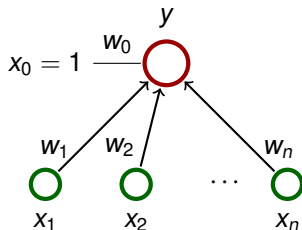


Perceptron a ADALINE

- ▶ Perceptron
- ▶ Perceptronové učící pravidlo
- ▶ Konvergence učení perceptronu
- ▶ ADALINE
- ▶ Učení ADALINE

Organizační dynamika:



$\vec{w} = (w_0, w_1, \dots, w_n)$ a $\vec{X} = (x_0, x_1, \dots, x_n)$ kde $x_0 = 1$.

Aktivní dynamika:

- ▶ vnitřní potenciál: $\xi = w_0 + \sum_{i=1}^n w_i x_i = \sum_{i=0}^n w_i x_i = \vec{w} \cdot \vec{X}$
- ▶ aktivační funkce: $\sigma(\xi) = \begin{cases} 1 & \xi \geq 0; \\ 0 & \xi < 0. \end{cases}$
- ▶ funkce sítě: $y[\vec{w}](x_1, \dots, x_n) = \sigma(\xi) = \sigma(\vec{w} \cdot \vec{X})$

Adaptivní dynamika:

- ▶ Dána množina **tréninkových vzorů**

$$\mathcal{T} = \{(\vec{x}_1, d_1), (\vec{x}_2, d_2), \dots, (\vec{x}_p, d_p)\}$$

Zde $\vec{x}_k = (x_{k1} \dots, x_{kn}) \in \mathbb{R}^n$ je vstup k -tého vzoru a $d_k \in \{0, 1\}$ je očekávaný výstup.

(d_k určuje, do které ze dvou kategorií dané \vec{x}_k patří).

- ▶ Označme: $\vec{X}_k = (x_{k0}, x_{k1} \dots, x_{kn}) \in \mathbb{R}^{n+1}$ kde $x_{k0} = 1$.
- ▶ Vektor vah \vec{w} je **konzistentní s \mathcal{T}** pokud $y[\vec{w}](x_{k1} \dots, x_{kn}) = \sigma(\vec{w} \cdot \vec{X}_k) = d_k$ pro každé $k = 1, \dots, p$. Množina \mathcal{T} je **vnitřně konzistentní** pokud existuje vektor \vec{w} , který je s ní konzistentní.
- ▶ Cílem je nalézt vektor \vec{w} , který je konzistentní s \mathcal{T} za předpokladu, že \mathcal{T} je vnitřně konzistentní.

Perceptron - adaptivní dynamika

Online učící algoritmus:

Idea: Cyklicky prochází vzory a adaptuje podle nich váhy, tj. otáčí dělící nadrovinu tak, aby se zmenšila vzdálenost špatně klasifikovaného vzoru od jeho příslušného poloprostoru.

Prakticky počítá posloupnost vektorů vah $\vec{w}^{(0)}, \vec{w}^{(1)}, \vec{w}^{(2)}, \dots$

- ▶ váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- ▶ v m -tém kroku je $\vec{w}^{(m)}$ vypočteno takto:

$$\vec{w}^{(m)} = \vec{w}^{(m-1)} - \varepsilon \cdot (\sigma(\vec{w}^{(m-1)} \cdot \vec{X}_k) - d_k) \cdot \vec{X}_k$$

Zde $k = ((m - 1) \bmod p) + 1$ (tj. cyklické procházení vzorů) a $0 < \varepsilon \leq 1$ je **rychlost učení**.

Věta (Rosenblatt)

Jestliže je \mathcal{T} vnitřně konzistentní, pak existuje m^ takové, že $\vec{w}^{(m^*)}$ je konzistentní s \mathcal{T} .*

Důkaz Rosenblattovy věty

Pro zjednodušení budeme dále předpokládat, že $\varepsilon = 1$.

Nejprve si algoritmus přepíšeme do méně kompaktní formy.

Označme

$$\vec{z}_k = \begin{cases} \vec{X}_k & d_k = 1 \\ -\vec{X}_k & d_k = 0 \end{cases}$$

Pak lze online učicí algoritmus přepsat takto:

- ▶ váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- ▶ v m -tém kroku je $\vec{w}^{(m)}$ vypočteno takto:
 - ▶ **Jestliže** $\sigma(\vec{w}^{(m-1)} \cdot \vec{X}_k) = d_k$, **pak** $\vec{w}^{(m)} = \vec{w}^{(m-1)}$
 - ▶ **Jestliže** $\sigma(\vec{w}^{(m-1)} \cdot \vec{X}_k) \neq d_k$, **pak** $\vec{w}^{(m)} = \vec{w}^{(m-1)} + \vec{z}_k$
(Řekneme, že nastala korekce.)

kde $k = ((m - 1) \bmod p) + 1$.

