zápis z přednášky

Lineární diskriminační funkce

a klasifikátory předpokládající normální rozdělení příznaků

Jan Vaňáček, Aleš Kulhánek

2.4.2002

Uvedené téma je zpracováno na základě:

- zápisu z přednášky r.2001 na podobné téma, kterou zpracovali studenti: Adéla Klimentová, Libor Beneš, Dušan Hlaváč
- přednášky RPZ - 2.4.2002
- přednášky RPZ - 21.5.2002

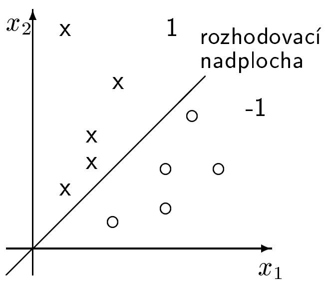
Obsah

1 Lineární diskriminační funkce

Máme-li množinu jevů $X = \{x_1, x_2, \ldots, x_n\}$ a množinu tříd $K = \{k_1, k_2, \ldots, k_m\}$, pak pro správnou klasifikaci v Bayesovské strategii je třeba nalézt sdruženou pravděpodobnost $P(X|k)$. To je však teoretická úloha. V praxi nejsme schopni tuto pravděpodobnost nalézt. Můžeme řešit pouze Bayesovskou úlohu - tedy pravděpodobnost $P(X|k)$ nahradit jejím odhadem $\hat{P} = (X|k)$ a snažit se minimalizovat střední ztrátu způsobenou nepřesností odhadu.

Odhad $\hat{P} = (X|k)$ hledáme z trénovací množiny $T = ((x_1, k_1), (x_2, k_2), \ldots, (x_n, k_m))$. Problémem trénovací množiny je omezená mohutnost, těžko odhadnutelná spolehlivost, častá přítomnost chybně klasifikovaných jevů, možnost existence stejných příznaků pro jevy různých tříd (např. výšku 160cm může mít muž i žena a přesto má tento prvek trénovací množiny význam pro správné rozhodování) atd.

Druhou možností je hledat přímo diskriminační funkci d(x) bez nutnosti nalezení odhadu pravděpodobnosti $\hat{P} = (X|k)$. Snažíme se nalézt princip rozhodování bez předchozího pochopení zkoumaných dat.



Obrázek 1: Lineární rozhodovací funkce

Nejjednodušší variantou $d(x)$ je lineární diskriminační funkce (LDF). Ta rozděluje množinu všech vstupních bodů na 2 množiny. Klasifikace do množin se provádí pomocí LDF a souřadnic daného prvku. Výsledkem rozhodování je např. ohodnocení na "1" pro prvky nad LDF a na "-1" pro prvky pod ní. LDF zasluhují bližší pozornost, neboť

- pro některé statistické modely se optimální strategie (bayesovská i nebayesovská) realizuje pomocí LDF ($P(x|w)$ s normálním rozdělením se stejnou kovarianční maticí nebo binární nezávislé příznaky)

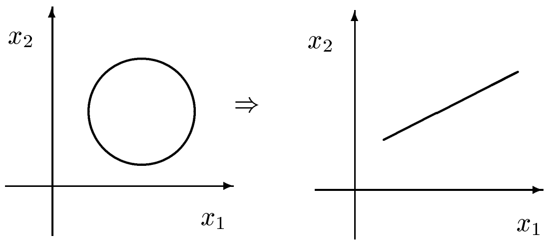
- některé nelineární diskriminační funkce lze vyjádřit jako lineární po vyrovnání příznakového prostoru[1]:

$$\Phi_i : \mathbb{R}^d \to \mathbb{R}, g_k(x) = \sum_i w_{ki} \Phi_i(x). \tag{1}$$

[1]Vyrovnávání příznakového prostoru se provádí vhodně zvolenou transformací souřadnic

- existuje univerzální způsob zobrazení příznakového prostoru do prostoru pravděpodobností, kde se řešení každé statistické úlohy realizuje lineární funkcí, důkaz existence LDF pro nezávislé binární příznaky (viz rovn. 5 str.7)
- kapacita (VC dimenze) lineárních strategií v n-rozměrném prostoru je $n+2$ a učení je korektní [2].
- existují postupy pro nalezení lineárních diskriminačních funkcí
  - Perceptron
  - Kozincův algoritmus
  - Support vector machine

