# Rozpoznávanie obrazcov - 6. cvičenie
# Lineárny klasifikátor a SVM

Viktor Kocur
viktor.kocur@fmph.uniba.sk

DAI FMFI UK

30.3.2020

# Lineárny klasifikátor

## Základný princíp

Jadrom lineárneho klasifikátora je lineárna funkcia $f : \mathbb{R}^n \mapsto \mathbb{R}, f(\vec{x}) = \vec{w}^T \vec{x} + b$, kde $\vec{x}$ je príznakový vektor, $\vec{w}$ je vektor váh a $b$ je tzv. bias člen.

## Klasifikácia

Ak máme dve triedy $\omega_1$ a $\omega_2$, tak príznakový vektor $\vec{x}$ priradíme do triedy $\omega_1$ ak $f(\vec{x}) \geq 0$, do triedy $\omega_2$ ak $f(\vec{x}) < 0$.

# Lineárny klasifikátor

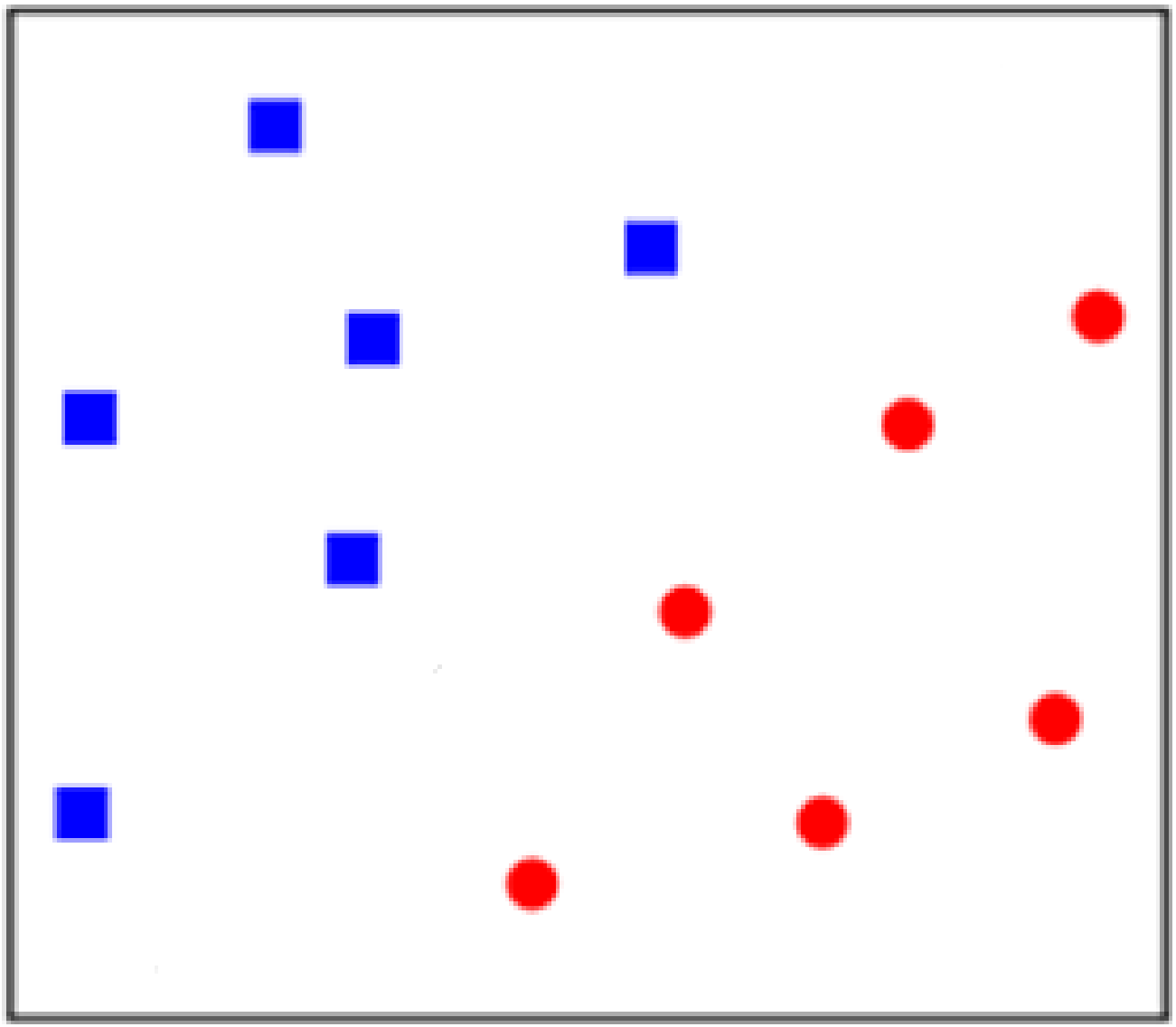
## Geometrický význam

Funkcia $f$ v skutošti delí príznakový priestor na dve časti oddelené nadrovinou. Body pre ktoré $f(\vec{x}) = 0$ ležia práve na tejto nadrovine.

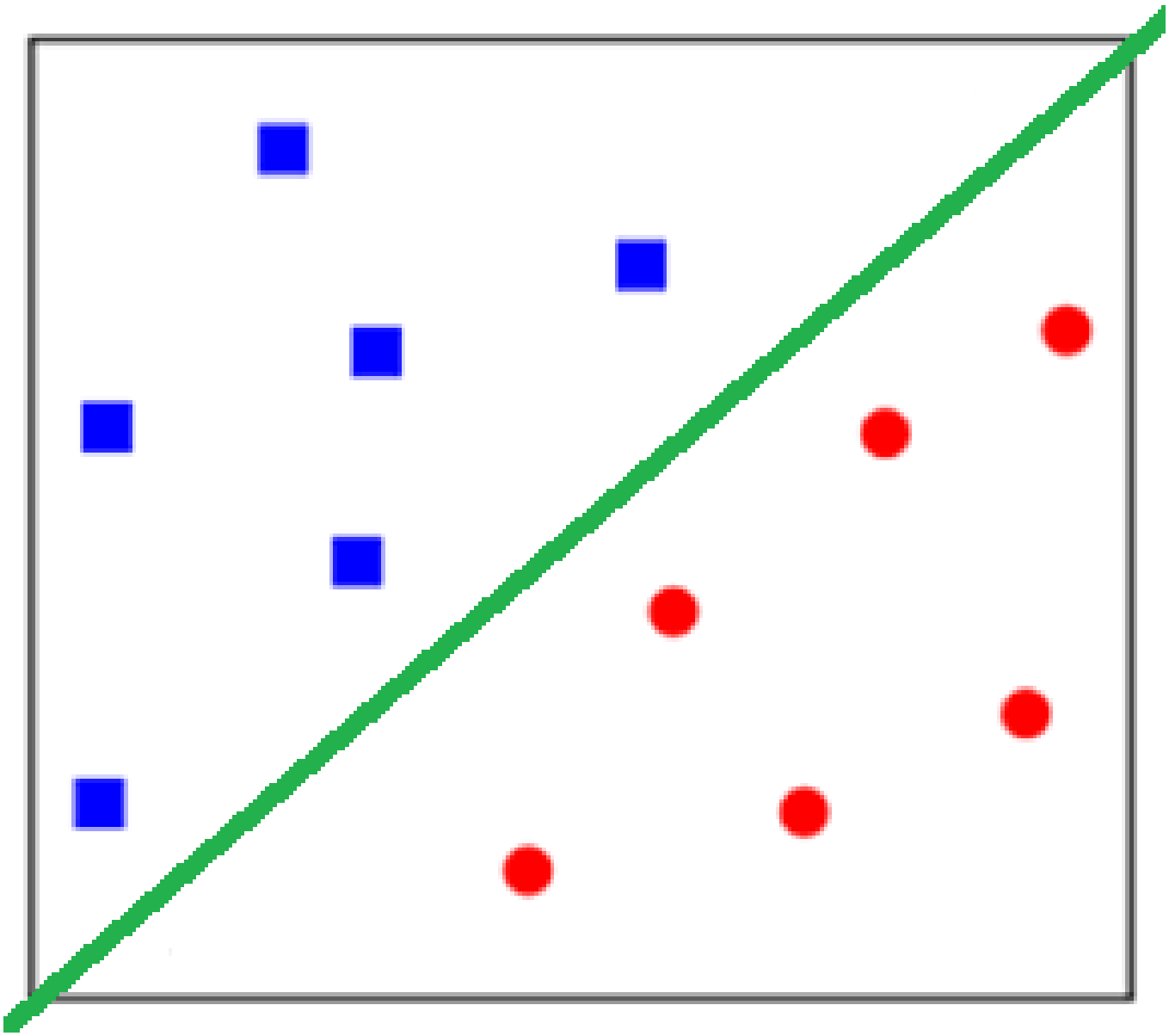
## Cieľ trénovania

Pri tréningu teda chceme nájsť také parametre, aby nadrovina, ktorú využívame rozdelovala trénovaciu množinu čo najlepšie.

