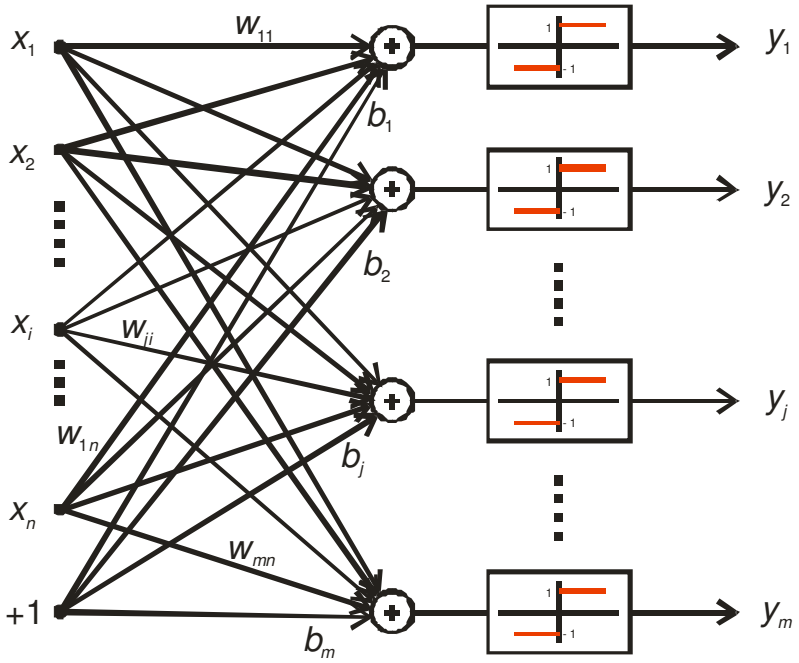
3. Vícevrstvé dopředné sítě

- Jsou tvořeny jednou nebo více vrstvami neuronů (perceptronů). Výstup jedné vrstvy je přitom připojen na vstup následující vrstvy a signál se v pracovní fázi sítě šíří pouze ze vstupu sítě na její výstup (tzn. vícevrstvé dopředné sítě nemají v pracovní fázi zpětnou vazbu).
- V literatuře se vyskytují pod různými názvy:
  - sítě s dopředným šířením
  - feedforward networks
  - multilayer perceptron (MLP)
  - MLP sítě
  - vícevrstvý perceptron
  - perceptronové sítě
  - …

3.1. Jednovrstvá síť s binární bipolární aktivační funkcí



$\boldsymbol{x} = [x_1, x_2, ..., x_i, ..., x_n]^T$ ... vstupní vektor

$$\boldsymbol{W} = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \dots & \dots & \dots & \dots \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{bmatrix}$$ ... váhová matice

$\boldsymbol{b} = [b_1, b_2, ..., b_j, ..., b_m]^T$ ... prahový vektor

$\boldsymbol{y} = [y_1, y_2, ..., y_j, ..., y_m]^T$ ... výstupní vektor

Pro výstup j-tého neuronu platí: $y_j = \mathbf{sgn}\left(\sum_{i=1}^{n} w_{ji} x_i + b_j \right)$

Trénování sítě pomocí učení s učitelem

- předpokládá se, že máme k dispozici trénovací množinu, tj. množinu P dvojic [vstup x_p, požadovaný výstup u_p]
- chceme nastavit váhy a prahy sítě tak, aby výstup sítě byl pro všechny vstupní vektory z trénovací množiny správný, tzn. aby celková chyba definovaná vztahem

$$E = \sum_{p=1}^{P} \varepsilon_p = \frac{1}{2} \sum_{p=1}^{P} \|\boldsymbol{u}_p - \boldsymbol{y}_p\|^2$$

byla nulová (minimální). Přitom

$\boldsymbol{y}_p = \mathrm{sgn}(\boldsymbol{W} \cdot \boldsymbol{x}_p + \boldsymbol{b})$ je skutečný výstup sítě pro vstup $\boldsymbol{x}_p$,

