Seminár Robotika.SK

Ako naučiť robota rozpoznávať kto je kto pomocou knižnice Dlib

Andrej Lúčny

Katedra aplikovanej informatiky FMFI UK

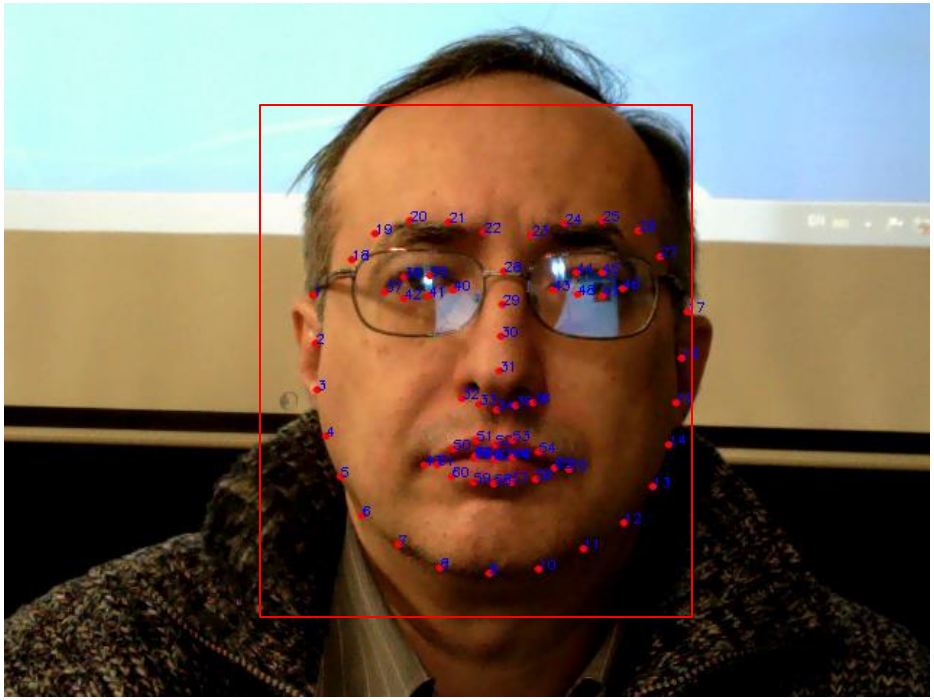
lucny@fmph.uniba.sk

http://dai.fmph.uniba.sk/w/Andrej_Lucny

`www.robotika.sk/seminar/2018/cviko7-faces.zip`
`www.robotika.sk/seminar/2018/cviko8-faces.zip`

Tvár i jej črty máme, čo ďalej?

- Od minulého cvičenia vieme, že Dlib obsahuje HOG detektor tváre (obdĺžnik) a detektor čŕt tváre (bodky)
- Ako to teraz môžeme využiť na určenie osoby, ktorej tvár na obraze vidíme?



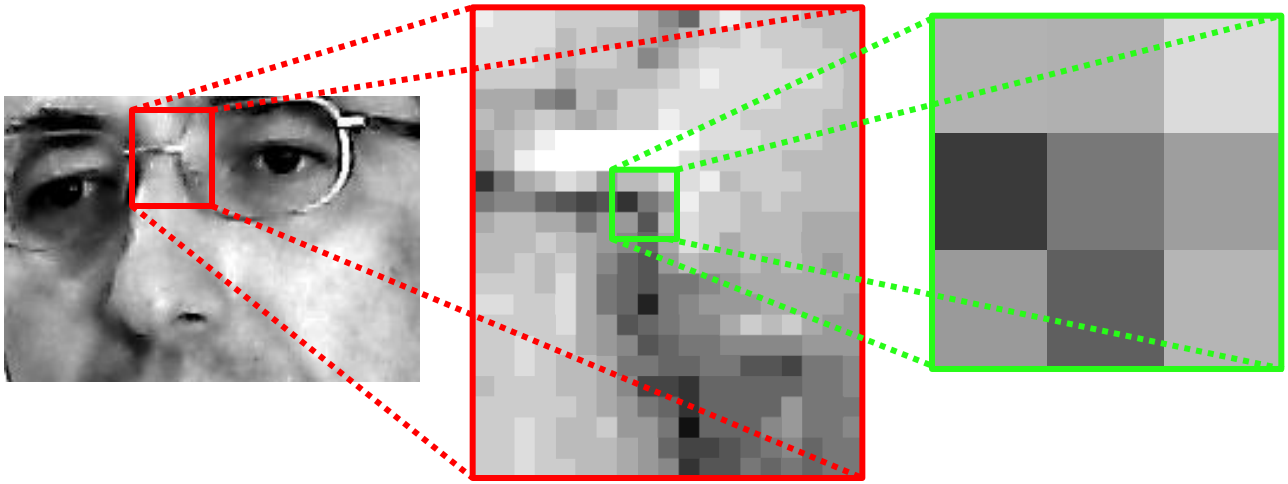
- V prvom rade potrebujeme transformovať tvár do určitého normálneho tvaru

Hľadáme: invariantné vyjadrenie obrazu

- Aká reprezentácia je rovnaká pre tieto obrázky?