# Lineárny klasifikátor



# Lineárny klasifikátor



# Trénovanie

## Trénovacie dáta

Na natrénovanie klasifikátora budeme potrebovať tzv. trénovacie dáta. Teda ku vektoru $\vec{x}$ potrebujeme označenie triedy $y \in \{0, 1\}$. Náš cieľ je aby náš klasifikátor fungoval dobre na trénovacích dátach.

## Regularizácia

Niekedy chceme klasifikátor, ktorý nieje najlpší na trénovacích dátach ale vie dobre generaliozovať. To sa nazýva regularizácia.

## Cenová funkcia

Dobrý klasifikátor dostaneme tak, že vyvtvoríme tzv. cenovú funkciu $C : \mathbb{R}^{n+1} \mapsto \mathbb{R}, C(b, \vec{w})$, ktorá má globálne minimum pre také parametre ktoré rozdelujú triedy čo najlepšie. Trénovanie je potom vlastne optimalizačná úloha.

# Cenová funkcia - I

## Zjednodušenie

Keďže člen $b$ je nám to to trocha komplikuje, tak zavedieme nové značenie: $\vec{\theta} = (b, \vec{w})$ a ako vektor $\vec{X} = (1, \vec{x})$. Toto nám umožní zapísať $f(\vec{X}) = \vec{\theta}^T \vec{X}$.