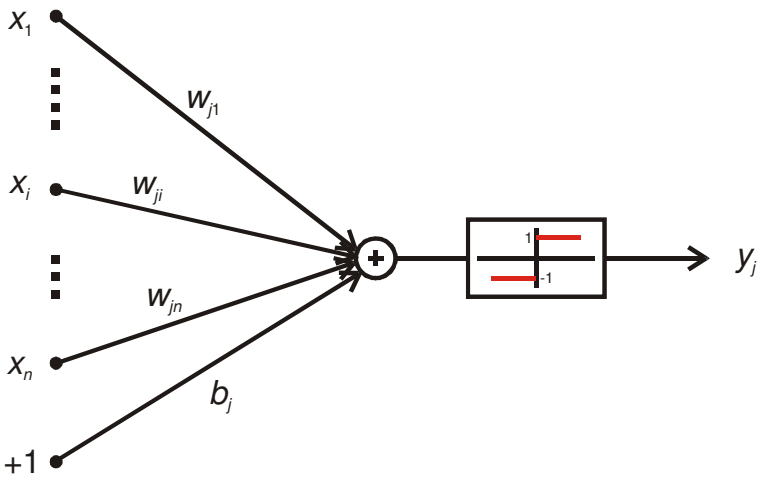
$\boldsymbol{u}_p$ je požadovaný výstup sítě pro vstup $\boldsymbol{x}_p$,

P je počet trénovacích dvojic,

ε_p je chyba od p-té trénovací dvojice.

Co vlastně po síti při trénování chceme?

Ilustrační představa pro 1 perceptron:



$$y_j = \mathrm{sgn}\left(\sum_{i=1}^{n} w_{ji} x_i + b_j \right)$$

- Předpokládejme, že chceme změnit jedinou váhu, např. w_{ji}, a využijeme k tomu konkrétní trénovací dvojici, např. [$\boldsymbol{x}$,u]; skutečný výstup pro vstup $\boldsymbol{x}$ označíme y_j.
- Jak w_{ji} najít?

  1. Pokud je výstup sítě správný, tj. $y_j = u$, není třeba w_{ji} měnit.

  2. Pokud je výstup sítě chybný, tj. $y_j \neq u$, je třeba w_{ji} upravit. Mohou nastat pouze 2 chybné případy:

     a) Chceme $u = +1$, a vyšlo $y_j = -1$.

     b) Chceme $u = -1$, a vyšlo $y_j = +1$.

Ad a)

Platí

$$y_j = \operatorname{sgn}\left(\sum_{i=1}^{n} w_{ji} x_i + b_j \right) =$$

$$= \operatorname{sgn}(w_{j1} x_1 + w_{j2} x_2 + \ldots + w_{ji} x_i + \ldots + w_{jn} x_n + b_j)$$

Protože vyšlo $y_j = -1$, muselo platit

$$w_{j1} x_1 + w_{j2} x_2 + \ldots + w_{ji} x_i + \ldots + w_{jn} x_n + b_j < 0.$$

Chceme ale, aby platilo $y_j = u = 1$, tzn. chceme, aby platilo

$$w_{j1} x_1 + w_{j2} x_2 + \ldots + w_{ji} x_i + \ldots + w_{jn} x_n + b_j \geq 0. \qquad (*)$$

Aby k tomu došlo, musí se některý ze sčítanců na levé straně nerovnosti (*) zvětšit. Předpokládáme, že chceme změnit pouze w_{ji}, tzn. všechno kromě w_{ji} považujeme konstanty. Upravíme tedy w_{ji} na w_{ji}^* tak, aby platilo $w_{ji}^* \cdot x_i > w_{ji} \cdot x_i$. To znamená, že pro $x_i > 0$ je třeba w_{ji} zvětšit a pro $x_i < 0$ je třeba w_{ji} zmenšit. To zaručuje vztah

$$w_{ji}^* = w_{ji} + c(u - y_j) \cdot x_i, \qquad (**)$$

kde $c > 0$ je tzv. konstanta učení.

Ad b) analogicky, vyjde též vztah (**).

Vztah (**) lze zobecnit na všechny váhy a prahy sítě, a pro změnu vah a prahů pak platí vztahy

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + c\,(\boldsymbol{u}_p - \boldsymbol{y}_p)\cdot \boldsymbol{x}_p^{\mathrm{T}}$$

$$\boldsymbol{b}(k+1) = \boldsymbol{b}(k) + c\,(\boldsymbol{u}_p - \boldsymbol{y}_p)$$

... pravidlo učení perceptronu (k je čas)

Princip trénovacího algoritmu

1. Inicializace vah a prahů sítě
2. Dokud není splněna zastavovací podmínka
   a. Výpočet skutečného výstupu pro zadaný vstup
   b. Modifikace vah a prahů

Algoritmus trénování jednovrstvé sítě s binární bipolární aktivační funkcí

Vstupní podmínky:

Je dána trénovací množina, tj. P dvojic $[\boldsymbol{x}_p, \boldsymbol{u}_p]$, $p=1,2,...,P$.
$\boldsymbol{x}_p$ je vstupní vektor dimenze n
$\boldsymbol{u}_p$ je požadovaný výstupní vektor dimenze m pro vstupní vektor $\boldsymbol{x}_p$

1. krok - Inicializace

- Váhovou matici $\boldsymbol{W}(1)$ typu $m \times n$ a prahový vektor $\boldsymbol{b}(1)$ dimenze m inicializujeme malými náhodnými čísly.
- Zvolíme $c > 0$, $E_{max} \geq 0$ a $TC_{max} > 0$.
- Stanovíme $k = 1$, $p = 1$, $q = 0$ a $E(0) = 0$.

E_{max} je maximální povolená chyba, na kterou nebo pod kterou se chceme dostat
TC_{max} je maximální počet trénovacích cyklů, ve kterých chceme síť natrénovat
q je počet trénovacích cyklů
E je chyba způsobená nesprávnou činností sítě během jednoho trénovacího cyklu
c je konstanta učení

2. krok - Výpočet výstupu

$\boldsymbol{x}(k) = \boldsymbol{x}_p$

$\boldsymbol{y}(k)$

$= \quad [\mathrm{sgn}(\boldsymbol{w}_1(k)\cdot\boldsymbol{x}(k)+b_1(k)), \quad \mathrm{sgn}(\boldsymbol{w}_2(k)\cdot\boldsymbol{x}(k)+b_2(k)), \quad \ldots \quad ,$
$\mathrm{sgn}(\boldsymbol{w}_m(k)\cdot\boldsymbol{x}(k)+b_m(k))]^{\mathrm{T}}$

$\boldsymbol{u}(k) = \boldsymbol{u}_p$

$\boldsymbol{w}_i$ je *i*-tá řádka matice $\boldsymbol{W}$

b_i je *i*-tý prvek vektoru $\boldsymbol{b}$

3. krok - Výpočet chyby

$E(k)=E(k\text{-}1)+½(\boldsymbol{u}(k)\text{-}\boldsymbol{y}(k))^{\mathrm{T}}\ (\boldsymbol{u}(k)\text{-}\boldsymbol{y}(k))$

4. krok - Modifikace vah a prahů

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + c\,(\boldsymbol{u}(k) - \boldsymbol{y}(k))\cdot\boldsymbol{x}^{\mathrm{T}}(k)$$

$$\boldsymbol{b}(k+1) = \boldsymbol{b}(k) + c\,(\boldsymbol{u}(k) - \boldsymbol{y}(k))$$

5. krok - Konec trénovacího cyklu

Je-li $p < P$ potom $p = p + 1$
$k = k + 1$
jdi na krok 2)
jinak $q = q + 1$
jdi na krok 6)

6. krok - Konec trénování

$E_c(q) = E(k)$

Je-li $E_c(q) \leq E_{max}$

potom KONEC (síť se podařilo natrénovat)

jinak je-li $q > TC_{max}$ potom KONEC (síť se nepodařilo natrénovat)

jinak $E(0) = 0$

$k = k + 1$

$p = 1$

jdi na krok 2)

Poznámky k trénování sítě:

- Jednovrstvou síť lze natrénovat v případě, že data z trénovací množiny jsou uspořádána do shluků tak, že shluky jsou lineárně separabilní, tj. oddělitelné přímkou. (Pozor na požadovaný výstup !!!). Pokud jsou data uspořádána jinak, síť natrénovat nelze.
- Pro výpočet celkové chyby E_c lze využít též vztah

$$E = \frac{1}{2PM} \sum_{p=1}^{P} \|\boldsymbol{u}_p - \boldsymbol{y}_p\|^2 = \frac{1}{2PM} \sum_{p=1}^{P} \sum_{m=1}^{M} (u_{pm} - y_{pm})^2 ,$$

  kde M je počet výstupů sítě, P je počet trénovacích dvojic v trénovací množině a u_{pm} resp. y_{pm} je m-tá složka vektoru $\boldsymbol{u}_p$, resp. $\boldsymbol{y}_p$. Takto definovaná chyba umožňuje lépe posoudit činnost sítí s různým počtem výstupů a různě velkou trénovací množinou.
- Jednovrstvou síť s binární unipolární aktivační funkcí lze trénovat stejným algoritmem.