Lineární diskriminační funkce aperceptron,

zápis 5. přednášky

Adéla Klimentová, Libor Beneš, Dušan Hlaváč

6. června 2001

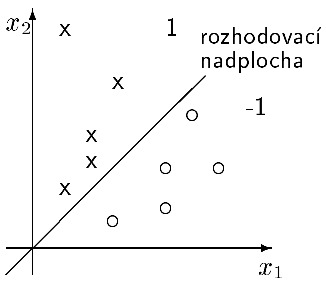
Obsah

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 nám rozděluje množinu všech vstupních bodů na 2 množiny. Určení, do které z množin vybraný prvek patří, se určuje pomocí LDF a souřadnic daného prvku. Výsledkem rozhodnutí je např. 1 pro prvky nad LDF a -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)

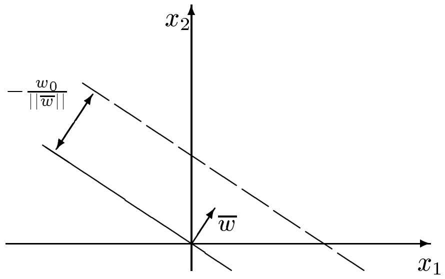
- některé nelineární diskriminační funkce lze vyjádřit jako lineární po tzv. "vyrovnání příznakového prostoru"

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

- existuje universalní způsob zobrazení příznakového prostoru do prostoru pravděpodobností, kde se řešení každé statistické úlohy realizuje lineární funkcí, důkaz existence LDF pro nezávislé binární přiznaky (viz. rovn. 5 str.5)

- kapacita (VC dimenze) lineárních strategií v n-rozměrném prostoru je$n+2$ a <u>učení</u> je korektní
- existují postupy pro nalezení linearních diskriminačních funkcí
  - Perceptron
  - Kozineův algoritmus
  - Support vector machine

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



Obrázek 2: Rozhodovací nadrovina, normálový vektor

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

kde $\overline{x}$ je vstupní vektor
$\overline{w}$ je normálový vektor k rozhodovací nadrovině (obr. 2)
w_{k0} je posunutí rozhodovací nadroviny

Hledaná funkce $d(x)$ potom bude:

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

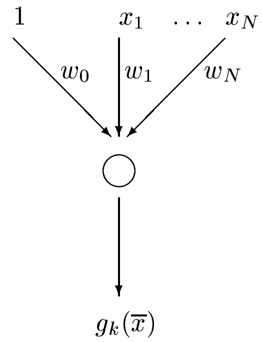
což je stejné jako $d(x) = k^* \Longleftrightarrow g_k^*(\overline{x}) \geq g_k(\overline{x}), k \in K$

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.

Lineární klasifikátor může být představován jako jednoduchý neuron s váženými vstupy a jedním výstupem. Konstantu posunutí $\omega_0$lze obsáhnout na začátek nomálového vektoru k rozhodovací nadrovině, pokud vstupní vektor taktéž na začátku rozíříme o 1. Tím získáme $N+1$vstupů $\overline{x}$a $N+1$vah $w_0 \ldots w_N$.

Pro dvě třídy jsou diskriminační funkce dány vztahem:
$g_1(\overline{x}) = \overline{w}_1\overline{x} + w_{10}$

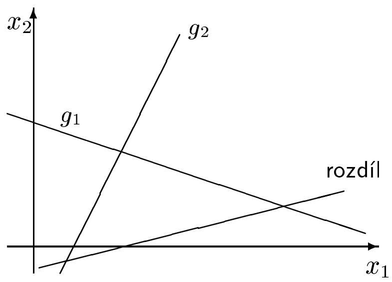
$g_2(\overline{x}) = \overline{w}_2\overline{x} + w_{20}$



Obrázek 3: Jednoduchý neuron

Pro porovnání, která funkce je pro dané x větší, nám stačí najít vektor rozdílů:

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



Obrázek 4: Diskriminační funkce pro dvě třídy jako funkce rozdílu

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

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 ($i = 1, 2, \dots, N$)

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

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

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:

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

1.1 2D Gaussovo rozložení tříd

$$p(x|k) = \frac{1}{(2\pi)^{\frac{d}{2}}|\mathbb{C}|^{\frac{1}{2}}} e^{-\frac{1}{2}\left((x-\mu_k)^T\mathbb{C}^{-1}(x-\mu_k)\right)} \tag{6}$$

$p(x\|k)$	...	sdružená pravděpodobnost jevu, že objekt je ve stavu k a pozorujeme příznak x
$\mathbb{C}$	...	kovarianční matice určující vzájemný posun Gaussovských rozdělení
d	...	dimenze příznakového prostoru

Výraz upravíme (pomocí ln):

$$\ln p(x|k) = \left(-\frac{1}{2}(x-\mu_k)^T\mathbb{C}(x-\mu_k)\right) + \ln \mathbb{C}_1 = \overline{wx} + w_0 \tag{7}$$

A úprava vede na lineární funkci $\overline{x}$.