Příklad transformace příznakového prostoru $x_1, x_2 \rightarrow y_1, y_2, y_3, y_4, y_5$:
$||\overline{x} - \overline{c}||^2 < r^2$
$(x_1 - c_1)^2 + (x_2 - c_2)^2 < r^2$
$x_1^2 - 2c_1^2 x_1 + c1^2 + x_2^2 - 2x_2c_2 + c_2^2 - r^2 < 0$
$y_1 + \omega_2 y_2 + y_4 + \omega_3 y_3 + y_5 < 0$



Obrázek 2: Transformace příznakového prostoru

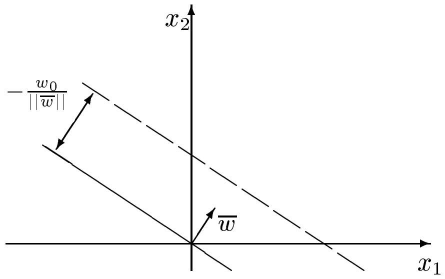
Při složitějších transformacích pak ale nastává problém s dimenzí prostoru příznaků (dimenze nového prostoru příznaků roste $\rightarrow \frac{(n+1)^2}{2}$, kde n je počet příznaků původního prostoru)

Rozhodovací funkci hledáme ve tvaru: $d(x) = \overline{w} \cdot \overline{x} + w_0$ kde w je vektor vah, x je vstupní vektor, w_0 konstanta posunutí.

Pro každou třídu z množiny $K = \{1, 2, 3, \ldots, M\}$ je diskriminační funkce dána vztahem: $g_k(\overline{x}) = \overline{w}_k \cdot \overline{x} + w_{k0}$.

kde $\overline{x}$ je vstupní vektor,
$\overline{w}$ je normálový vektor k rozhodovací nadrovině (obr. 3),
w_{k0} je posunutí rozhodovací nadroviny.

[2] Vlastnost korektní učení znamená, že na nekonečné trénovací množině konverguje chyba na trénovacích datech ke střední chybě.



Obrázek 3: Rozhodovací nadrovina ve 2D, normálový vektor

Hledaná funkce $d(x)$ potom bude:

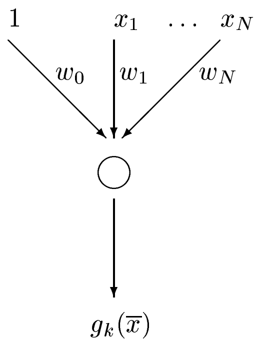
$$d(x) = \arg\max_{k \in K} g_k(\overline{x}), \tag{2}$$

což je stejné jako $d(x) = k^* \iff g_k^*(\overline{x}) \geq g_k(\overline{x}), k \in K$,
kde $g_k(\overline{x})$ je věrohodnost třídy k pro vektor $\overline{x}$.

Výsledkem tedy bude pro zadaný vstupní vektor třída, která má maximální věrohodnost.

1.1 Lineární klasifikátor

Lineární klasifikátor může být představován jako jednoduchý neuron s váženými vstupy a jedním výstupem. Konstantu posunutí ω_0 lze přičíst na začátek normálového vektoru rozhodovací nadroviny, pokud vstupní vektor rovněž na začátku rozšíříme o 1. [3] Tím získáme $N+1$ vstupů $\overline{x}$ a $N+1$ vah $w_0 \dots w_N$.



Obrázek 4: Jednoduchý neuron

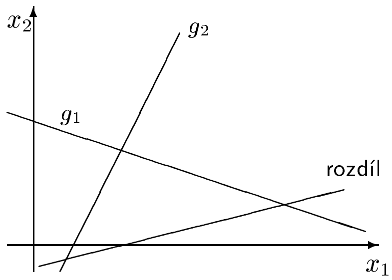
Pro dvě třídy jsou diskriminační funkce dány vztahem:

$$g_1(\overline{x}) = \overline{w}_1\overline{x} + w_{10}), \quad g_2(\overline{x}) = \overline{w}_2\overline{x} + w_{20}$$

[3] zavedeme homogenní souřadnice

Pro porovnání, která funkce je pro dané $\overline{x}$ větší, nám stačí najít vektor rozdílů:

$$g_1 - g_2 \rightarrow 0 \leq (\overline{w}_1 - \overline{w}_2)\overline{x} + w_{10} - w_{20} \tag{3}$$



Obrázek 5: Diskriminační funkce pro dvě třídy jako funkce rozdílu

Ke změně rozhodování potom dochazí, když funkce daná rozdílem obou diskriminačních funkcí prochazí nulou.