Local Binary Patterns

- Invariant: ak je jeden pixel svetlejší od druhého na jednom obraze, bude tento vzťah zachovaný aj obraze so zmenenou jasnosťou, kontrastom či ostrosťou.



0	0	0
1		0
0	1	0

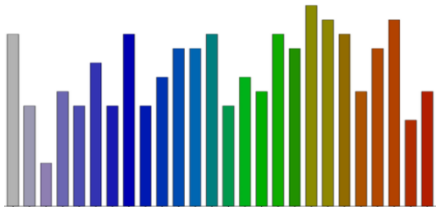
00000101

5

- Každému pixelu vieme priradiť jeho LBP – číslo od 0 do 255

Local Binary Pattern Histogram

- Každý pixel má svoj LBP
  – číslo od `0` do `255`



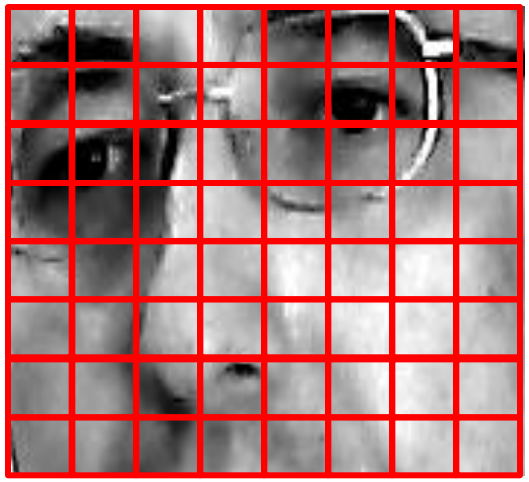
- Spočítame koľko je akých: napr. 587 krát 0, 32 krát 1, 564 krát 2, ani raz 3, ... a dva krát 127, dostaneme histogram

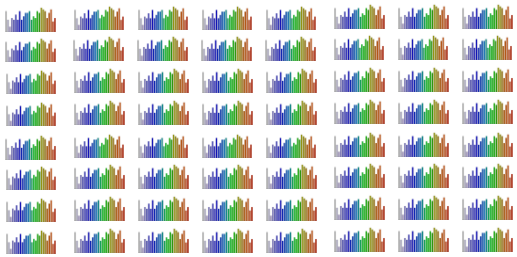
`3F 2E 01 ... FF`

256 bytov

- Tento histogram normalizujeme tak, že jedným bytom vyjadríme relatívnu početnosť

