

# Vytěžování dat, přednáška 9:

## Lineární klasifikátor, rozšíření báze, LDA, logistická regrese

Miroslav Čepek

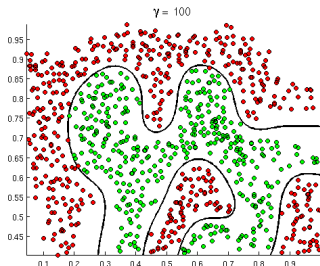


Evropský sociální fond  
Praha & EU: Investujeme do vaší budoucnosti

*Fakulta elektrotechnická, ČVUT*

- ▶ Jaké znáte klasifikační algoritmy?
  - ▶ Naive Bayes
  - ▶ Rozhodovací strom
  - ▶ KNN
- ▶ Jak fungují?
  - ▶ Naive Bayes počítá pravděpodobnost přiřazení do třídy za podmínky dané pozorováním.
  - ▶ Rozhodovací strom generuje posloupnost rozhodnutí.
  - ▶ KNN hledá nejbližší instance z trénovací množiny.

- ▶ Další možný pohled je z hlediska rozhodovací hranice.
- ▶ Rozhodovací hranice je místo, kde instance přestávají patřit do třídy A a začínají patřit do třídy B.

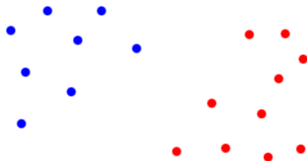


Převzato z [http:](http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex8/ex8.html)

[//openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex8/ex8.html](http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex8/ex8.html)

## Rozhodovací hranice (2)

- ▶ Co když mám následující data:
- ▶ Jak bude vypadat nejjednodušší rozhodovací hranice, která data oklasifikuje bez chyby?
- ▶ Stačí přece přímka!



- ▶ Takovou přímku umíme celkem jednoduše najít a vyrobit :).
- ▶ Jak vypadá rovnice přímky?

$$y = \sum_j w_j x_j = \overline{w}x$$

- ▶ Jak zjistíme, do které třídy patří vzor  $\bar{x}$ ?
- ▶ Prostým dosazením! Pokud  $y > 0$  vzor patří do jedné třídy,  $y < 0$  vzor patří do druhé třídy.
- ▶  $y = 0$  je rozhodovací hranice, kde se nelze rozhodnout. Naštěstí se toto nestává často.
- ▶ Porovnání  $< 0$  nebo  $> 0$  můžeme nahradit například funkcí *signum*( $y$ ) která vrací  $+1$  pro kladná čísla a  $-1$  pro záporná.

- ▶ Vidíte v rovnici nějaký problém?
- ▶ Zkusme dosadit do rovnice  $x_1 = x_2 = 0$ . Co vyjde?
- ▶  $y = 0$  bez ohledu na nastavení vah  $w_1, w_2$ .
- ▶ Rozhodovací hranice libovolného klasifikátoru tedy musí procházet nulou resp. vektorem  $(0, 0, \dots, 0)$ .
- ▶ To je docela nepříjemné omezení – rozhodovací hranice se vlastně pouze otáčí kolem nuly. Co s tím?
- ▶ Můžeme přidat změnit rovnici přímky na  $\sum_j w_j x_j + \Theta = 0$ .
- ▶  $\Theta$  je aditivní konstanta a označuje se jako práh.

- ▶ Rovnice pro lineární klasifikátor se tedy mírně změní:

$$y = \sum_j w_j x_j + \Theta = \overline{w}x + \Theta$$

- ▶ Ale jinak vše zůstane stejné.
- ▶ Pokud tedy vyjde, že  $y > 0$  vzor patří do jedné třídy,  $y < 0$  vzor patří do druhé třídy.