1.2 Příklad klasifikace

Příklad:

$$P(x_i|k) = P_{ki}^{x_i}(1 - P_{ki})^{1-x_i} = \begin{cases} P_k^{x_i} & \text{pro} x_i = 1 \\ 1 - P_k^{x_i} & \text{pro} x_i = 0 \end{cases}$$

kde x_i jsou binární na sobě nezávislé příznaky $(i = 1, 2, \ldots, N)$

	1	2		N - 1	N
x:			...		

$$\left.\begin{array}{l} P(x_i = 0|k) \\ P(x_i = 1|k) \end{array}\right\} \text{potřebujeme znát pro všechna } k$$

0	0.9	0.6	...	0.3
1	0.1	0.4	...	0.7

$p(x_i|k_i)$ - jednotlivé „příznaky“ s určitou pravděpodobností

$$p(\overline{x}|k) = \prod_{i=1}^{N} P_{ki}^{x_i}(1 - P_{ki})^{1-x_i}$$

např: $\overline{x} = (0, 0, \ldots, 1)$

$P((0, 0, \ldots, 1)|k_1) = \prod p(x_i|k) = 0.9 * 0.6 * \ldots * 0.7$

Součin z předchozího příkladu

$$p(\overline{x}|k) = \prod_{i=1}^{N} P_{ki}^{x_i} (1 - P_{ki})^{1-x_i} \tag{4}$$

převedeme pomocí přirozeného logaritmu na součet:

$$\begin{aligned}
&\sum_{i=1}^{N} \Big(\overline{x}_i \ln P_{ki} + (1 - \overline{x}_i) \ln(1 - P_{ki})\Big) \\
&\sum_{i=1}^{N} \Big(\overline{x}_i (\ln P_{ki} - \ln(1 - P_{ki})) + \ln(1 - P_{ki})\Big) \\
&\sum_{i=1}^{N} (\overline{x}_i \overline{w}_{ki} + w_{k0}) \ldots \text{rozepsaný skalární součin} \\
&\overline{x} \cdot \overline{w} + w_0 \ldots \text{výsledkem je lin. funkce proměnné } x.
\end{aligned} \tag{5}$$

2 Formulace úloh učení

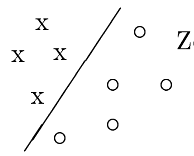
2.1 Metody učení

Dle různých metod učení volíme algoritmy, které mají za cíl určit parametry diskriminační funkce (w, b). Takových algoritmů existuje celá řada. Zde je jeden možný pohled na jejich rozdělení - dle metod učení.

1. algoritmy přizpůsobené podle typu trénovací množiny
   (a) konečná množina pozorování a jejich skrytých stavů $(x_1, k_1), (x_2, k_2)$
   (b) třídy jsou popsány Gaussovským rozdělením (nekonečné množiny)
2. algoritmy přizpůsobené pro danou formulaci úlohy, kterou řeší

2.2 Úloha minimalizace empirického rizika

$$R_{emp}(f) = \sum_{l} C(k_i, f_i) \quad (C = 0 \text{ pro } k_i = f(x_i);\ C = 1 \text{ pro } k_i \neq f(x_i)) \tag{6}$$



Zde, pro perceptronový algoritmus je $R_{emp} = 0$

2.3 Úloha minimalizace strukturálního rizika

$$R_{struct}(f) = \int_x \sum_k (C(k_i, f(x)) \cdot P(x, k) dx \tag{7}$$

Mez podle Vapnika:

$$R(f) \leq R_{emp}(f) + R_{struct}(h, \frac{1}{e}), \tag{8}$$

h je VC dimenze.