2 Kozineův algoritmus

2.1 Popis

Kozineův algoritmus hledá vektor α a práh θ, pro které platí:
$\alpha' x \geq \theta$ pro každé x z první třídy
$\alpha' x < \theta$ pro každé x ze druhé třídy.

Nalezení α 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 α a θ. Tento algoritmus je podobný učícímu algoritmu perceptronu.

2.2 Vstupy algoritmu

X $[NM]$ - matice obsahující M bodů trénovací množiny v N-dimenzionálním prostoru. $X = [x_1, x_2, ..., x_M]$, kde x_i je i-tý sloupcový vektor
J $[1M]$ - vektor obsahující zařazení bodů z X do tříd
$t_{max}[11]$ - maximální počet kroků algoritmu

t $[11]$
α $[N1]$
θ $[11]$ - počáteční podmínky algoritmu

2.3 Výstupy algoritmu

α $[N1]$ - 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

2.4 Algoritmus

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

3 Perceptron

Diskriminační funkce může být zobecněna nahrazením lineární funkce

$$g_k(x) = \overline{w}_k\overline{x} + w_{k0} \tag{8}$$

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

$$g_k(x) = f(\overline{w}_k\overline{x} + w_{k0}). \tag{9}$$

Funkce f se nazývá aktivační funkce.

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

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

což odpovídá funkci sign.

4 Perceptronový algoritmus

4.1 Adaptační fáze (učení)

Uvažujme 2 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 kladné číslo (>0), byla klasifikace správná a nedojde k žádné změně

$$\overline{w}^{t+1} = \overline{w}^t. \tag{10}$$

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

$$\overline{w}^{t+1} = \overline{w}^t + k_t x_t. \tag{11}$$

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

4.2 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 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 $\overline{w}^{t+1} = \overline{w}^t + k_t x_t$.
4. Přejdi na krok 2.

4.3 Pracovní fáze

Za předpokladu, že máme naučen perceptron, 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 $k = 1$ a v opačném případě je klasifikován do třídy $k = -1$.

4.4 Novikoffova věta

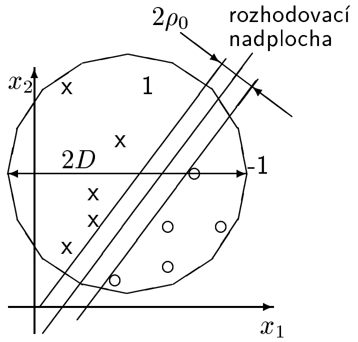
Je-li T (trénovací multi-množina) lineárně separabilní (tj. existuje w' : $k_i(\overline{w}'\overline{x}_i) > 0, i = 1, \ldots, n$), pak perceptronové učení konverguje, tzn. po časet_s(pro$t > t_s$)

$$\overline{w}^t = \overline{w}^{ts}, s < \frac{D^2}{\rho_0^2} \tag{12}$$

kde D je poloměr nejmenší opsané koule datům 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



Obrázek 5: 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)*

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

Swarz - Cauchyova nerovnost

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

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

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

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

5 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

$$P_k = \Big(1 - P(w_k)\Big)^{m(k)} < (1-\varepsilon)^{m(k)}. \tag{13}$$

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

$$P < \sum_{k=1}^{\infty} P_k < \sum_{k=1}^{\infty} (1-\varepsilon).^{m(k) \leftarrow \text{můžeme volit}} \tag{14}$$

Počet správně klasifikovaných vzorků tedy musí být

$$m(k) = \Big[\frac{2\ln K - \ln m + \ln \frac{\pi^2}{6}}{-\ln(1-\varepsilon)}\Big] + 1. \tag{15}$$

6 Příklad

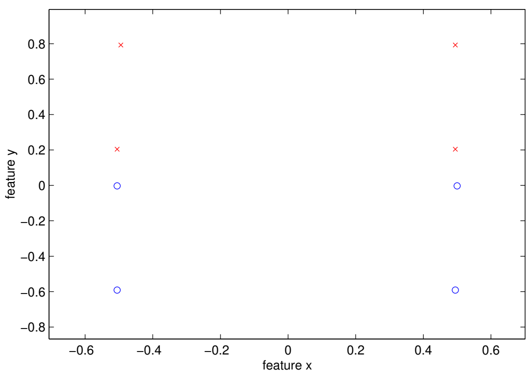
6.1 Použitý algoritmus

Příklad hledá lineární diskriminační funkci pomocí perceptronového algoritmu. Pod každým grafem je uvedeno, co se právě provádí, jaký je aktuální tvar LDF a který bod tomuto tvaru nevyhovuje. Navíc špatně klasifikovaný bod je označen kroužkem. Pokud v grafu žádná LDF není, nevešla se do zobrazené výseče.

