Търсене и извличане на информация. Приложение на дълбоко машинно обучение

Стоян Михов

Лекция 8: Логистична регресия. Невронни мрежи. Многослойни перцептрони.

План на лекцията

1. **Формалности за курса (5 мин)**
2. Логистична регресия (15 мин)
3. Обучение чрез спускане по градиента (15 мин)
4. Логистична регресия при много класове (15 мин)
5. Изкуствени невронни мрежи (10 мин)
6. Представимост на функции с невронни мрежи (10 мин)
7. Многослойни перцептрони (15 мин)

Формалности

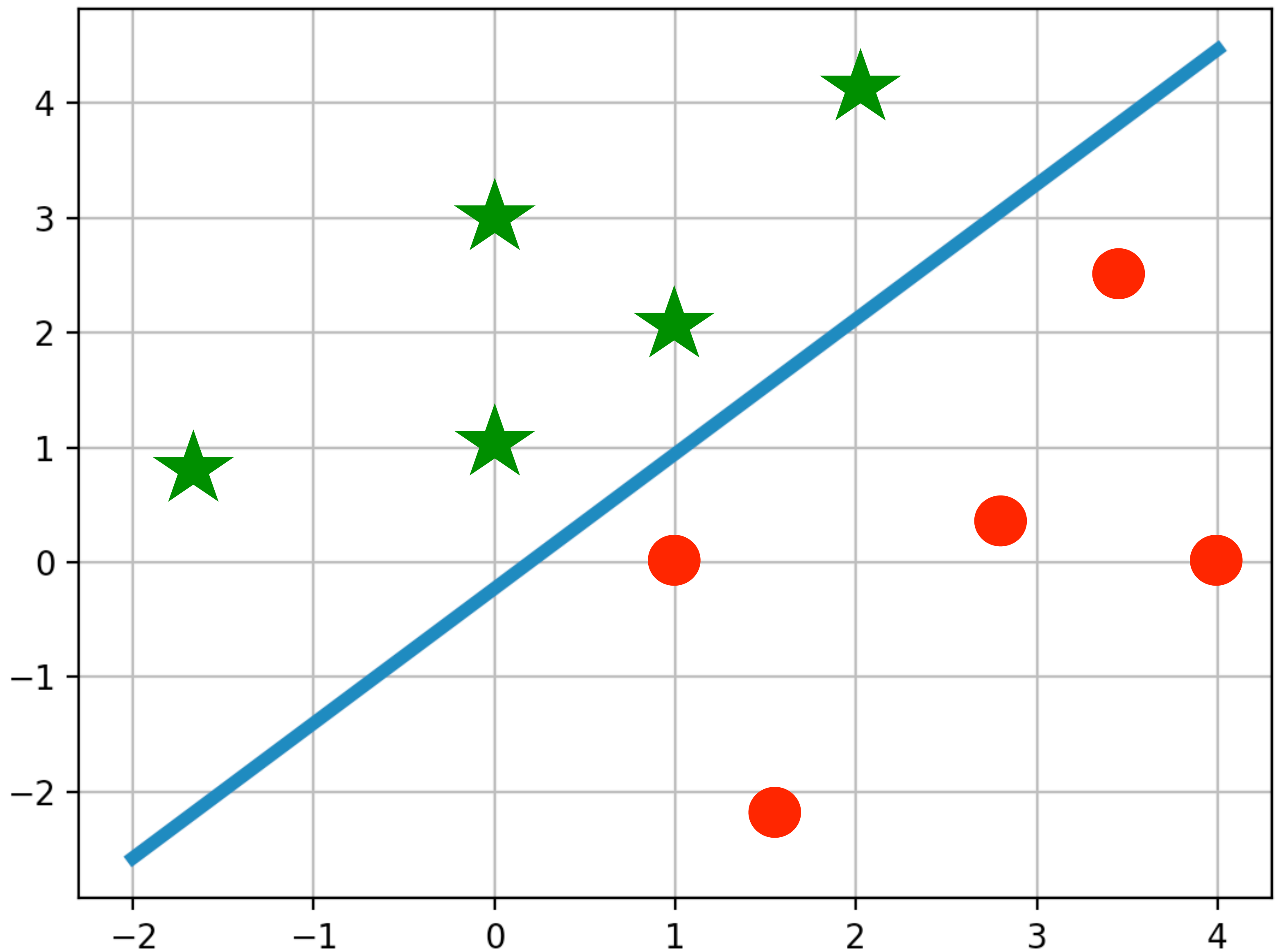
- Засега ще провеждаме занятията онлайн всяка сряда от 8:15 до 12:00 часа.
- Засега ще използваме платформата Google meet: meet.google.com/hue-frfx-axb
- Днес ще използваме едновременно слайдове и бяла дъска. Моля следете съответния екран.
- Домашното задание следва да бъде предадено до 10 часа на на 30.11.2020г.
- Осмата лекция се базира на глави 2, 3 и 4 от втория учебник.

План на лекцията

1. Формалности за курса (5 мин)
2. **Логистична регресия (15 мин)**
3. Обучение чрез спускане по градиента (15 мин)
4. Логистична регресия при много класове (15 мин)
5. Изкуствени невронни мрежи (10 мин)
6. Представимост на функции с невронни мрежи (10 мин)
7. Многослойни перцептрони (15 мин)

Линеен класификатор

$$\gamma(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$$



Проблеми при дискретната класификация

- В практиката рядко можем да класифицираме нещата в две крайности — черно или бяло.
- Ако наблюдението е близо до разделителната хиперравнина нашата увереност в класификацията би следвало да е по-ниска.
- Колкото по-далеч е наблюдението от разделителната хиперравнина, толкова по уверени можем да бъдем в правилността на класификацията.
- Желателно е класификатора да върне степен на увереност в класификацията.
- Вместо увереност е по-удобно да върнем вероятност.

Вероятностен линеен класификатор — логистична регресия

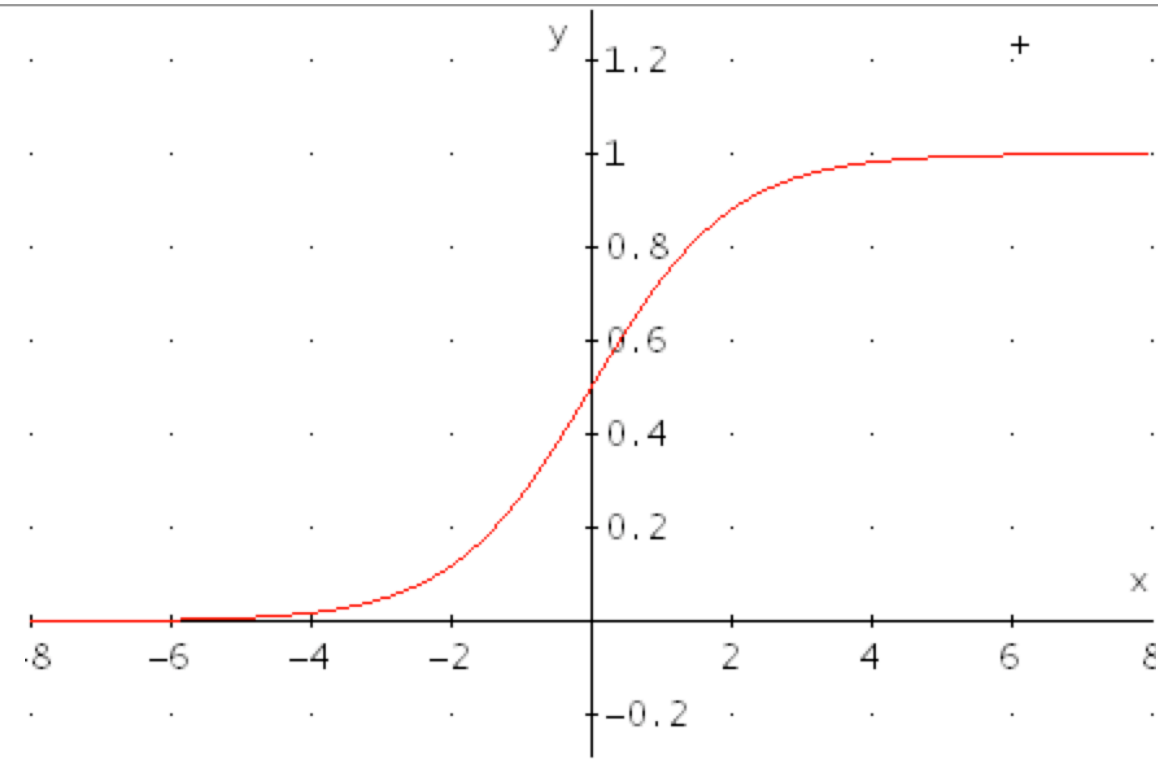
- Разглеждаме бинарен класификатор с класове $\mathscr{Y} = \{0,1\}$.
- **Задача:** Разстоянието на вектора $\mathbf{x}$ до разделителната хиперравнина $\mathbf{w}^\top \mathbf{u} + b = 0$ е $\dfrac{|\mathbf{w}^\top \mathbf{x} + b|}{\|\mathbf{w}\|}$ (докажете го).
- Нека с $y \in \mathscr{Y}$ означим класа на наблюдението $\mathbf{x}$. Дефинираме:

$$\Pr_{\mathbf{w},b}[y = 1 \,|\, \mathbf{x}] = \sigma(\mathbf{w}^\top \mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^\top \mathbf{x} + b)}}$$

$$\Pr_{\mathbf{w},b}[y = 0 \,|\, \mathbf{x}] = 1 - \sigma(\mathbf{w}^\top \mathbf{x} + b) = 1 - \frac{1}{1 + e^{-(\mathbf{w}^\top \mathbf{x} + b)}}$$

- Използвайки логистичната функция (сигмоид) $\sigma(z) = \dfrac{1}{1 + e^{-z}}$, бинарните предсказания прекарани през лог-линейното преобразувание интерпретираме като вероятности за принадлежност.

Сигмоид — логистичната функция



- $\sigma(z) = \dfrac{1}{1+e^{-z}} = \dfrac{e^z}{1+e^z},$
- $1 - \sigma(z) = \dfrac{e^{-z}}{1+e^{-z}}$
- $(\sigma(z))' = \dfrac{e^{-z}}{(1+e^{-z})^2} = e^{-z}\sigma^2(z) = (1-\sigma(z))\sigma(z)$
- $(\log\sigma(z))' = (\log 1 - log(1+e^{-z}))' = -\dfrac{-e^{-z}}{1+e^{-z}} = 1 - \sigma(z)$
- $(\log(1-\sigma(z)))' = (\log e^{-z} - log(1+e^{-z}))' = -1 + (1-\sigma(z)) = -\sigma(z)$

Целева функция

- Дадена е извадка от наблюдения заедно със съответните им етикети $X = ((\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})), (\mathbf{x}^{(i)}, y^{(i)}) \in \mathbb{R}^N \times \{0,1\}$.

- За дадена права $\mathbf{w}^\top \mathbf{x} + b = 0, \mathbf{w} \in \mathbb{R}^N, b \in \mathbb{R}$, правдоподобието е:
  $L_{\mathbf{w},b}((\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})) = \prod_{i=1}^{m} \Pr_{\mathbf{w},b}[y = y^{(i)} \,|\, \mathbf{x}^{(i)}]$, където
  $\Pr_{\mathbf{w},b}[y = 1 \,|\, \mathbf{x}^{(i)}] = \sigma(\mathbf{w}^\top \mathbf{x}^{(i)} + b)$ и
  $\Pr_{\mathbf{w},b}[y = 0 \,|\, \mathbf{x}^{(i)}] = 1 - \sigma(\mathbf{w}^\top \mathbf{x}^{(i)} + b)$

- Целта е да намерим параметрите, при които се максимизира правдоподобието:
  $\hat{\mathbf{w}}, \hat{b} = \arg\max_{\mathbf{w},b} L_{\mathbf{w},b}((\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})) = \arg\max_{\mathbf{w},b} \prod_{i=1}^{m} \Pr_{\mathbf{w},b}[y = y^{(i)} \,|\, \mathbf{x}^{(i)}]$

Максимизиране на правдоподобието = минимизиране на кросентропията

$$\hat{\mathbf{w}}, \hat{b} = \arg\max_{\mathbf{w},b} \prod_{i=1}^{m} \Pr_{\mathbf{w},b}[y = y^{(i)} \,|\, \mathbf{x}^{(i)}] =$$

$$= \arg\max_{\mathbf{w},b} \log \prod_{i=1}^{m} \Pr_{\mathbf{w},b}[y = y^{(i)} \,|\, \mathbf{x}^{(i)}] =$$

$$= \arg\min_{\mathbf{w},b} - \log \prod_{i=1}^{m} \Pr_{\mathbf{w},b}[y = y^{(i)} \,|\, \mathbf{x}^{(i)}] =$$

$$= \arg\min_{\mathbf{w},b} - \frac{1}{m} \sum_{i=1}^{m} \log \Pr_{\mathbf{w},b}[y = y^{(i)} \,|\, \mathbf{x}^{(i)}] =$$

$$= \arg\min_{\mathbf{w},b} H_X(\Pr \| \Pr_{\mathbf{w},b})$$

План на лекцията

1. Формалности за курса (5 мин)
2. Логистична регресия (15 мин)
3. **Обучение чрез спускане по градиента (15 мин)**
4. Логистична регресия при много класове (15 мин)
5. Изкуствени невронни мрежи (10 мин)
6. Представимост на функции с невронни мрежи (10 мин)
7. Многослойни перцептрони (15 мин)

Обучение чрез спускане по градиента

- При по-сложни функции аналитичното намиране на параметрите, при които се минимизира кросентропията, е невъзможно.

- Нека е дадена целева функция $J(\theta)$, където параметрите, по които минимизираме, са θ, която е частично-диференцируема отностно θ.

- **Спускането по градиента** е следния итеративен алгоритъм:

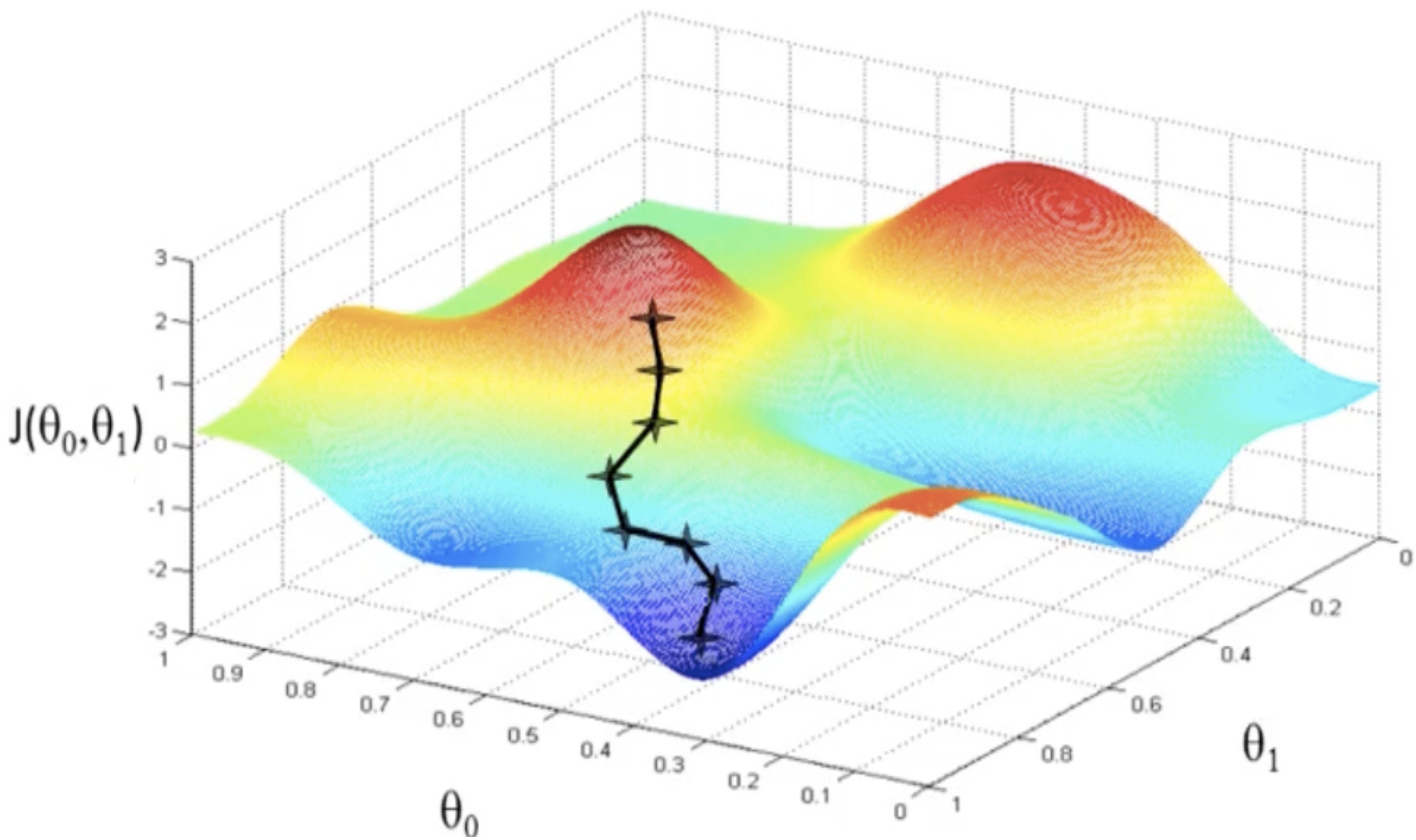
    1. Започваме с начална стойност на параметрите θ_0.

    2. На стъпка $i+1$ намираме: $\theta_{i+1} = \theta_i - \alpha \frac{\partial}{\partial \theta} J(\theta_i)$.

    3. Повтаряме стъпки 2-3 докато не удовлетворим условие за край.

- Параметърът α наричаме **скорост на учене**. Той оказва съществено значение за броя на итерациите и намирането на минимум.

Илюстрация за спускането по градиента



Намиране на градиента при логистичната регресия

$$\frac{\partial}{\partial b}\log \Pr_{\mathbf{w},b}[y=1\,|\,\mathbf{x}] = \frac{\partial}{\partial b}\log \sigma(\mathbf{w}^\top\mathbf{x}+b) =$$

- $$= (1-\sigma(\mathbf{w}^\top\mathbf{x}+b))\frac{\partial}{\partial b}(\mathbf{w}^\top\mathbf{x}+b) = 1-\sigma(\mathbf{w}^\top\mathbf{x}+b)$$

$$\frac{\partial}{\partial b}\log \Pr_{\mathbf{w},b}[y=0\,|\,\mathbf{x}] = \frac{\partial}{\partial b}(1-\log \sigma(\mathbf{w}^\top\mathbf{x}+b)) =$$

- $$= -\,\sigma(\mathbf{w}^\top\mathbf{x}+b)\frac{\partial}{\partial b}(\mathbf{w}^\top\mathbf{x}+b) = -\,\sigma(\mathbf{w}^\top\mathbf{x}+b)$$

- Следователно: $\frac{\partial}{\partial b}\log \Pr_{\mathbf{w},b}[y=y^{(i)}\,|\,\mathbf{x}^{(i)}] = y^{(i)}-\sigma(\mathbf{w}^\top\mathbf{x}^{(i)}+b)$

$$\frac{\partial}{\partial \mathbf{w}} \log \Pr_{\mathbf{w},b}[y = 1 \,|\, \mathbf{x}] = \frac{\partial}{\partial \mathbf{w}} \log \sigma(\mathbf{w}^\top \mathbf{x} + b) =$$

- $$= (1 - \sigma(\mathbf{w}^\top \mathbf{x} + b))\frac{\partial}{\partial \mathbf{w}}(\mathbf{w}^\top \mathbf{x} + b) = (1 - \sigma(\mathbf{w}^\top \mathbf{x} + b))\mathbf{x}$$

$$\frac{\partial}{\partial \mathbf{w}} \log \Pr_{\mathbf{w},b}[y = 0 \,|\, \mathbf{x}] = \frac{\partial}{\partial \mathbf{w}} \log(1 - \sigma(\mathbf{w}^\top \mathbf{x} + b)) =$$

- $$= -\,\sigma(\mathbf{w}^\top \mathbf{x} + b)\frac{\partial}{\partial \mathbf{w}}(\mathbf{w}^\top \mathbf{x} + b) = -\,\sigma(\mathbf{w}^\top \mathbf{x} + b)\mathbf{x}$$

- Следодателно:
$$\frac{\partial}{\partial \mathbf{w}} \log \Pr_{\mathbf{w},b}[y = y^{(i)} \,|\, \mathbf{x}^{(i)}] = (y^{(i)} - \sigma(\mathbf{w}^\top \mathbf{x}^{(i)} + b))\mathbf{x}^{(i)}$$

Градиент на логистична регресия

- $$\frac{\partial}{\partial b} H_X(\Pr \| \Pr_{\mathbf{w},b}) = -\frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - \sigma(\mathbf{w}^\top \mathbf{x}^{(i)} + b))$$
- $$\frac{\partial}{\partial \mathbf{w}} H_X(\Pr \| \Pr_{\mathbf{w},b}) = -\frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - \sigma(\mathbf{w}^\top \mathbf{x}^{(i)} + b))\mathbf{x}^{(i)}$$

План на лекцията

1. Формалности за курса (5 мин)
2. Логистична регресия (15 мин)
3. Обучение чрез спускане по градиента (15 мин)
4. **Логистична регресия при много класове (15 мин)**
5. Изкуствени невронни мрежи (10 мин)
6. Представимост на функции с невронни мрежи (10 мин)
7. Многослойни перцептрони (15 мин)

Логистична регресия при много класове

- Разглеждаме класификатор при класове $\mathscr{Y} = \{1,2,\ldots,K\}$.
- Можем да разглеждаме K ,разделителни хиперравнини $\mathbf{w}_c^\top \mathbf{x} + b_c = 0$.
- Дадена е извадка от наблюдения заедно със съответните им етикети $X = ((\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})), (\mathbf{x}^{(i)}, y^{(i)}) \in \mathbb{R}^N \times \{1,2,\ldots,K\}$
- Дефинираме $W \in \mathbb{R}^{K\times N}, W = \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_K \end{bmatrix}$, $\mathbf{b} \in \mathbb{R}^K, \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_K \end{bmatrix}$
- $\Pr_{W,\mathbf{b}}[y = c \,|\, \mathbf{x}] = \operatorname{softmax}(W\mathbf{x} + \mathbf{b})_c = \dfrac{e^{(W\mathbf{x}+\mathbf{b})_c}}{\sum_{j=1}^{K} e^{(W\mathbf{x}+\mathbf{b})_j}}$
- Обучаваме модела, като минимизираме кросентропията $H_X[\Pr \| \Pr_{W,\mathbf{b}}]$.
- **Задача**: Покажете, че при бинарен модел $\operatorname{softmax}$ е еквивалентен на сигмоид.