Pro lineární diskriminiační funkce

$$h \leq \frac{R^2}{\rho_0^2} + 1$$

2.4 Úloha nalezení optimální nadroviny

Zde již nestačí perceptronový algoritmus pro rozdělení bodů do dvou tříd. Pomocí SVM (Support Vector Machine) nebo jiných algoritmů hledáme rozdělovací funkci:

$$f(x) = (w^*, b^*) = \arg\max_{f(x)} \min_x \left[\frac{(w, x) + b}{|w|} \right]. \tag{9}$$

2.5 Zobecněná Andersonova úloha

Máme dvě třídy $k = \{1, 2\}$, $P(x|k, z)$ (z je nenáhodný zásah) má Gaussovské rozdělení.

$$P(x|k=1) = B(\mu^j, \sigma^j) \;\; j \in J_1$$
$$P(x|k=2) = B(\mu^j, \sigma^j) \;\; j \in J_2$$

Hledáme lineární strategii

$$f(x) = \langle w, x\rangle + b \;\; \overset{(1)}{\underset{(2)}{\gtrless}} \;\; 0, \tag{10}$$

která minimalizuje $max_{j \in J_1 \bigcup J_2} \varepsilon(f, \mu^j, \sigma^j)$,
kde $\varepsilon(f, \mu^j, \sigma^j)$ je pravděpodobnost, náhodný vektor x s Gaussovským rozdělením $N(\mu^j, \sigma^j)$ bude klasifikován do špatné třídy, tj.

$$f(x) < 0 \text{ pro } j \in J_1$$
$$f(x) > 0 \text{ pro } j \in J_2.$$

2.5.1 Algoritmus pro řešení zobecněné Andersonovy úlohy

1. Nejprve nalezneme vektor α, aby všechny skalární součiny $\langle \alpha, \mu^j\rangle$, $j \in J$ byly kladné.

2. Potom vypočítáme body dotyku ${x_0}^j$, $j \in J$, čísla r^{*j}, vybereme množinu J^0 těch j, pro která platí $r^{*j} = \min_{j \in J} r^{*j}$ a najít takový směr $\Delta\alpha$, který vyhovuje podmínce

$$\langle \Delta\alpha, {x_0}^j\rangle > 0, \;\; j \in J^0. \tag{11}$$

   Jestliže existuje vektor $\Delta\alpha$, který vyhovuje podmínce (11), potom není jediný ale takových vektorů je více. Vybíráme takový vektor, aby derivace funkce $-\varepsilon(\alpha, \mu^j, \sigma^j)$ ve směru $\Delta\alpha$ byla co největší pro všechna $j \in J^0, tj.$

$$\Delta\alpha = \arg\max_{\Delta\alpha} \min_{j} \left(- \frac{\partial \varepsilon\Big(\alpha + t \cdot \frac{\Delta\alpha}{|\Delta\alpha|}\Big)}{\partial t} \right). \tag{12}$$

3. Nyní hledáme takový parametr t, aby

$$t = \arg\max_{t} \min_{j \in J} \varepsilon(\alpha + t \cdot \Delta\alpha, \mu^j, \sigma^j) \tag{13}$$

   a vektor $\alpha = \alpha + t\Delta\alpha$. Takový vektor také musí vyhovovat podmínce (11).

4. Přejdeme znovu na bod (2) algoritmu.

Tento algoritmus má tu nevýhodu, že není zaručeno jeho ukončení v konečném čase. Výhodné ale je, že během iterací stále klesá $max_{j \in J} \varepsilon(\alpha, \mu^j, \sigma^j)$. Při použití algoritmu v programu potom stačí hlídat, jak se předchozí hodnota mění a rozhodovat, zda má cyklus ještě pokračovat.