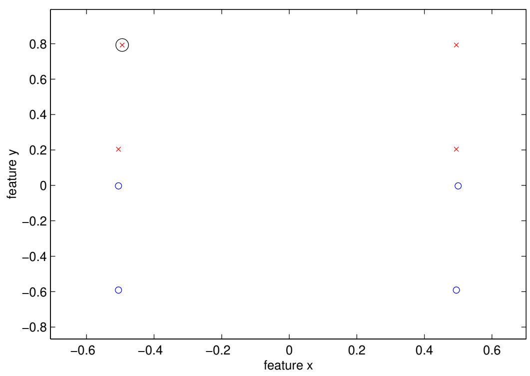
6.2 Počáteční rozmístění bodů

$$\begin{array}{llllllll}
x_1 = & [& -0.4945; & 0.7927 &] & x_2 = & [& 0.4945; & 0.7927 &] \\
x_3 = & [& -0.5055; & 0.2045 &] & x_4 = & [& 0.4945; & 0.2045 &] \\
x_5 = & [& -0.5055; & -0.0028 &] & x_6 = & [& 0.5000; & -0.0028 &] \\
x_7 = & [& -0.5055; & -0.5910 &] & x_8 = & [& 0.4945; & -0.5901 &]
\end{array}$$

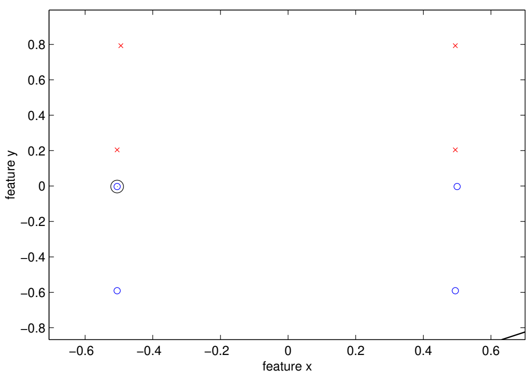
6.3 Postup učení



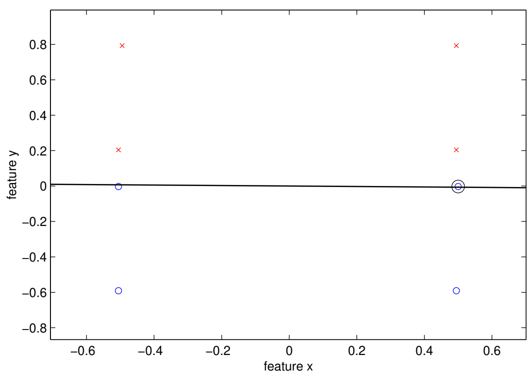
Počáteční rozmístění 8 prvků



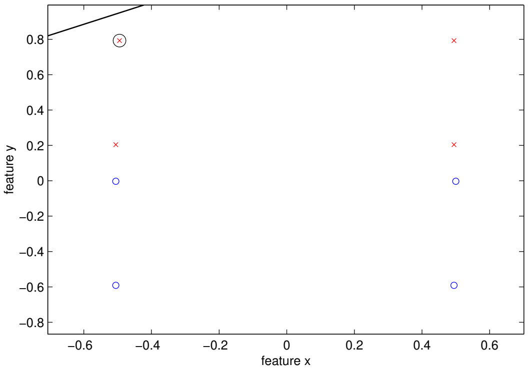
1. krok: funkce 0x + 0y = 0, špatně klasifikován bod x1



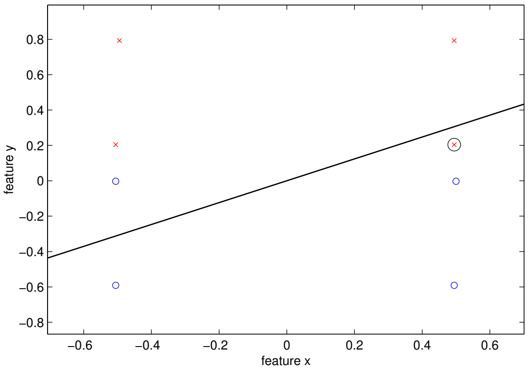
2. krok: funkce -0.49x + 0.79y = -1, špatně klasifikován bod x5