Верижно правило за диференциране на композиция на функции на много променливи

- Нека $f : \mathbb{R}^m \to \mathbb{R}$, $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$, $\mathbf{x} \in \mathbb{R}^n$

- Тогава:
$$\frac{\partial}{\partial x_k} f(\mathbf{g}(\mathbf{x})) = \frac{\partial}{\partial x_k} f(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n)) = \sum_{j=1}^{m} \frac{\partial f}{\partial g_j} \frac{\partial g_j}{\partial x_k}$$

- Векторен запис с използване на **якобияни**:
$$\frac{\partial}{\partial x_k} f(\mathbf{g}(\mathbf{x})) = \left(\frac{\partial f}{\partial \mathbf{g}}\right)^\mathsf{T} \frac{\partial \mathbf{g}}{\partial x_k}, \text{ тук } \frac{\partial f}{\partial \mathbf{g}}, \frac{\partial \mathbf{g}}{\partial x_k} \in \mathbb{R}^m,$$
$$\frac{\partial}{\partial \mathbf{x}} f(\mathbf{g}(\mathbf{x})) = \left(\frac{\partial f}{\partial \mathbf{g}}\right)^\mathsf{T} \frac{\partial \mathbf{g}}{\partial \mathbf{x}}, \text{ тук } \frac{\partial \mathbf{g}}{\partial \mathbf{x}} \in \mathbb{R}^{m \times n}.$$

$$\arg\min_{W,\mathbf{b}} H_X(\Pr \| \Pr_{W,\mathbf{b}}) = \arg\min_{W,\mathbf{b}} -\frac{1}{m}\sum_{i=1}^{m} \log \Pr_{W,\mathbf{b}}[y = y^{(i)} \mid \mathbf{x}^{(i)}] =$$

$$= \arg\min_{W,\mathbf{b}} -\frac{1}{m}\sum_{i=1}^{m} \log \operatorname{softmax}(W\mathbf{x}^{(i)} + \mathbf{b})_{y^{(i)}} =$$

$$= \arg\min_{W,\mathbf{b}} -\frac{1}{m}\sum_{i=1}^{m} \log \frac{e^{(W\mathbf{x}^{(i)}+\mathbf{b})_{y^{(i)}}}}{\sum_{j=1}^{K} e^{(W\mathbf{x}^{(i)}+\mathbf{b})_j}}$$

$$\frac{\partial}{\partial u_k} \log \operatorname{softmax}(\mathbf{u})_y = \frac{\partial}{\partial u_k} \log \frac{e^{u_y}}{\sum_{j=1}^{K} e^{u_j}} = \frac{\partial}{\partial u_k} u_y - \frac{\partial}{\partial u_k} \log \sum_{j=1}^{K} e^{u_j} =$$

$$= \delta_{k=y} - \frac{1}{\sum_{j=1}^{K} e^{u_j}} \frac{\partial}{\partial u_k} \sum_{j=1}^{K} e^{u_j} = \delta_{k=y} - \frac{e^{u_k}}{\sum_{j=1}^{K} e^{u_j}} =$$

$$= \delta_{k=y} - \operatorname{softmax}(\mathbf{u})_k$$

$$\frac{\partial}{\partial \mathbf{u}} \log \operatorname{softmax}(\mathbf{u})_y = \bar{\delta}_y - \operatorname{softmax}(\mathbf{u}), \text{ където } \bar{\delta}_y \in \mathbb{R}^K, (\bar{\delta}_y)_k = \delta_{k=y}$$

$$\frac{\partial}{\partial \mathbf{b}} \log \frac{e^{(W\mathbf{x}+\mathbf{b})_y}}{\sum_{j=1}^{K} e^{(W\mathbf{x}+\mathbf{b})_j}} = (\bar{\delta}_y - \mathrm{softmax}(W\mathbf{x}+\mathbf{b}))^\top \left(\frac{\partial}{\partial \mathbf{b}} (W\mathbf{x}+\mathbf{b}) \right) =$$

$$= (\bar{\delta}_y - \mathrm{softmax}(W\mathbf{x}+\mathbf{b}))^\top \mathbf{I} =$$

$$= \bar{\delta}_y - \mathrm{softmax}(W\mathbf{x}+\mathbf{b})$$

Радзглеждаме функцията: $u : \mathbb{R}^{KN} \to \mathbb{R}^{K}, u(W) = W\mathbf{x} + \mathbf{b}$. Якобиянът $\dfrac{\partial \mathbf{u}}{\partial W}$ е матрица $\mathbb{R}^{K\times KN}$.

$$\left(\frac{\partial \mathbf{u}}{\partial W_{p,q}} \right)_k = \frac{\partial \mathbf{u}_k}{\partial W_{p,q}} = \frac{\partial \sum_{l=1}^{N} W_{k,l} x_l}{\partial W_{p,q}} = \begin{cases} 0 & \text{if } k \neq p \\ x_q & \text{if } k = p \end{cases} = \delta_{p=k} x_q$$

$$\frac{\partial}{\partial W} \log \frac{e^{(W\mathbf{x}+\mathbf{b})_y}}{\sum_{j=1}^{K} e^{(W\mathbf{x}+\mathbf{b})_j}} = (\bar{\delta}_y - \mathrm{softmax}(W\mathbf{x}+\mathbf{b}))^\top \left(\frac{\partial}{\partial W} (W\mathbf{x}+\mathbf{b}) \right) =$$

$$= (\bar{\delta}_y - \mathrm{softmax}(W\mathbf{x}+\mathbf{b}))^\top \frac{\partial \mathbf{u}}{\partial W} =$$

$$= (\bar{\delta}_y - \mathrm{softmax}(W\mathbf{x}+\mathbf{b})) \otimes \mathbf{x}$$

Защото, ако $\mathbf{v} = \bar{\delta}_y - \mathrm{softmax}(W\mathbf{x}+\mathbf{b})$, то:

$$\left(\mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial W} \right)_{p,q} = \mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial W_{p,q}} = \sum_{k=1}^{K} \mathbf{v}_k \left(\frac{\partial \mathbf{u}}{\partial W_{p,q}} \right)_k = \sum_{k=1}^{K} \mathbf{v}_k \delta_{p=k} \mathbf{x}_q = \mathbf{v}_p \mathbf{x}_q$$