2.5.2 ε - řešení zobecněné Andersonovy úlohy

Úlohu lze převést na problém rozdělení nekonečných množin:

$$(\overline{w}, \overline{x}) + b \geq 0 \tag{14}$$

$$\overline{x} \in X_1(v), \;\; \overline{x} \in X_2(v), \quad \text{kde } X_1(v) = \bigcup_{j \in J_1} x^j(v), \;\; X_2 = \bigcup_{j \in J_2} x^j(v),$$

$$x^j(v) = \{\overline{x} | \langle (\overline{x} - \mu^j), (\sigma^{-j})^{-1} \cdot (x - \mu^j) \rangle \leq r^2, \;\; j \in J_1,$$

$$\text{pak} \quad \varepsilon_0 = \int_r^\infty \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} dx. \tag{15}$$

3 Stručně o perceptronu

Diskriminační funkce může být zobecněna nahrazením lineární funkce

$$g_k(x) = \overline{w}_k \overline{x} + w_{k0} \tag{16}$$

monotónní funkcí aplikovanou na lineární kombinaci $\overline{w}$ a $\overline{x}$

$$g_k(x) = f(\overline{w}_k \overline{x} + w_{k0}). \tag{17}$$

Perceptron je jednovrstvá neuronová síť s prahovou aktivační funkcí, která se většinou volí jako lichá:

$$f(a) = \begin{cases} 1 & \text{pro a>0} \\ -1 & \text{pro a<0,} \end{cases}$$

což odpovídá funkci *sign*. Funkce f se nazývá aktivační funkce.

3.1 Perceptronový algoritmus

3.1.1 Adaptační fáze (učení)

Uvažujme dvě třídy, označíme je 1 a -1 t.j. $K = \{-1, 1\}$ (lze zobecnit na libovolný počet). Dále známe trénovací multi-množinu

$$T = ((\overline{x}_1, k_1), (\overline{x}_2, k_2), (\overline{x}_3, k_3), \ldots, (\overline{x}_n, k_n)),$$

což je vstup perceptronového algoritmu. Pak na počátku (v čase $t = 0$) uvažujeme $\overline{w}^0 = (0, 0, 0, \ldots, 0)$. V čase $t + 1$ klasifikujeme vstupní vektor podle $k_t(\overline{w}^t \overline{x}_t)$. Pokud je výsledek klasifikace nenulové kladné číslo, byla klasifikace správná a nedojde k žádné změně

$$\overline{w}^{t+1} = \overline{w}^t. \tag{18}$$

Pokud je výsledkem nekladné číslo ($\leq$0), nebyla klasifikace správná a vektor vah musíme upravit podle vztahu

$$\overline{w}^{t+1} = \overline{w}^t + k_t \overline{x}_t. \tag{19}$$

Potom musíme projít celou trénovací multi-množinu znovu, abychom měli jistotu, že nová rozdělující nadrovina nezpůsobila chybnou klasifikaci jiného prvku.

3.1.2 Algoritmus učení perceptronu:

1. Inicializuj váhy $\overline{w}^0 = 0$ (horní index označuje číslo kroku).
2. Nalezni v trénovací množině špatně klasifikovaný vektor (označ ho $\overline{x}_t$ a jeho váhu w^t), pro který platí $k_t(\overline{w}^t\overline{x}_t) \leq 0$.
3. Pokud nebyl nalezen žádný špatně klasifikovaný vektor, jsou váhy adaptovány a algoritmus se ukončí. Byl-li takový vektor nalezen, uprav vektor vah podle pravidla $(\overline{w}^{t+1} = \overline{w}^t + k_t\overline{x}_t)$.
4. Přejdi na krok 2.

3.1.3 Pracovní fáze

Za předpokladu, že bylo dokončeno učení perceptronu, tj. známe vektor vah $\overline{w}$ a práh w_0, klasifikujeme vstupní vektor $\overline{x}$ dosazením do vztahu $k_t(\overline{w}^t\overline{x}_t + w_0)$. Pokud je vnitřek závorek (tzv. potenciál neuronu) větší nebo roven nule, je vektor $\overline{x}$ klasifikován do třídy $k = 1$, v opačném případě je klasifikován do třídy $k = -1$.