- ▶ Teď zbývá pouze najít vektor koeficientů (vah)  $(w_1, w_2, \dots, w_n)$  a prahu  $\Theta$ .
- ▶ Nejprve se zamysleme zda práh  $\Theta$  představuje nějakou anomálii, která potřebuje speciální zacházení.
- ▶ Tak napůl :). Nepotřebuje, pokud trochu upravíme vstupní data.
- ▶ Co kdyby všechny vzory měli jeden atribut (vstupní proměnnou, řekněme  $x_0$ ) se stejnou hodnotou?
- ▶ Když pak budu počítat  $w_0 x_0$ , vyjde mi vždy stejná hodnota... to je přece náš práh!
- ▶ Takže přidám ke každému vzoru ještě jednu dimenzi, která bude mít konstantní hodnotu – pro jednoduchost 1 (ale můžete mu přiřadit libovolnou nenulovou hodnotu).
- ▶ Pak už se o práh nemusím speciálně starat.



- ▶ Gradientní sestup je iterativní optimalizační metoda.
- ▶ Jde o to, že máme funkci  $f(\bar{x})$  a hledáme její minimum.
- ▶ A jak minimum najít? Budu hledat takové  $\bar{x}$ , kde má funkce  $f(\bar{x})$  nejnižší hodnotu.
- ▶ Mám nějaké  $\bar{x}_0$  a znám hodnotu  $f(\bar{x}_0)$ . A hledám v okolí  $\bar{x}_0$ , nějaké  $\bar{x}_1$ , kde bude hodnota  $f(\bar{x}_1)$  nižší. A to budu opakovat stále dokola, až to dál nepůjde.
- ▶ "Na horách se chci dostat do údolí. Podívám se, kterým směrem se kopec nejvíc snižuje a udělám krok tím směrem. A zase se podívám, kam je to nejvíc z kopce a udělám krok. A tak dále, až jednou zjistím, že to dál nejde, čili jsem v údolí."

- ▶ V matematice existuje možnost, jak zjistit směr největšího vzestupu hodnoty funkce - gradient. To je první parciální derivace podle všech dimenzí. Když půjdu přesně opačně, půjdu ve směru největšího poklesu.
- ▶ Gradient je vektor parciálních derivací podle jednotlivých proměnných:

$$\text{grad}(f(x_1, x_2, \dots, x_n)) = \left( \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right)$$

- ▶ Stejně jako u jiných klasifikačních metod, hledáme nastavení klasifikátoru (zde vah) takové, aby klasifikoval správně co nejvíc vzorů.
- ▶ Mějme tedy chybovou funkci  $J(\bar{w}, \chi)$ , která vrací počet špatně klasifikovaných vzorů  $\bar{x}$  z množiny vzorů  $\chi$  (například z testovací množiny) při daných vahách  $\bar{w}$ .
- ▶ Ale taková se nám nehodí – neumím spočítat parciální derivace (a tudíž ani gradient).
- ▶ Zkusme jinou –  $E(\bar{w}) = \sum_{x \in \chi^{err}} (-w(t)x)$ , kde  $\chi^{err}$  jsou špatně klasifikované vzory z  $\chi$ .
- ▶ To už je spojitá funkce.
- ▶ A chybovou funkci  $E$  mohu minimalizovat změnou vah – pomocí gradientního sestupu.

# Chybová funkce a gradientní sestup

- Parciální derivace chybové funkce  $E$  podle jedné proměnné vypadá takto:

$$\frac{\partial E}{\partial w_1} = \frac{\partial \sum_{x_1 \in \chi^{err}} (-w_1(t)x_1)}{\partial w_1}$$

- Kde  $x_1$  je první souřadnice z  $\bar{x}$  a  $w_1$  je první souřadnice z  $\bar{w}$ .

$$\frac{\partial \sum_{x_1 \in \chi^{err}} (-w_1(t)x_1)}{\partial w_1} = \sum_{x_1 \in \chi^{err}} x_1$$

- Z toho plyne závěr, že nejvíce chybu snížím, pokud váhu  $w_1$  změním o součet hodnot prvních vstupních proměnných  $x_1$  vzorů na kterých jsem udělal chybu.
- Analogicky i ostatní vstupní proměnné (včetně prahu).