Local Binary Pattern Histogram





3F 2E 01 03 ... FF

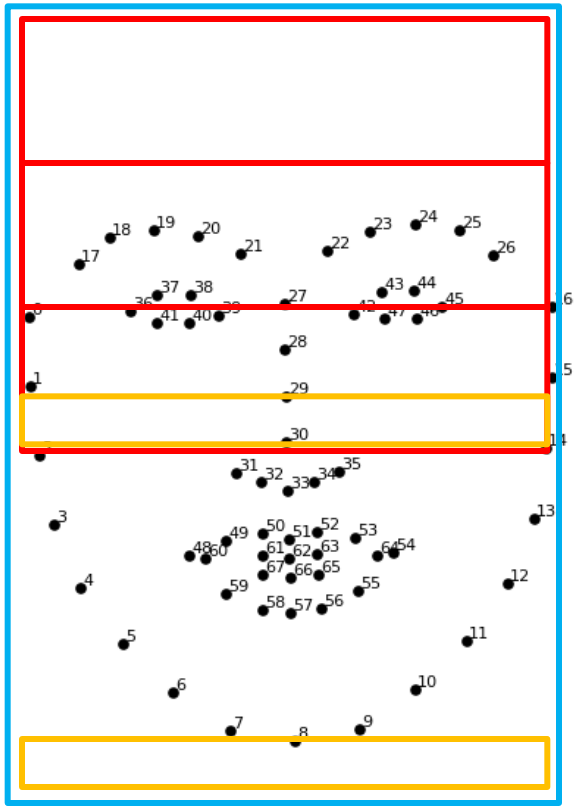
16384 bytov

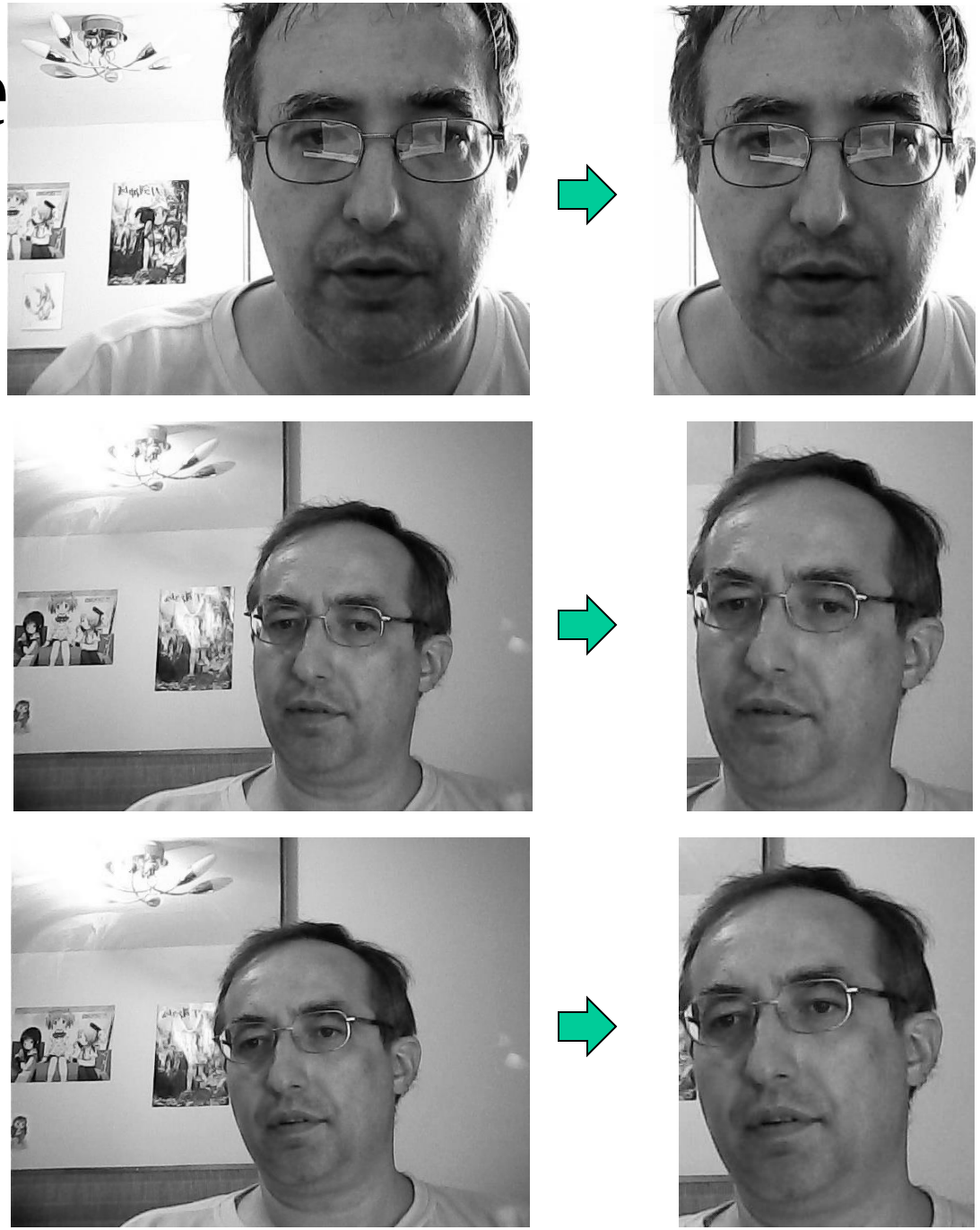
- Aby sme zachytili aj priestorové rozloženie, môžeme obraz štruktúrovať na oblasti napr. obraz 64x64 pokryjeme mriežkou 8x8 a pre každú časť spočítame jej LBP, tieto potom spojíme do vektora za sebou, čiže v tomto prípade dostaneme 64 x 256 = 16384 bytov