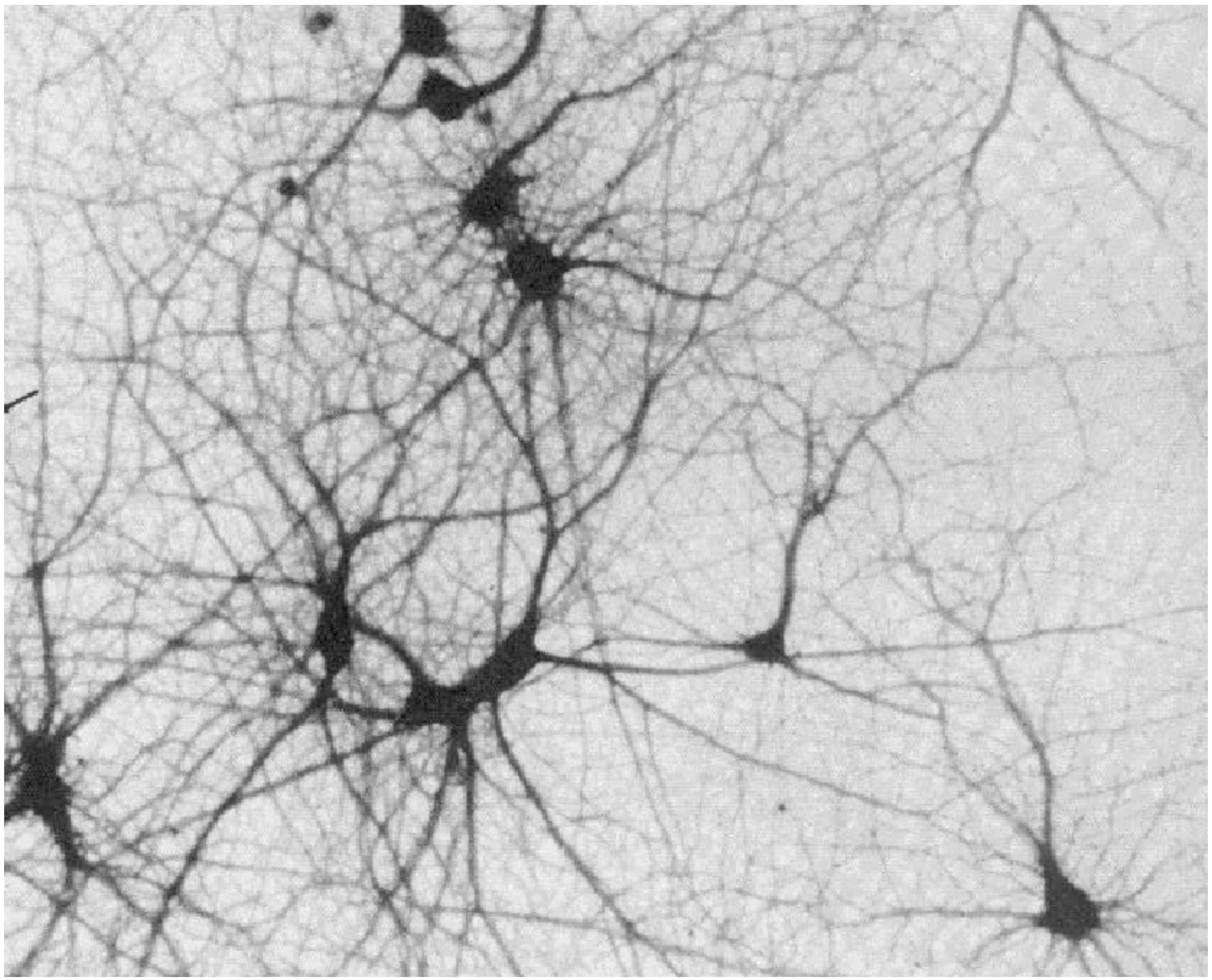
Градиент на логистична регресия при много класове

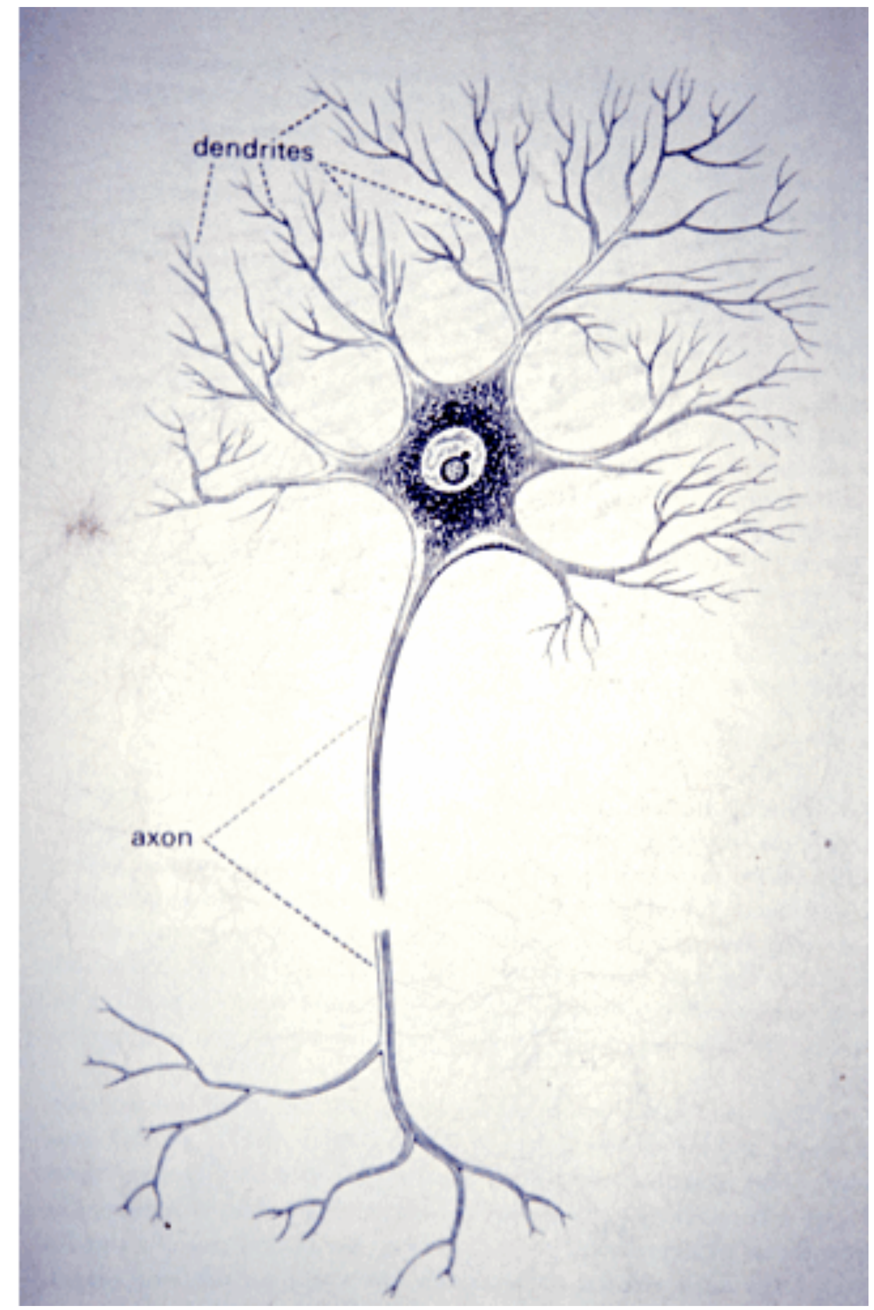
- $$\frac{\partial}{\partial \mathbf{b}} H_X(\Pr \| \Pr_{W,\mathbf{b}}) = -\frac{1}{m} \sum_{i=1}^{m} (\bar{\delta}_{y^{(i)}} - \mathrm{softmax}(W\mathbf{x}^{(i)} + \mathbf{b}))$$
- $$\frac{\partial}{\partial W} H_X(\Pr \| \Pr_{W,\mathbf{b}}) = -\frac{1}{m} \sum_{i=1}^{m} (\bar{\delta}_{y^{(i)}} - \mathrm{softmax}(W\mathbf{x}^{(i)} + \mathbf{b})) \otimes \mathbf{x}^{(i)}$$

План на лекцията

1. Формалности за курса (5 мин)
2. Логистична регресия (15 мин)
3. Обучение чрез спускане по градиента (15 мин)
4. Логистична регресия при много класове (15 мин)
5. **Изкуствени невронни мрежи (10 мин)**
6. Представимост на функции с невронни мрежи (10 мин)
7. Многослойни перцептрони (15 мин)