Důkaz Rosenblattovy věty

Idea:

- ▶ Pro daný vektor $\vec{w} = (w_0, \dots, w_n)$ označme $\|\vec{w}\|$ jeho Euklidovskou délku $\sqrt{\vec{w} \cdot \vec{w}} = \sqrt{\sum_{i=0}^n w_i^2}$
- ▶ Uvážíme *hodně dlouhý vektor* (spočítáme jak dlouhý) \vec{w}^* , který je konzistentní s \mathcal{T} .
- ▶ Ukážeme, že pokud došlo v m -tém kroku ke korekci vah (tedy $\vec{w}^{(m)} = \vec{w}^{(m-1)} + \vec{z}_k$), pak

$$\|\vec{w}^{(m)} - \vec{w}^*\|^2 \leq \|\vec{w}^{(m-1)} - \vec{w}^*\|^2 - \delta$$

kde $\delta > 0$ je fixní hodnota (kterou také spočítáme).

- ▶ Z toho plyne, že algoritmus nemůže udělat nekonečně mnoho korekcí.

Dávkový učící algoritmus:

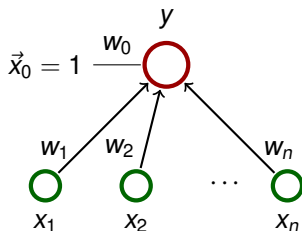
Vypočte posloupnost $\vec{w}^{(0)}, \vec{w}^{(1)}, \vec{w}^{(2)}, \dots$ váhových vektorů.

- ▶ váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- ▶ v m -tém kroku je $\vec{w}^{(m)}$ vypočteno takto:

$$\vec{w}^{(m)} = \vec{w}^{(m-1)} - \varepsilon \cdot \sum_{k=1}^p (\sigma(\vec{w}^{(m-1)} \cdot \vec{X}_k) - d_k) \cdot \vec{X}_k$$

Zde $0 < \varepsilon \leq 1$ je rychlost učení.

Organizační dynamika:



$\vec{w} = (w_0, w_1, \dots, w_n)$ a $\vec{X} = (x_0, x_1, \dots, x_n)$ kde $x_0 = 1$.

Aktivní dynamika:

- ▶ vnitřní potenciál: $\xi = w_0 + \sum_{i=1}^n w_i x_i = \sum_{i=0}^n w_i x_i = \vec{w} \cdot \vec{X}$
- ▶ aktivační funkce: $\sigma(\xi) = \xi$
- ▶ funkce sítě: $y[\vec{w}](x_1, \dots, x_n) = \sigma(\xi) = \vec{w} \cdot \vec{X}$

Adaptivní dynamika:

- ▶ Dána množina **tréninkových vzorů**

$$\mathcal{T} = \{(\vec{x}_1, d_1), (\vec{x}_2, d_2), \dots, (\vec{x}_p, d_p)\}$$

Zde $\vec{x}_k = (x_{k0}, x_{k1}, \dots, x_{kn}) \in \mathbb{R}^{n+1}$, $x_{k0} = 1$, je vstup k -tého vzoru a $d_k \in \mathbb{R}$ je očekávaný výstup.

Intuice: chceme, aby síť počítala afinní aproximaci funkce, jejíž (některé) hodnoty nám předepisuje tréninková množina.

- ▶ Označme: $\vec{X}_k = (x_{k0}, x_{k1}, \dots, x_{kn}) \in \mathbb{R}^{n+1}$ kde $x_{k0} = 1$.
- ▶ **Chybová funkce:**

$$E(\vec{w}) = \frac{1}{2} \sum_{k=1}^p (\vec{w} \cdot \vec{X}_k - d_k)^2 = \frac{1}{2} \sum_{k=1}^p \left(\sum_{i=0}^n w_i x_{ki} - d_k \right)^2$$

- ▶ Cílem je nalézt \vec{w} , které minimalizuje $E(\vec{w})$.

Gradient chybové funkce

Uvažme **gradient** chybové funkce:

$$\nabla E(\vec{w}) = \left(\frac{\partial E}{\partial w_0}(\vec{w}), \dots, \frac{\partial E}{\partial w_n}(\vec{w}) \right) = \sum_{k=1}^p (\vec{w} \cdot \vec{X}_k - d_k) \cdot \vec{X}_k$$