- ▶ A přesně na této myšlence je založen **Perceptronový algoritmus**. (Proč zrovna perceptronový, necháme teď stranou).

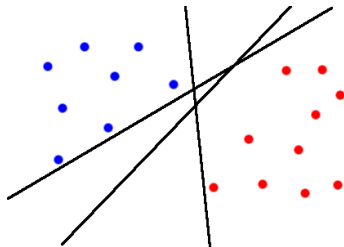
1. Inicializuj váhy na náhodné hodnoty.
2. Vezmi náhodný vstupní vzor  $\bar{x}$ . A spočítej hodnotu  $y(\bar{x}) = \text{sgn}(\sum_j w_j x_j = \overline{w\bar{x}})$ .
3. Pokud se požadovaný výstup  $d(\bar{x})$  liší od skutečného  $y(\bar{x})$ , oprav váhy ( $w$ ) takto:
  - ▶  $e(\bar{x}) = d(\bar{x}) - y(\bar{x})$
  - ▶  $w_i(t+1) = w_i(t) + \eta e(\bar{x}) x_i$
4. Pokud chceš, pokračuj bodem 2).

$\eta$  je koeficient učení, který s postupem času může klesat k nule.

<http://neuron.felk.cvut.cz/courseware/data/chapter/36nan028/s04.html>

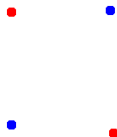
# Jednoznačnost výsledné přímky

- ▶ Jak poznám, že se perceptronový algoritmus zastavil?
- ▶ Je výsledná přímka jediná možná?
- ▶ Ne, takových přímek je dokonce nekonečně mnoho.
- ▶ Přesto jsou některé lepší než jiné.



- ▶ Kterou mám vybrat? A kterou nám vybere Perceptronový algoritmus?

- ▶ Lze pomocí tohoto algoritmu klasifikovat bezchybně následující data?

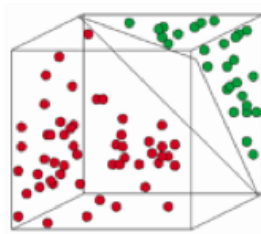
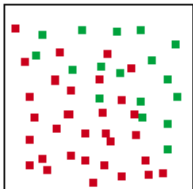


- ▶ Ne!
- ▶ Třídám (datům), které lze oddělit přímkou říkáme lineárně separovatelná.

- ▶ Lze pomocí lineárního klasifikátoru vyřešit lineárně ne-separovatelné problémy?
- ▶ Ne. K tomu potřebuji přidat další kvalitu.
  1. Jednou z možností je zkombinovat lin. klasifikátory. (O tom bude příští přednáška.)
  2. Rozšíření báze.



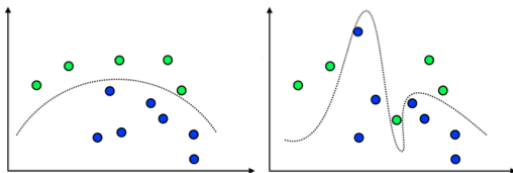
- Když data nejsou separabilní v originálním prostoru, zkusme jestli by nebyla lineárně separovatelná ve více-dimenzionálním prostoru.



- ▶ Zase existuje mnoho transformací.
- ▶ Jedna z těch jednodušších přidává ke stávajícím dimenzím také součiny originálních dimenzí.
- ▶ Je charakterizovaná parametrem  $s$  – maximálním stupněm součinu.
- ▶ Pro  $s = 2$  z původních dimenzí  $(x_1, x_2)$  získám 5D prostor dimenzí  $(x_1, x_2, \chi_1, \chi_2, \chi_3)$ , kde
  - ▶  $\chi_1 = x_1^2$
  - ▶  $\chi_2 = x_1 * x_2$
  - ▶  $\chi_3 = x_2^2$

- ▶ Když mám teď nové dimenze, jak naučím můj klasifikátor?
- ▶ Díky rozšíření báze teď mám více-dimenzionální prostor a (třeba) teď dokáží třídy lineárně separovat.
- ▶ Úplně stejně jako v předchozím případě. Tj použiji například perceptronový algoritmus.
- ▶ Pokud se mi nepodaří získat lineární separaci, mohu zkusit zvýšit maximální stupeň součinu a zkusit separaci znovu.