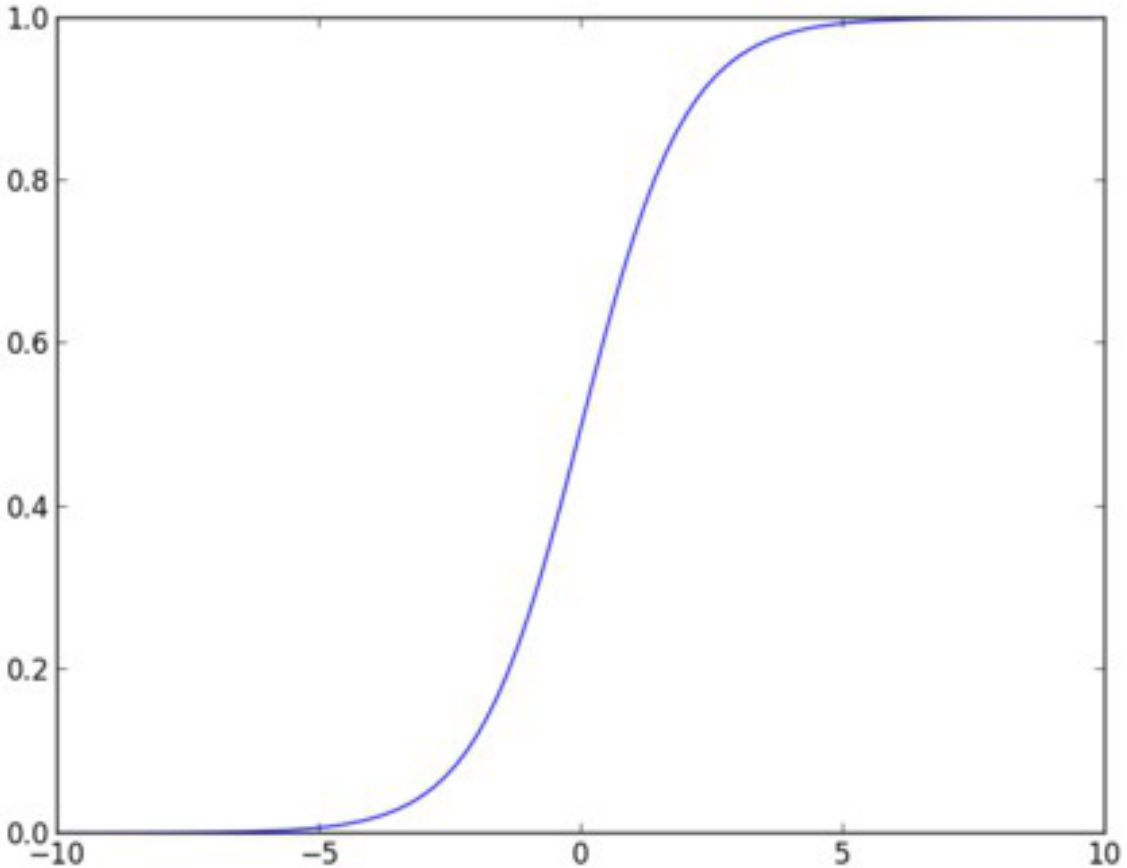
## Sigmoid

Pri definícii použijeme sigmoidálnu funkciu: $\sigma(z) = \frac{1}{1+e^{-z}}$.

## Sigmoid - derívacia

$\sigma(z)' = \sigma(z)(1 - \sigma(z))$.

# Sigmoid



# Cenová funkcia - II

## Zjednodušenie

Zavedieme ešte jednu funkciu: $h_\theta = \sigma(f(\vec{x}))$.

## Cenová funkcia - binary crossentropy

$$J(\vec{\theta}) = \frac{1}{m} \sum_{i=1}^{m} \left( -y^{(i)} log(h_\theta(\vec{x}^{(i)})) - (1 - y^{(i)}) log(1 - h_\theta(\vec{x}^{(i)})) \right)$$

## Cenová funkcia - derívacia

$$\frac{\partial J}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} \left( h_\theta(\vec{x}^{(i)}) - y^{(i)} \right) x_j^{(i)}$$

# Optimalizácia

## Gradientný zostup

$\theta_i := \theta_i - \eta \frac{\partial J}{\partial \theta_i}$

## Skutočná optimalizácia

V skutočnosti sa používajú sofistikovanejšie algoritmy. Ako napríklad SGD, alebo metódy založené na Hessovej matici.

## Optimalizácia v matlabe

x = fminunc(fun,x0) - nájde optimálne parametre x pre ktoré je funkcia fun minimálna (ak sa to podarí). Keďže sa používajú iteratívne metódy, tak je nutné zadať inicializačné hodnoty x0.

# Optimalizácia

## Úloha

Prezrite si skript LinearClassifier.m

## Úloha

Dokončite funkciu costFunction. Pomocou cenovej funkcie ktorá je na pár slidoch dozadu.

# Lineárny klasifikátor - Matlab

## Regularizácia

K cenovej funkcii pridáme aj regularizačný člen: $C_R(\vec{\theta}) = C(\vec{\theta}) + R(\vec{\theta})$. Napríklad $R(\vec{\theta}) = \sum_{i=2}^{n} \theta_i^2$, alebo $\sum_{i=2}^{n} |\theta_i|$

## fitclinear

Mdl = fitclinear(x,y) - vráti lineárny klasifikačný model Mdl pre príznakové vektory ktoré su riadkami matice x a k nim korešpondujúcimi triedami y. Táto funkcia dokáže aj SVM aj logistickú regresiu. Pre možnosti sa pozrite do helpu. Defualtne sa tu využíva aj regularizácia.

## Mdl.predict

Mdl.predict(x) - vráti triedu pre daný príznakový vektor.

# Lineárny klasifikátor - Matlab

## Mdl.Beta, Mdl.Bias

Mdl.Beta - vráti to čo sme si označovali ako vektor $\vec{w}$. Mdl.Bias - vráti to čo sme si označili ako $b$.