Неврони





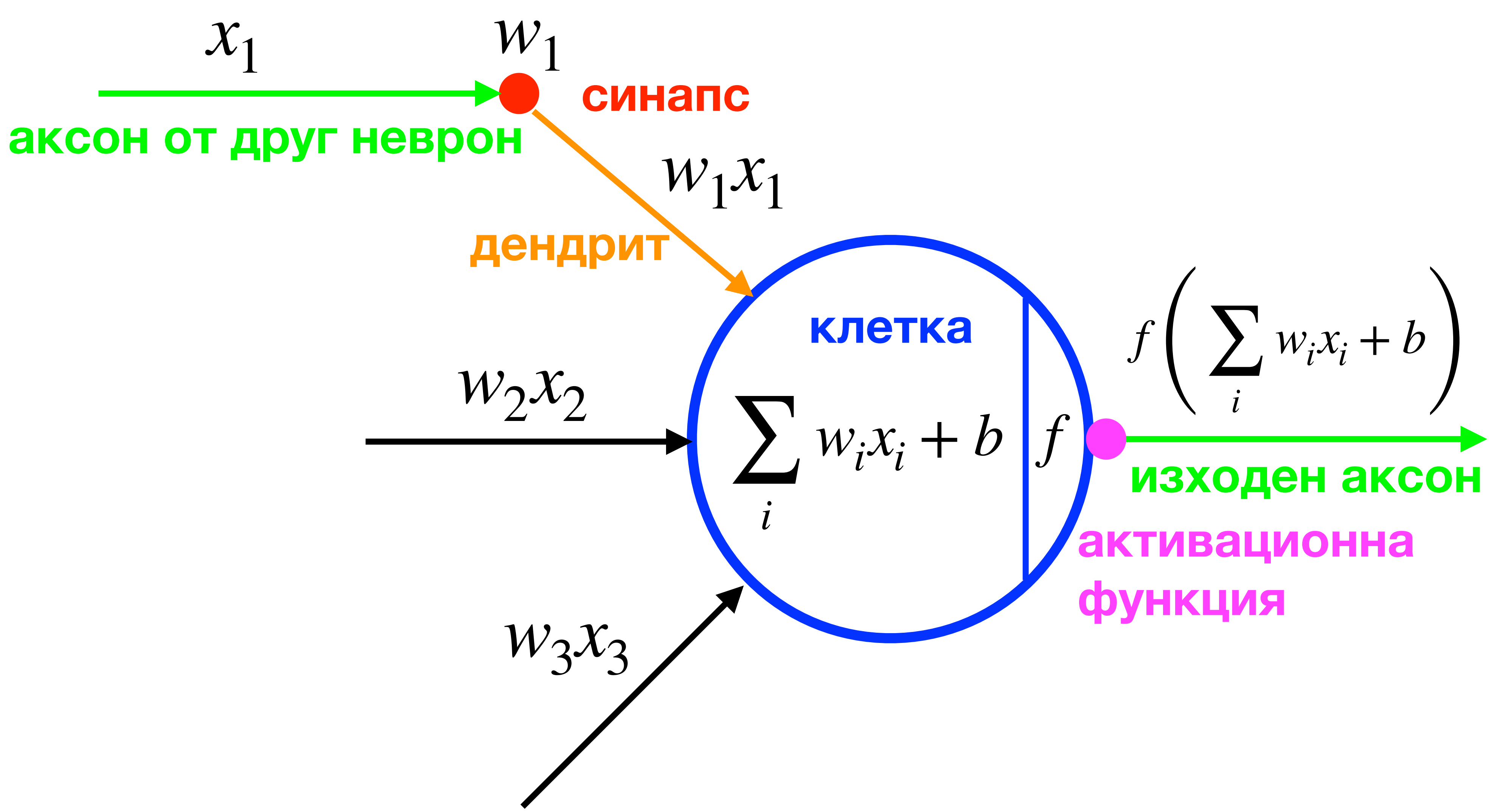
ANN или SNN

- Изкуствените невронни мрежи (ANN) са много опростен модел на биологичните невронни мрежи. Те са в основата на съвременните методи на изкуствения интелект, чрез които в последните години са постигнати забележителни успехи

- Импулсните невронни мрежи (SNN) обхващат модели, имитиращи невронната динамика на мозъка. В допълнение към невронното и синаптичното състояние, SNN включват концепцията за време в своя оперативен модел.

- В рамките на нашия курс ще разглеждаме само Изкуствените невронни мрежи (ANN).

Груба симулация на неврон — изкуствени неврони



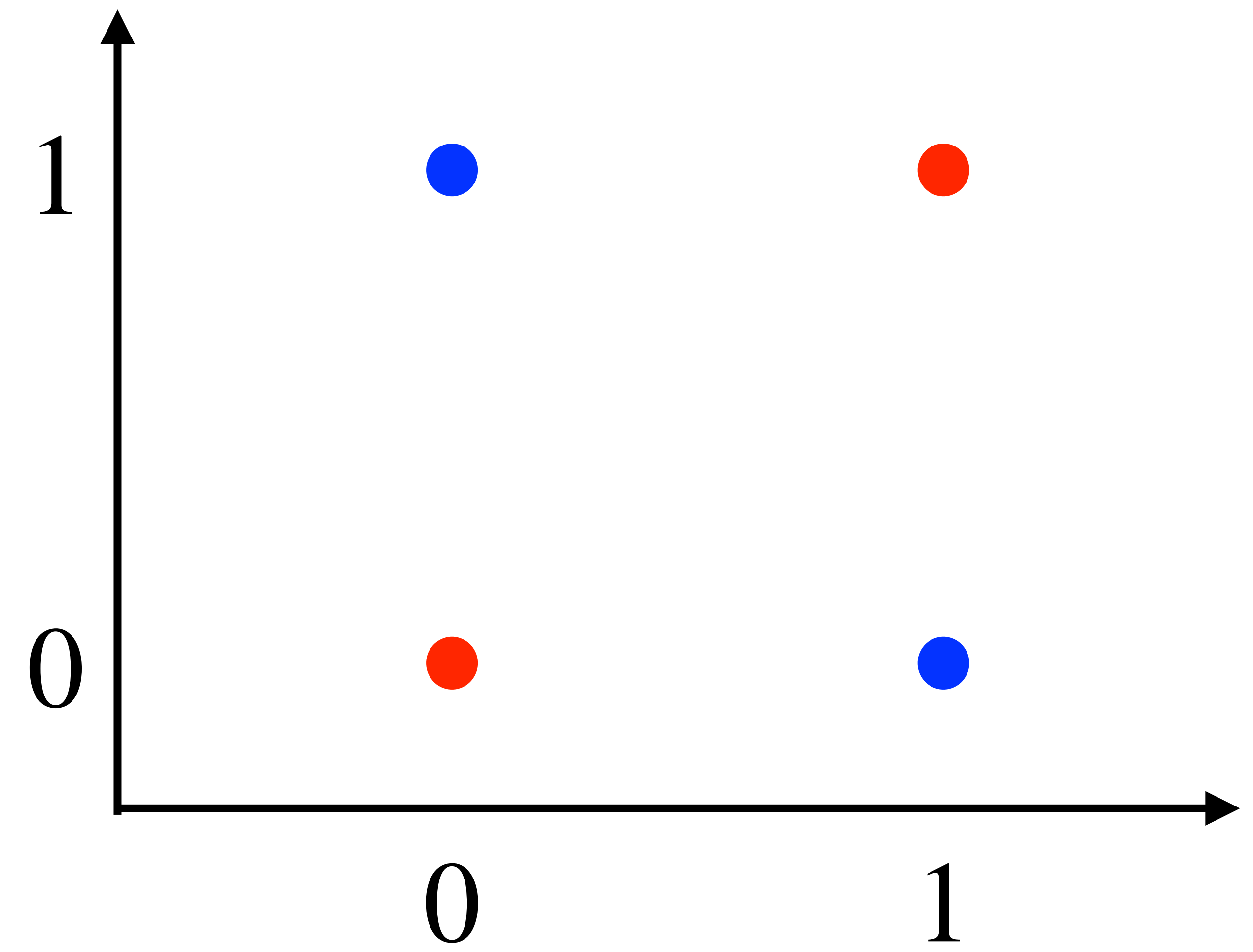
При $f = \sigma$ получаваме логистичната регресия