Local Binary Pattern Histogram

- LBPH sú len málo odolné voči posunutiu a vôbec nie voči otočeniu či zväčšeniu/zmenšeniu
- Obraz preto musíme vzhľadom na tieto transformácie normalizovať vo vlastnej réžii
- Pomocou Facial Landmark detektoru ľahko určíme otočenie i lepší výrez obrazu než nám dá HOG detektor (ktorému mierne posunutie nevadí)
- Potom stačí vyrovnaný a otočený výrez zmenšiť na štandardnú veľkosť (napr. 64x64), čím riešime zväčšenie a zmenšenie

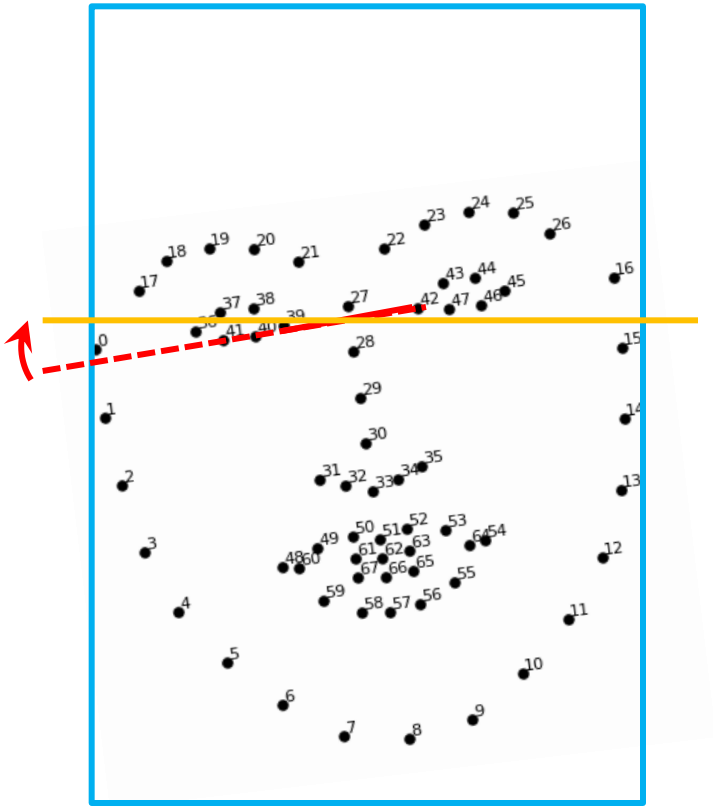
Predspracovanie