3.1.4 Novikoffova věta

Je-li T (trénovací multi-množina) lineárně separabilní (tj. existuje w' : $k_i(\overline{w}'\overline{x}_i) > 0, i = 1, \ldots, n$), pak perceptronové učení konverguje, tzn. po čase t_s pro $t > t_s$

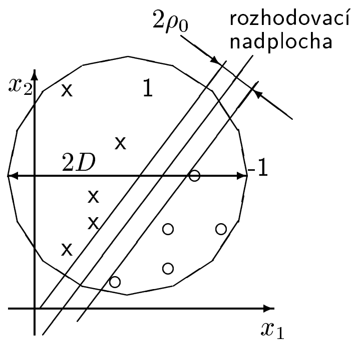
$$\overline{w}^t = \overline{w}^{ts}, s < \frac{D^2}{\rho_0^2}, \tag{20}$$

kde D je poloměr nejmenší opsané koule dat x_i tj. $|\overline{x}_i| < D$,

ρ_0 je minimální vzdálenost prvku od separační nadplochy $\rho_0 \leq \min_i k_i \frac{\overline{w}_0\overline{x}_0}{|\overline{w}_0|}$,

w_0 určuje optimální rozdělující nadplochu.

t je počet kroků algoritmu



Obrázek 6: Separabilní data v konvexním obalu

Důkaz 1

1. $\overline{\omega}_0 = \overline{0}$

2. *Hledáme horní odhad velikosti* $|\overline{w}^t|$. *Velikost* $\overline{w}^t$ *roste pouze při korekci, maximálně o* D^2 $|\overline{w}^t|^2 = |\overline{w}^{t-1}|^2 + \underbrace{2k_t(\overline{x}_t\overline{w}^{t-1})}_{\leq 0} + |x_t|^2$ $|x_t|^2 \leq D^2$ *(z definice D)* $\Rightarrow |\overline{w}^t|^2 \leq |\overline{w}^{t-1}|^2 + D^2$ *do času t došlo k maximálně t korekcím a tudíž*

$$\boxed{|\overline{w}^t| \leq tD^2},$$

neboť $|\overline{w}^0| = 0$.

3. $\exists$ *vektor* $\overline{w}_0$: $\dfrac{k_i(\overline{w}^0\overline{x}_i)}{|\overline{w}^0|} \geq \rho_0$ *(předpoklad věty)*

$$\frac{\overline{w}^t\overline{w}^0}{|\overline{w}^0|} = \frac{\overline{w}^{t-1}\overline{w}^0}{|\overline{w}^0|} + \frac{k_t(\overline{x}_t\overline{w}^0)}{|\overline{w}^0|} > \frac{\overline{w}^{t-1}\overline{w}^0}{|\overline{w}^0|} + \rho_0 \geq t\rho_0$$

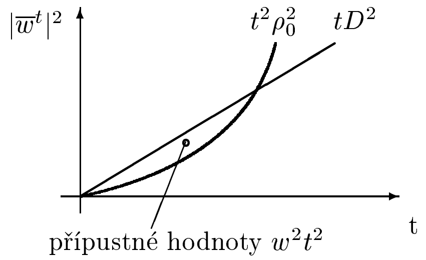
Schwarzova - Cauchyova nerovnost

$$|\overline{w}^t||\overline{w}^0| \geq \overline{w}^0\overline{w}^t$$

$$\boxed{|\overline{w}^t| \geq \frac{\overline{w}^t\overline{w}^0}{|\overline{w}^0|} > t\rho_0}$$

$t^2\rho_0^2 \leq |\overline{w}^t| \leq tD^2$

$$\boxed{t \leq \frac{D^2}{\rho_0^2}}$$



Obrázek 7: Geometrické znázornění konvergence

3.2 Problém neseparabilnosti dat

Neseparabilní data mají za následek složitou analýzu a problém minimalizace chyby (počet chybně zařazených vzorů) je NP úplná úloha. Proto je třeba se spokojit s pravděpodobnostním algoritmem.

Řešíme následující úlohu: Jak zastavit učení perceptronu, aby s pravděpodobností $1-\eta$ byla chyba maximálně ε.

$\Rightarrow$ Zavedeme zastavovací pravidlo typu „ukonči učení po k krocích, jestliže $m(k)$ vzorků za sebou je klasifikováno správně".