- Jak pak bude vypadat projekce rozhodovací hranice zpět do originálních dimenzí?



- Tím se nám podařilo dosáhnout toho, že se rozhodovací hranice "ohnula".
- A pokud budu zvyšovat parametr  $s$ , bude hranice stále "poddajnější".

- ▶ Můžu zvyšovat maximální stupeň součinu (parametr  $s$ ) do nekonečna? Nebo někde narazím?
- ▶ S vyšším maximálním stupněm  $s$  vyvstávají dva problémy:
  1. Jednak mám více vah, které musím nastavovat a perceptronový (nebo jakýkoliv jiný) algoritmus to nemusí zvládnout – čili váhy se nastaví špatně.
  2. Zvyšuji pravděpodobnost, že se klasifikátor přeučí. Tj. naučí se šum a chyby v trénovacích datech a nedokáže vlastnosti trénovacích dat zobecnit. (Podobně jako například 1-NN klasifikátor.)

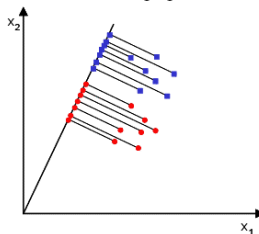
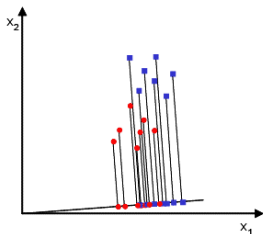
- ▶ Co když mám více než dvě třídy? Mohu použít (ne-)lineární separaci?
- ▶ Případně jak?
- ▶ Budu rozhodovat o každé třídě samostatně.
- ▶ Čili pro problém s  $N$  třídami budu mít  $N$  klasifikátorů. Každý bude říkat – "Instance patří do mé třídy" / "Instance **nepatří** do mé třídy".
- ▶ Při rozhodování o neznámé instanci  $x$  se zeptám všech  $N$  klasifikátorů na jejich rozhodnutí.
  1.  $x$  patří právě do jedné třídy.
  2.  $x$  patří do více tříd – pak se musím rozhodnout například podle hodnoty  $\overline{xw}$  (čím je větší, tím je  $x$  dále od rozhodovací hranice).
  3. analogoicky, pokud  $x$  nepatří do žádné třídy. Hledám tedy třídu, kam  $x$  nepatří nejméně :).

- ▶ Linearita
- ▶ Pokud se dostatečně pohnu i jen v jedné proměnné, můžu získat obrácený výsledek.
- ▶ Hodnota  $\overline{xw}$  roste se vzdáleností od rozhodovací hranice a nemá žádnou interpretaci.
- ▶ Lineární klasifikátory špatně snášejí chybějící hodnoty.
- ▶ Pokud provádím rozšíření báze, může trvat učení velmi dlouho.

- ▶ LDA je statistický přístup k problému lineární separace.
- ▶ Pamatujete si PCA? (Principal Component Analysis – mluvili jsme o něm u SOM)
- ▶ LDA redukuje původní prostor na přímku (1 dimenzi) a to tak, co nejvíce oddělil vzory různých tříd.
- ▶ V jistém smyslu je to opak rozšíření báze.



# Linear Discriminant Analysis (2)



- Obráz vzoru v nové souřadnici pak získám jako  $\bar{x}\bar{w}$ , kde  $\bar{w}$  jsou váhy.
- V principu počítám to samé jako u lineárního klasifikátoru, ale interpretace a postup jsou úplně jiné.