## Úloha

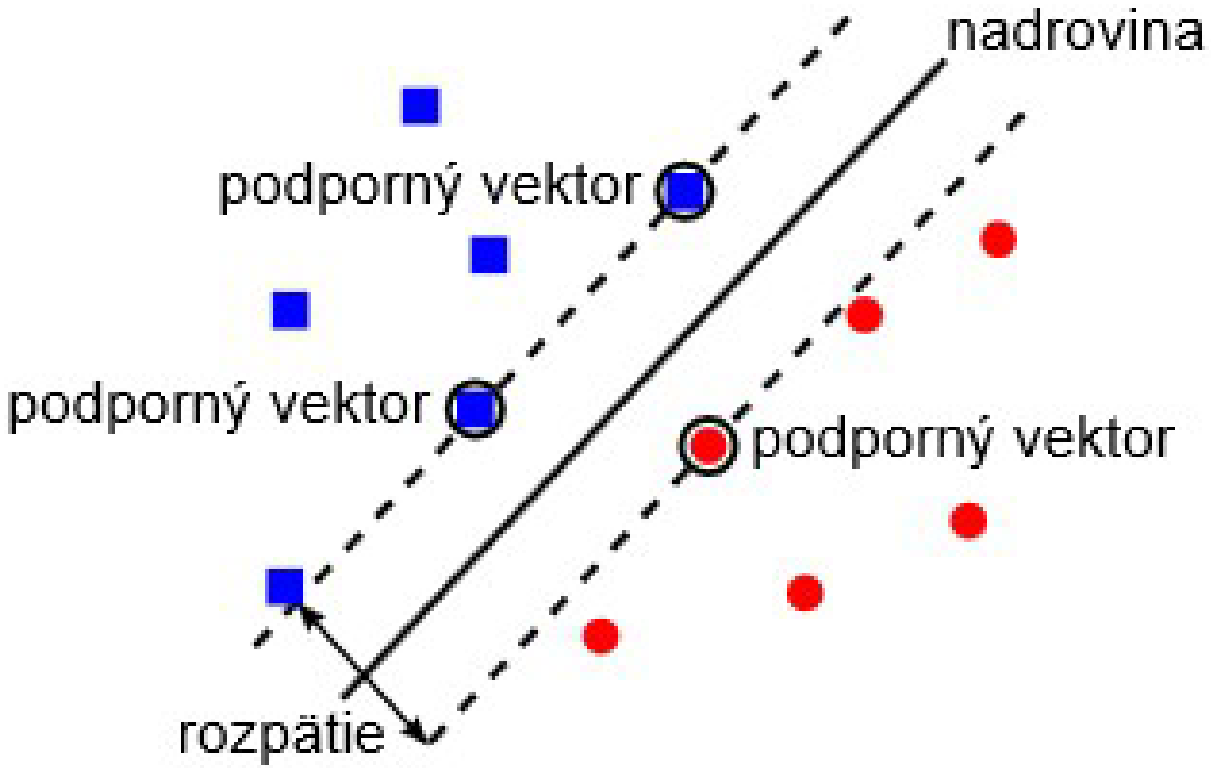
Do obrázku (gscatter) s dátami z ex2data1.txt dokreslite delaicu priamku pre klasifikátor z fitclinear. Môžete použiť plot, alebo refline. Pozn.: pre refline použijeme:

$$\beta_1 \cdot x_1 + \beta_2 \cdot x_2 + bias = 0 \iff x_2 = m \cdot x_1 + b$$

$$x_2 = -\frac{\beta_1}{\beta_2} \cdot x_1 - \frac{bias}{\beta_2}$$

$$m = -\frac{\beta_1}{\beta_2}, b = -\frac{bias}{\beta_2}$$

# Princíp


nadrovina
podporný vektor
podporný vektor
podporný vektor
rozpätie

# SVM

## Princíp

SVM nájde podporné vektory a snaží sa nájsť deliacu čiaru tak, aby bolo rozdelovací pás čo najširší pomocou podmienky pre podporné vektory vo forme $\vec{w}^T\vec{x} + b = \pm 1$.

## Kernelový trik

Dáta často niesu lineárne separabilné. Preto je nutné príznakový priestor transformovať pomocou tzv. kernelu. Teda funkcie $\phi : \mathbb{R}^n \mapsto \mathbb{R}^m$, pre ktorú platí, že existuje funkcia $k$, tž: $k(x_i, x_j) = \phi(x_i)\phi(x_j)$. SVM potom hľadá lineárny klasifikátor v novom priestore $\mathbb{R}^m$.

# SVM

## fitcsvm

SVMMdl = fitcsvm(X,y) - vráti SVM model natrénovaný na príznakoch X a triedach y.

## fitcsvm

SVMMdl = fitcsvm(X,y, 'KernelFunction',nazov, 'KernelScale', 'auto') - vráti SVM s kernelovým trikom. Pozor nezabudnite na KernelScale.

# SVM - Úloha

## showSVM

showSVM(SVMMdl, X, y) - zobrazí SVM model pre 2D dáta X, y (je to .m file v zipe)

## Úloha

Zobrazte si SVM s rôznymi kernelmi. Skúste čo sa stane ak nenastavíte KernelScale.