Pravděpodobnost, že učení bude zastaveno, i když pravděpodobnost chyby $P^{\varepsilon}(\overline{w}_k) > \varepsilon$ po $m(k)$ úspěšných klasifikacích, je

$$P_k = \Big(1 - P(w_k)\Big)^{m(k)} < (1-\varepsilon)^{m(k)}. \tag{21}$$

K této chybě může dojít v kterémkoli kroku. Celková pravděpodobnost zastavení i při chybě větší než ε je

$$P < \sum_{k=1}^{\infty} P_k < \sum_{k=1}^{\infty} (1-\varepsilon).^{m(k) \leftarrow \text{můžeme volit}} \tag{22}$$

Počet správně klasifikovaných vzorků tedy musí být

$$m(k) = \left[\frac{2\ln K - \ln\eta + \ln\frac{\pi^2}{6}}{-\ln(1-\varepsilon)}\right] + 1. \tag{23}$$

4 Kozincův algoritmus

4.1 Popis

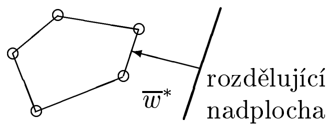
Kozincův algoritmus hledá vektor $\overline{w}^*$ a práh θ, pro které platí:
$\overline{w}^* \cdot \overline{x} \geq \theta$ pro každé $\overline{x}$ z první třídy,
$\overline{w}^* \cdot \overline{x} < \theta$ pro každé $\overline{x}$ ze druhé třídy.

Nalezení $\overline{w}^*$ a θ určuje rozhodovací nadplochu v n-dimenzionálním prostoru. Algoritmus pracuje iterativně a pokud jsou vstupní třídy lineárně separabilní, pak se zastaví po konečném počtu kroků. Některé implementace dovolují omezit maximální počet kroků a začít z definovaného stavu daného hodnotami $\overline{w}^*$ a θ. Tento algoritmus je podobný učícímu algoritmu perceptronu.
Problém nalezení optimálního rozdělení řešíme kritériem:

$$\overline{w}^* = \arg\max_{\overline{w}} \min_{\overline{x} \in \overline{X}} \left(\frac{\overline{w}}{\|\overline{w}\|}, overlinex \right). \tag{24}$$

Toto kritérium lze převést na hledání nejbližšího bodu z konvexního obalu

$$\overline{w}^* = \arg\min_{\overline{x} \in \overline{X}} \|\overline{x}\|.$$



Obrázek 8: Geometrická představa

Platí:

$$\underbrace{\min_{\overline{x} \in \overline{X}} \left(\frac{\overline{w}}{\|\overline{w}\|}, \overline{x}_i \right)}_{\text{dolní mez}} \quad \leq \rho_0 \leq \quad \underbrace{\|\overline{w} \in \overline{X}\|}_{\text{horní mez}},$$

kde ρ_0 je vzdálenost nejbližšího bodu od rozdělující nadroviny.

4.2 ε – optimální řešení

Pokud upřednostňujeme minimalizaci strukturálního rizika (vyloučení náhodných "šumů"), můžeme formulovat úlohu hledání rozdělující nadplochy tak, že ne všechny body budou klasifikovány správně. Zvolíme určité ε - okolí rozdělující nadplochy, na kterém mohou být body špatně klasifikovány.
Potom kritérium (24) přechází na:

$$\|\overline{w}\| - \left(\frac{\overline{w}}{\|\overline{w}\|}, \overline{x} \right) \leq \varepsilon. \tag{25}$$

4.3 Vstupy algoritmu

$\mathbb{X}$	$[NM]$	- matice obsahující M bodů trénovací množiny v N-dimenzionálním prostoru. $\mathbb{X} = [\overline{x}_1, \overline{x}_2, ..., \overline{x}_M]$, kde $\overline{x}_i$ je i-tý sloupcový vektor
J	$[1M]$	- vektor obsahující zařazení bodů z $\mathbb{X}$ do tříd
t_{max}	$[11]$	- maximální počet kroků algoritmu

t $[11]$, $\overline{w}$ $[N1]$, θ $[11]$ - počáteční podmínky algoritmu

4.4 Výstupy algoritmu

$\overline{w}$	$[N1]$	- normálový vektor hledané rozhodovací nadplochy
θ	$[11]$	- práh rozhodovací nadplochy
solution		- rovno jedné pokud je řešení nalezeno, rovno nule pokud není
t	$[11]$	- obsahuje údaj o počtu provedených kroků