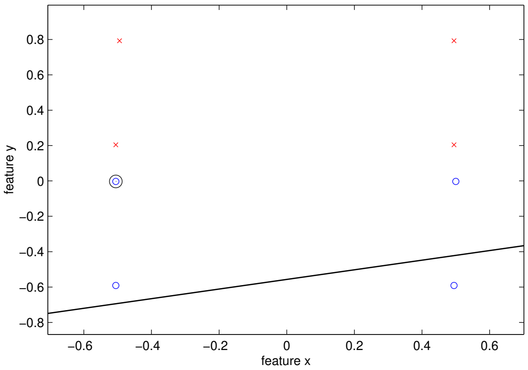
3. krok: funkce -0.01x + 0.79y = 0, špatně klasifikován bod x6



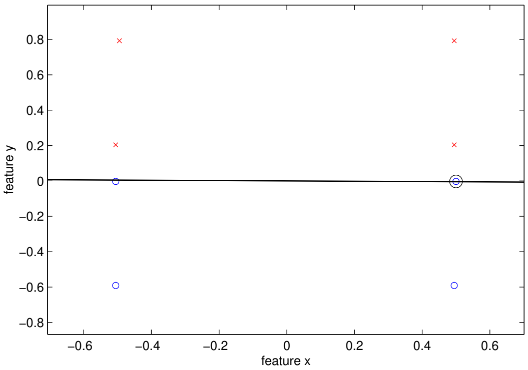
4. krok: funkce -0.49x + 0.6y = 1, špatně klasifikován bod x1



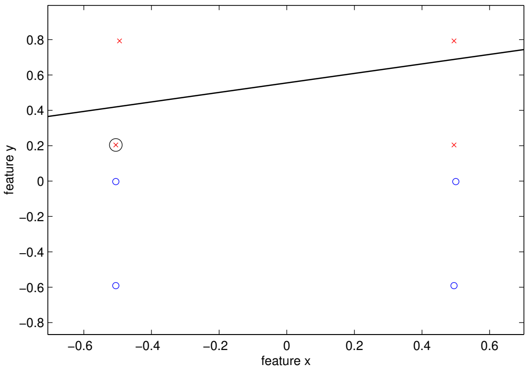
5. krok: funkce -0.96x + 1.59y = 0, špatně klasifikován bod x4



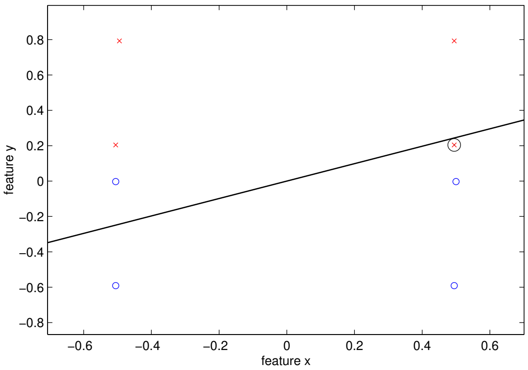
6. krok: funkce -0.49x + 1.6y = -1, špatně klasifikován bod x5



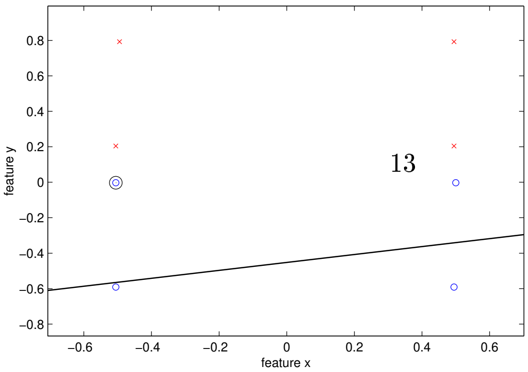
7. krok: funkce 0.02x + 1.6y = 0, špatně klasifikován bod x6



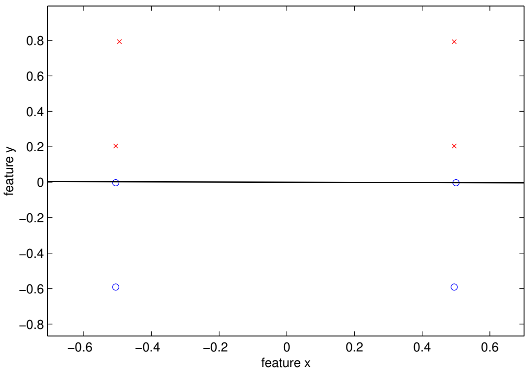
8. krok: funkce -0.46x + 1.6y = 1, špatně klasifikován bod x3



9. krok: funkce -x + 2y = 0, špatně klasifikován bod x4



10. krok: funkce -0.5x + 2.2y = -1, špatně klasifikován bod x5



Výsledná rozhodovací funkce 0.01x + 2.21y = 0.