План на лекцията

1. Формалности за курса (5 мин)
2. Логистична регресия (15 мин)
3. Обучение чрез спускане по градиента (15 мин)
4. Логистична регресия при много класове (15 мин)
5. Изкуствени невронни мрежи (10 мин)
6. **Представимост на функции с невронни мрежи (10 мин)**
7. Многослойни перцептрони (15 мин)

Ограничения на линейните модели — проблемът XOR

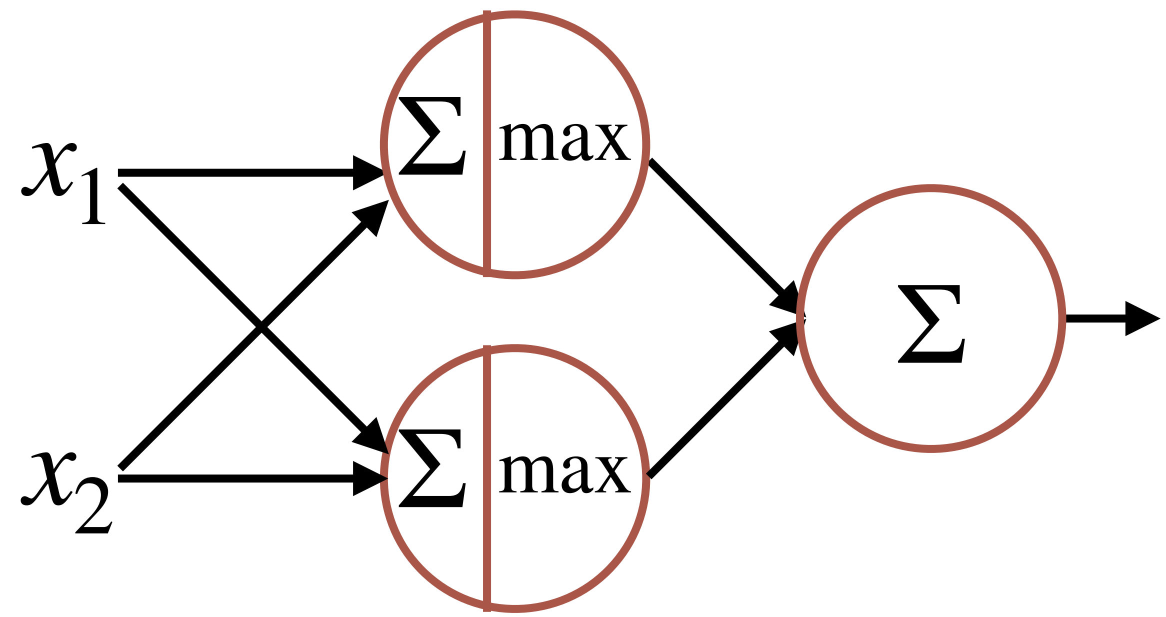
$$0w_1 + 0w_2 + b < 0$$
$$0w_1 + 1w_2 + b \geq 0$$
$$1w_1 + 0w_2 + b \geq 0$$
$$1w_1 + 1w_2 + b < 0$$



Не съществува права, която да раздели тези наблюдения

Двуслойна невронна мрежа

- $W' = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, b' = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$
- $\mathbf{w} = \begin{bmatrix} 3 \\ -1 \end{bmatrix}, b = \frac{1}{2}$
- $\mathbf{z} = \max(W'\mathbf{x} + \mathbf{b}', 0)$
  $y = \mathbf{w}^\top \mathbf{z} + b$
- Така дефинираната функция разделя наблюденията.



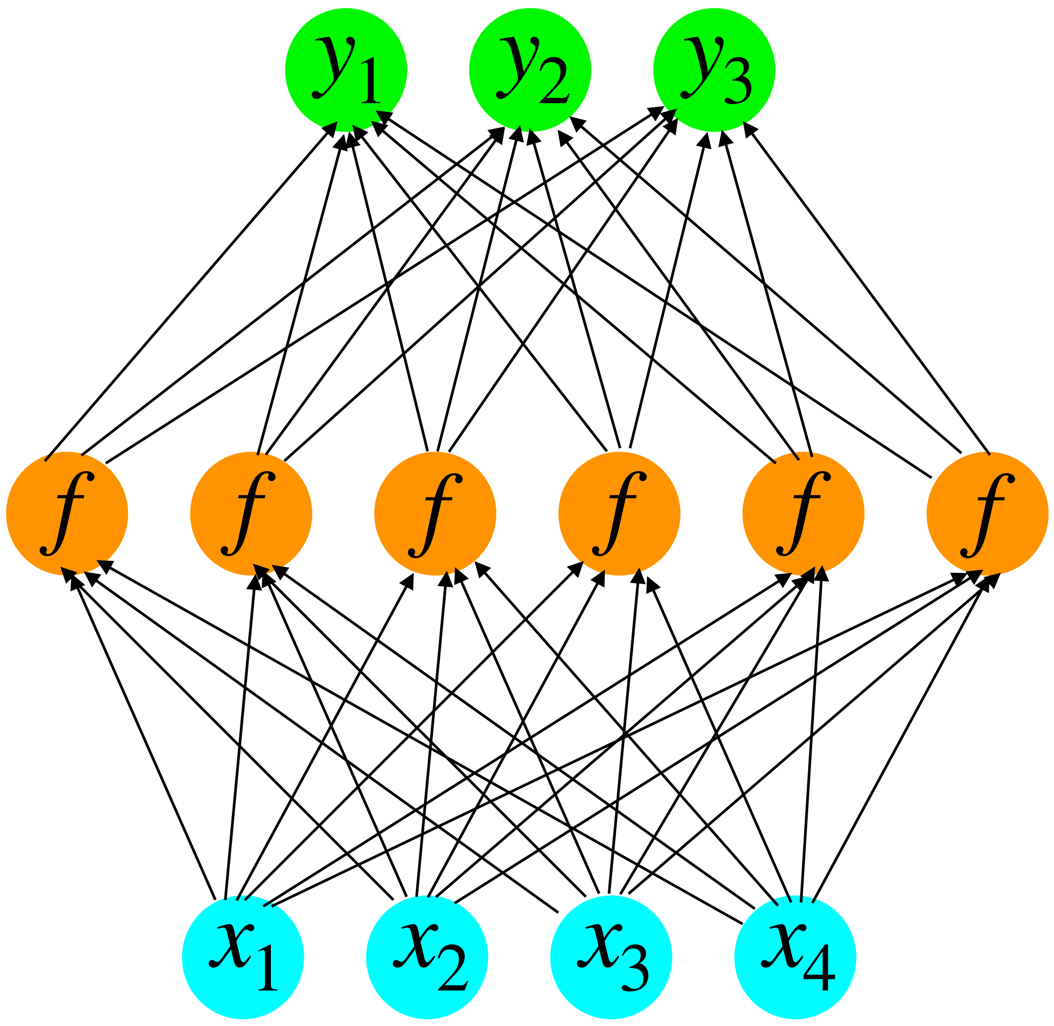
План на лекцията

1. Формалности за курса (5 мин)
2. Логистична регресия (15 мин)
3. Обучение чрез спускане по градиента (15 мин)
4. Логистична регресия при много класове (15 мин)
5. Изкуствени невронни мрежи (10 мин)
6. Представимост на функции с невронни мрежи (10 мин)
7. **Многослойни перцептрони (15 мин)**