4.5 Algoritmus

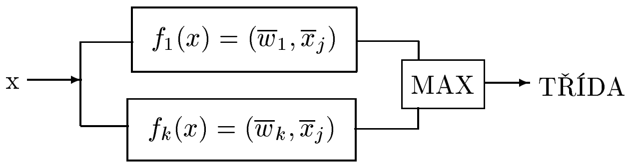
- zadání vstupních parametrů
- transformace původního charakteristického prostoru do homogenních souřadnic ($\theta = 0$)
- první krok v čase $t = 0$ - inicializace $\overline{w}$ libovolným vektorem z $\mathbb{X}$
- vrátí první špatně klasifikovaný bod, který by byl použit pro změnu hodnoty *solution*:
$$(\overline{w}^t, \overline{x}_t < 0)$$
- iteruje dokud není nalezeno řešení nebo dosaženo limitního počtu kroků
  - předpokládá, že řešení existuje ($solution = 1$)
  - hledá vektor $\overline{x}$ z $\mathbb{X}$, pro který $\overline{w} \cdot \overline{x} \leq 0$
  - podle něj upraví $\overline{w}$
- transformuje nalezené řešení z homogenních souřadnic do původního prostoru

Modifikace bodu 4) algoritmu pro ε- optimální řešení:

$$||\overline{w}|| - \min_t \Big(\frac{\overline{w}^t}{||\overline{w}^t||}, \overline{x}_t\Big).$$

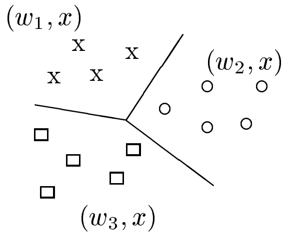
5 Fischerův klasifikátor

Problém učení klsifikátoru pro k tříd lze převést na problém učení pro jednu třídu



Obrázek 9: Fischerův klasifikátor

Mějme trénovací množinu $\{(\overline{x}_1, k_1); (\overline{x}_2, \overline{k}_2); \ldots\}$. Úlohou, kterou zde řešíme je nalezení všech vektorů $w_1 \ldots w_k$.



Obrázek 10: Trénovací množina

Takovou úlohu lze převést na problém

$$(\overline{w}, \overline{y}) > 0, \text{ pro } \overline{y} \in \mathbb{Y}, \text{ kde} \tag{26}$$

$$\mathbb{Y} = \begin{array}{l} \{[\overline{x}_j, -\overline{x}_j, 0, \ldots, 0], \\ \ [\overline{x}_j, 0, -\overline{x}_j, 0, \ldots, 0], \\ \ [\overline{x}_j, 0, \ldots, 0, -\overline{x}_j]\} \end{array} \quad \begin{array}{l} \overline{w} = [w_1, w_2, \ldots, w_k] \\ |Y| = (k-l) \cdot l \end{array}$$

Příklad transformace:
Vybere jeden bod z třídy 3 a zapíšeme podmínky pro $w_1 \ldots w_3$:

$$\begin{array}{r} (w_3, x_j) > (w_1, x_j) \\ \underline{(w_3, x_j) > (w_2, x_j)} \\ -(w_1, x_j) + (w_3, x_j) > 0 \\ \underline{-(w_2, x_j) + (w_3, x_j) > 0} \\ \overline{y}_1 = [-x_j, 0, x_j] \quad (w_3, y_1) > 0 \\ \underline{\overline{y}_2 = [0, -x_j, x_j] \quad (w_2, y_2) > 0} \end{array}$$

Takto formulovanou úlohu již řeší např. perceptronový nebo Kozincův algoritmus.
Pozn.: Fisherův klasifikátr můžeme použít pro zobecenění SVM na více tříd.

Reference

[1] Adéla Klimentová, Libor Beneš, Dušan Hlaváč:
Zápis z přednášky. FEL, Kybernetika 2001
(http://cmp.felk.cvut.cz/cmp/courses/recognition/zapis_01/5/rpz5.pdf)

[2] Michail L Schlesinger, Václav Hlaváč: *10 přednášek z rozpoznávání.*
Vydavatelství ČVUT, Praha 1999