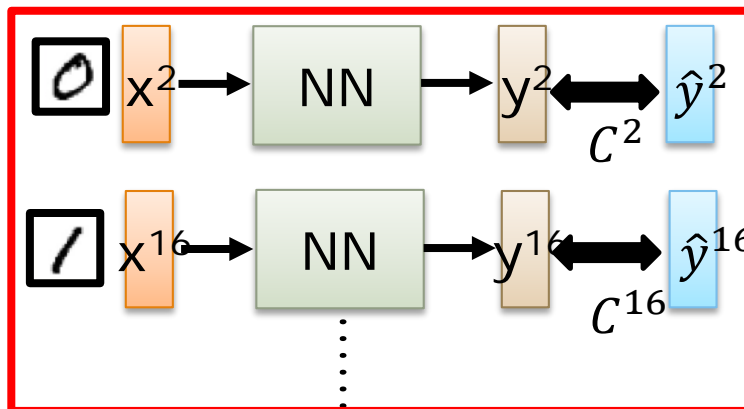
По-бързо

По-добро!

Мини -  
партида



Мини -  
партида



➤ Произволно задаване на  $\theta^0$

➤ Избира се 1<sup>ва</sup> партида

$$C = C^1 + C^{31} + \dots$$

$$\theta^1 \leftarrow \theta^0 - \eta \nabla C(\theta^0)$$

➤ Избира се 2<sup>ра</sup> партида

$$C = C^2 + C^{16} + \dots$$

$$\theta^2 \leftarrow \theta^1 - \eta \nabla C(\theta^1)$$

⋮

➤ Докато не се изберат всички мини-партиди

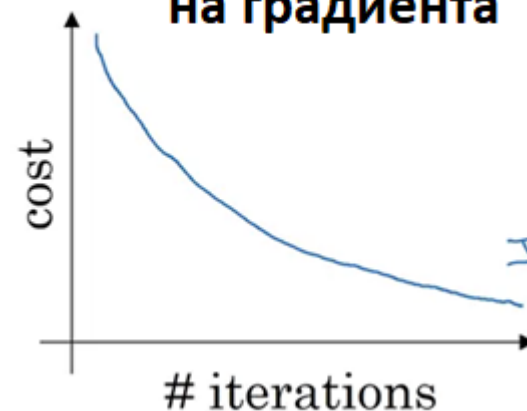
Една епоха

Повтори горния процес

## По какъв начин можем да изберем размера на Мини-партидите?

- Ако размерът на мини партидата =  $m$ 
  - Представява пакетно намаляне на градиента, където всички примери за обучение се използват във всяка итерация. Отнема твърде много време на итерация.
- Ако размерът на мини партидата = 1
  - Нарича се стохастично градиентно спускане, където всеки пример за обучение е своя собствена мини партида.
  - Тъй като във всяка итерация вземаме само един пример, той може да стане изключително зашумен и да отнеме много повече време, за да достигне до глобалните минимума
- Ако размерът на мини партидата е между 1 и  $m$ 
  - Това представлява мини-партидното спускане на градиента. Размерът на мини-партидата не трябва да е твърде голям или твърде малък.

Пакетно намаляне на градиента



Намаляне на градиента с мини-партиди