Перцептрони

- Прост перцептрон — MLP0:
  $\mathbf{x} \in \mathbb{R}^{d_{in}}, W \in \mathbb{R}^{d_{out} \times d_{in}}, \mathbf{b} \in \mathbb{R}^{d_{out}}$
  $\mathrm{NN}_{\mathrm{MLP0}}(\mathbf{x}) = W\mathbf{x} + \mathbf{b}$

- Еднослоен перцептрон (един скрит слой) — MLP1:
  $\mathbf{x} \in \mathbb{R}^{d_{in}}, W^{(1)} \in \mathbb{R}^{d_1 \times d_{in}}, \mathbf{b}^{(1)} \in \mathbb{R}^{d_1}$
  $W^{(2)} \in \mathbb{R}^{d_{out} \times d_1}, \mathbf{b}^{(2)} \in \mathbb{R}^{d_{out}}, f : \mathbb{R} \to \mathbb{R}$
  $\mathrm{NN}_{\mathrm{MLP1}}(\mathbf{x}) = W^{(2)} f(W^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) + \mathbf{b}^{(2)}$



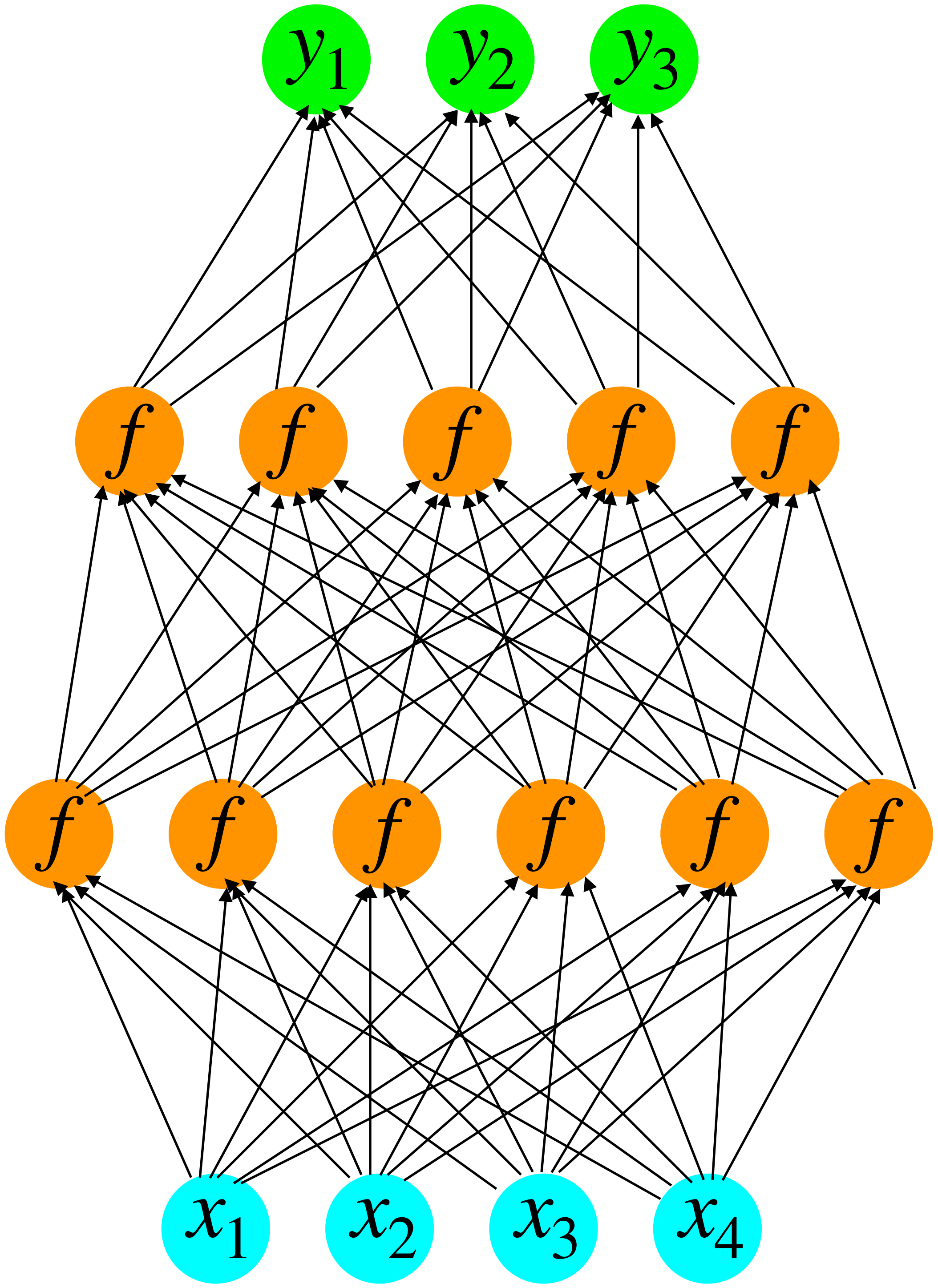
Многослоен перцептрон

Multi Layer Perceptron — MLP

Двуслоен перцептрон (два скрити слоя) — MLP2:

$\mathbf{x} \in \mathbb{R}^{d_{in}}, W^{(1)} \in \mathbb{R}^{d_1 \times d_{in}}, \mathbf{b}^{(1)} \in \mathbb{R}^{d_1}$
$W^{(2)} \in \mathbb{R}^{d_2 \times d_1}, \mathbf{b}^{(2)} \in \mathbb{R}^{d_2}, f^{(1)} : \mathbb{R} \to \mathbb{R}$
$W^{(3)} \in \mathbb{R}^{d_{out} \times d_2}, \mathbf{b}^{(3)} \in \mathbb{R}^{d_{out}}, f^{(2)} : \mathbb{R} \to \mathbb{R}$

$\mathrm{NN}_{\mathrm{MLP2}}(\mathbf{x}) =$
$= W^{(3)} f^{(2)}(W^{(2)} f^{(1)}(W^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) + \mathbf{b}^{(2)}) + \mathbf{b}^{(3)}$



Теорема за представимост на Борелово измеримите функции

- Всяка Борелово измерима функция $f : \mathbb{R}^n \to \mathbb{R}^m$ дефинирана върху компактно множество може да се приближи произволно точно с многослоен перцептрон с поне един скрит слой (MLP1).

- Съществува многослойна невронна мрежа с даден размер, която не може да се приближи с невронна мрежа с по-малък брой скрити слоеве, освен ако броят на невроните в междинните слоеве не е експоненциално по-голям от броя им в първоначалната мрежа.

- Доказателство — в курса “Теория на машинното обучение и някои нейни приложения в невронните мрежи“

Ограничения

- Теоремите за представимост са резултати за съществуване.
  - Те не разглеждат въпроса как да се намери съответно представяне или как да се извърши обучение.
  - Те не дават насоки за необходимия брой параметрите на модела или каква архитектура е необходима.

Заключение

- Изкуствените невронни мрежи са сравнително универсален модел за апроксимация на сложни функции.
- Чрез дълбоки (многослойни) архитектури значително се разширява изразителната им способност.
- Обучението на дълбоките невронни мрежи се извършва предимно с градиентни методи.
- Следващия път ще разгледаме ефективен и автоматичен метод за намиране на градиентите на сложни функции известен като Backpropagation
- Този метод се ползва на практика във всички софтуерни системи за дълбоко обучение — pytorch, tensorflow, CNTK, …