- ▶ Při počítání LDA se snažím maximalizovat "separaci" mezi třídami.
- ▶ Nejprve pro každou třídu musím spočítat rozptyl v obraze (scatter).

$$y_i = \overline{x_i W}$$

$$\mu_{s_1} = \frac{1}{\|s_1\|} \sum_{\forall x_i \in s_1} x_i \quad \tilde{\mu}_{s_1} = \frac{1}{\|s_1\|} \sum_{\forall x_i \in s_1} y_i = \frac{1}{\|s_1\|} \sum_{\forall x_i \in s_1} x_i W$$

$$\tilde{s}_1^2 = \sum_{\forall x_1 \in s_1} y_i - \tilde{\mu}_{s_1}$$

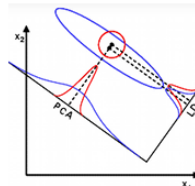
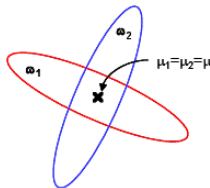
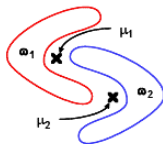
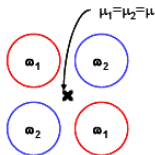
- A nyní můžu spočítat míru "separace" pro projekci danou vahami  $\bar{w}$ .

$$J(\bar{w}) = \frac{\|\tilde{\mu}_{s_1} - \tilde{\mu}_{s_2}\|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

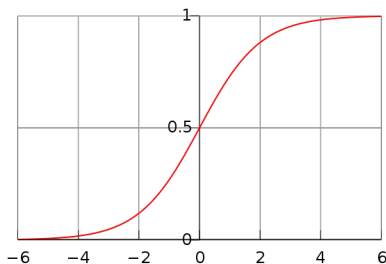
- Hledáme tedy projekci, která co nejvíc seskupí vzory stejné třídy a co nejvíc zároveň oddělí vzory různých tříd.

- ▶ LDA lze zobecnit tak, aby dokázalo klasifikovat do více tříd.
- ▶ Pak se nevytváří pouze jedna nová dimenze, ale "počet tříd"-1 nových dimenzí a mírně se upraví vzorečky :).

- ▶ Stejně jako všechno, má i LDA nějaké mouchy:
  - ▶ Počet dimenzí vytvořený LDA nemusí stačit ke správném zařazení do třídy.
  - ▶ LDA předpokládá normální rozložení dat ve třídách. A selhává, pokud tomu tak není. (Například třídy mají komplexní tvary).
  - ▶ Rozíly mezi třídami jsou spíše v rozptylu hodnot než v rozdílu středních hodnot.



- ▶ Co kdybych chtěl, abych dokázal vzdálenosti od rozhodovací hranice nějak interpretovat.
- ▶ Ideálně ještě tak, že je to pravděpodobnost příslušnosti k dané třídě.
- ▶ Existuje logistická funkce, která je definovaná takto
$$f(z) = \frac{1}{1+e^{-z}}$$
- ▶ Její hodnoty jsou omezené na interval  $0 \dots 1$ . A navíc kolem  $z = 0$  (rozhodovací hranice) je její změna největší.



- ▶ Zkusíme zavést pojem  $sance = \frac{p}{1-p}$
- ▶ Abychom ji dostali do rozmezí  $0 \dots 1$  zkusme ji zlogaritmovat.
- ▶ Tím získáme funkci  $logit(p) = \log(sance(p)) = \log(\frac{p}{1-p})$
- ▶ Zkusme předpokládat, že
$$logit(p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k$$
- ▶ Kde  $\beta_1, \dots, \beta_k$  jsou regresní koeficienty a  $\beta_0$  je posun.
- ▶ A teď zkusme opět dopočítat pravděpodobnost vzoru  $\bar{x}$ .
- ▶ 
$$p(x) = \frac{1}{1 + e^{-\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k}}$$

- ▶ LDA převzata z  
`research.cs.tamu.edu/prism/lectures/pr/pr_l10.pdf`
- ▶ `http://www.omidrouhani.com/research/  
logisticregression/html/logisticregression.htm`