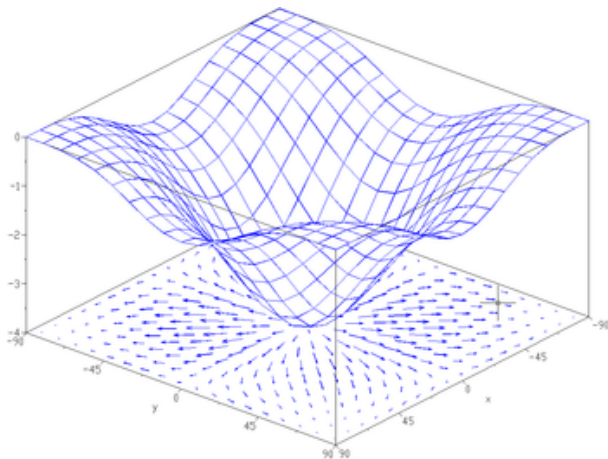
Intuice: $\nabla E(\vec{w})$ je vektor ve **váhovém prostoru**, který ukazuje směrem nejstrmějšího „růstu“ funkce $E(\vec{w})$. Uvědomte si, že vektory \vec{X}_k zde slouží pouze jako parametry funkce $E(\vec{w})$ a jsou tedy fixní.

Fakt

Pokud $\nabla E(\vec{w}) = \vec{0} = (0, \dots, 0)$, pak \vec{w} je globální minimum funkce E .

Námi uvažovaná chybová funkce $E(\vec{w})$ má globální minimum, protože je kvadratickou funkcí (konvexní paraboloid).

Gradient - ilustrace



Pozor! Tento obrázek pouze ilustruje pojem gradientu, nezobrazuje chybovou funkci $E(\vec{w})$

Dávkový algoritmus (gradientní sestup):

- ▶ váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- ▶ v m -tém kroku je $\vec{w}^{(m)}$ vypočteno takto:

$$\begin{aligned}\vec{w}^{(m)} &= \vec{w}^{(m-1)} - \varepsilon \cdot \nabla E(\vec{w}^{(m-1)}) \\ &= \vec{w}^{(m-1)} - \varepsilon \cdot \sum_{k=1}^p (\vec{w}^{(m-1)} \cdot \vec{X}_k - d_k) \cdot \vec{X}_k\end{aligned}$$

Zde $0 < \varepsilon \leq 1$ je rychlost učení.

(Všimněte si, že tento algoritmus je téměř stejný jako pro perceptron, protože $\vec{w}^{(m-1)} \cdot \vec{X}_k$ je hodnota funkce sítě (tedy $\sigma(\vec{w}^{(m-1)} \cdot \vec{X}_k)$ kde $\sigma(\xi) = \xi$.)

Tvrzení

Pro dostatečně malé $\varepsilon > 0$ posloupnost $\vec{w}^{(0)}, \vec{w}^{(1)}, \vec{w}^{(2)}, \dots$ konverguje (po složkách) ke globálnímu minimu funkce E (tedy k vektoru \vec{w} , který splňuje $\nabla E(\vec{w}) = \vec{0}$).

Online algoritmus (Delta-rule, Widrow-Hoff rule):

- ▶ váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- ▶ v m -tém kroku je $\vec{w}^{(m)}$ vypočteno takto:

$$\vec{w}^{(m)} = \vec{w}^{(m-1)} - \varepsilon(m) \cdot (\vec{w}^{(m-1)} \cdot \vec{X}_k - d_k) \cdot \vec{X}_k$$

kde $k = ((m - 1) \bmod p) + 1$ a $0 < \varepsilon(m) \leq 1$ je rychlost učení v m -tém kroku.

Všimněte si, že tento algoritmus nepracuje s celým gradientem, ale jenom s jeho částí, která přísluší právě zpracovávanému vzoru!

Věta (Widrow & Hoff)

Pokud $\varepsilon(m) = \frac{1}{m}$ pak $\vec{w}^{(0)}, \vec{w}^{(1)}, \vec{w}^{(2)}, \dots$ konverguje ke globálnímu minimu chybové funkce E .

- ▶ Množina **tréninkových vzorů** je

$$\mathcal{T} = \left\{ (\vec{x}_1, d_1), (\vec{x}_2, d_2), \dots, (\vec{x}_p, d_p) \right\}$$

kde $\vec{x}_k = (x_{k1}, \dots, x_{kn}) \in \mathbb{R}^n$ a $d_k \in \{1, -1\}$.

- ▶ Označme $\vec{X}_k = (x_{k0}, x_{k1}, \dots, x_{kn}) \in \mathbb{R}^{n+1}$ kde $x_{k0} = 1$.
- ▶ Síť se natrénuje ADALINE algoritmem.
- ▶ Očekáváme, že bude platit následující:
 - ▶ jestliže $d_k = 1$, pak $\vec{w} \cdot \vec{X}_k \geq 0$
 - ▶ jestliže $d_k = -1$, pak $\vec{w} \cdot \vec{X}_k < 0$
- ▶ To nemusí vždy platit, ale často platí. Výhoda je, že se ADALINE algoritmus postupně stabilizuje i v neseparabilním případě (na rozdíl od perceptronového algoritmu).