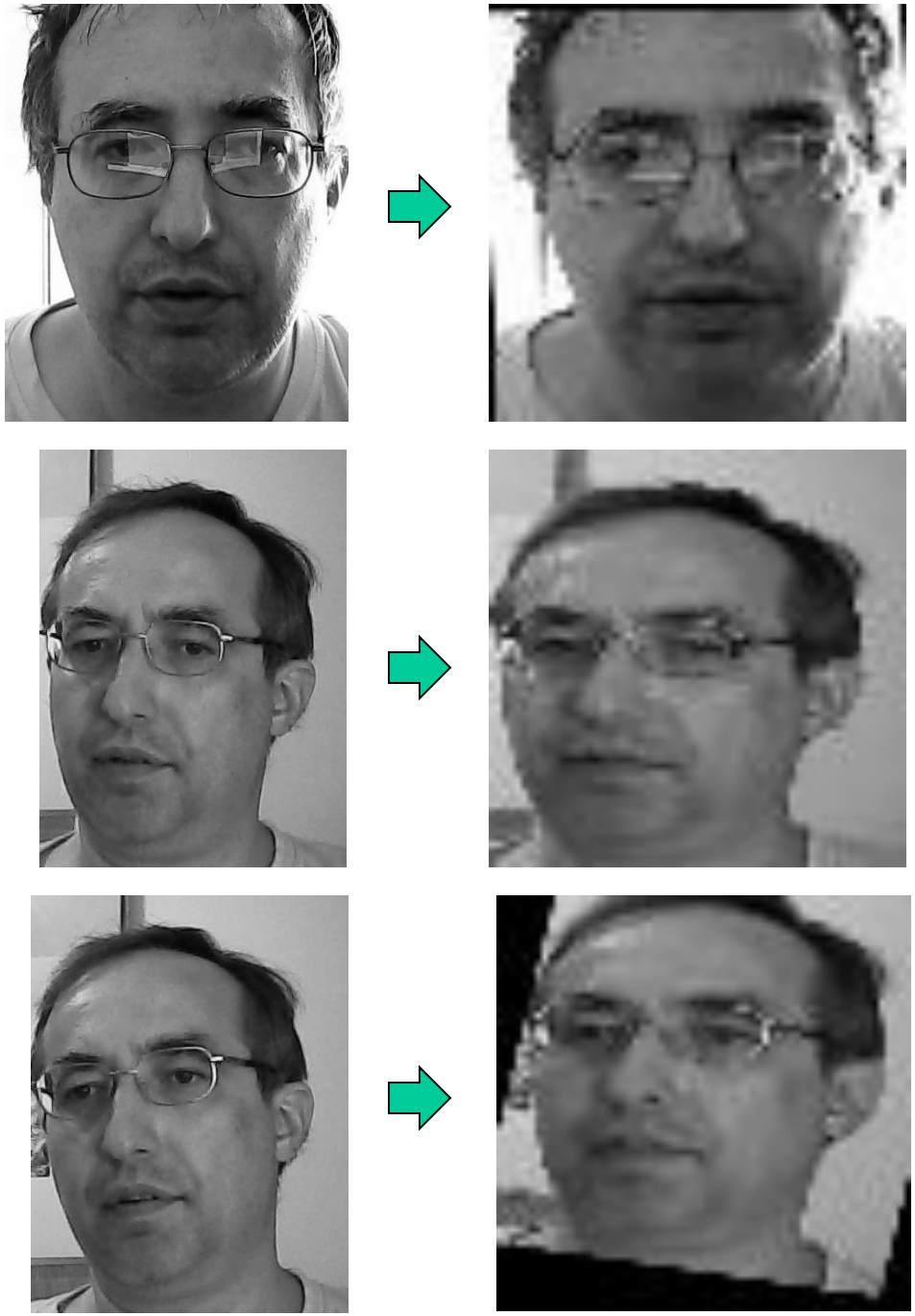




FaceAligner

Predspracovanie

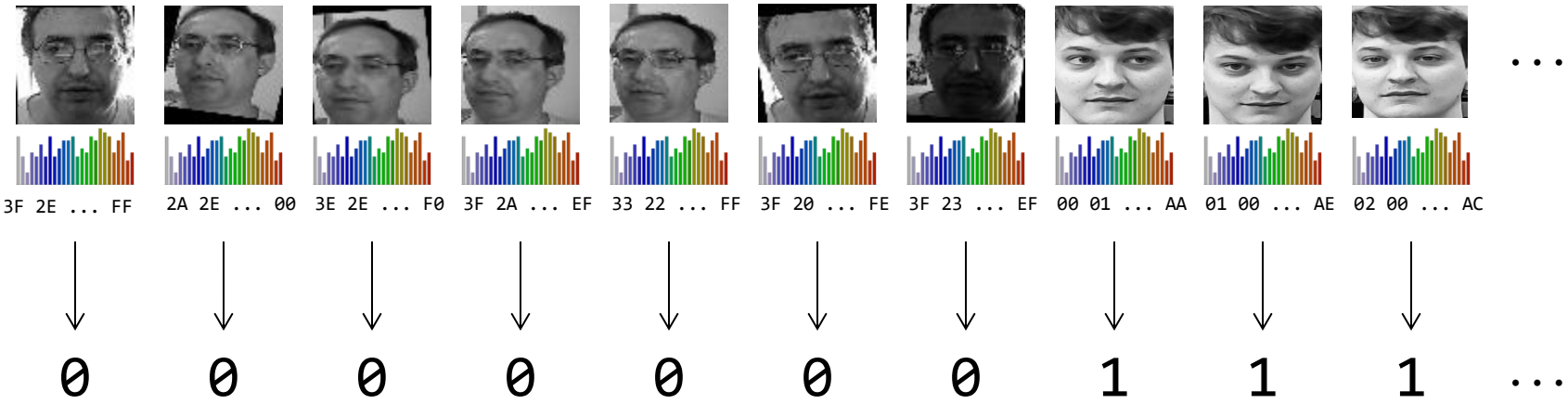




- normalizujeme veľkosť na 64 x 64
- normalizujeme rotáciu
- posunutie nás netrápi

Dataset pre Machine learning

- z určitého pohľadu na konkrétnu tvár teraz vieme dostať približne rovnaký LBPH bez ohľadu na to, či je bližšie alebo ďalej od kamery, či je naklonená vľavo, vpravo alebo je rovno
- Iný LBPH dostaneme, keď zmeníme uhol pohľadu na tvár. Jednu tvár nám teda bude reprezentovať viacero snímok tváre z rôznej strany

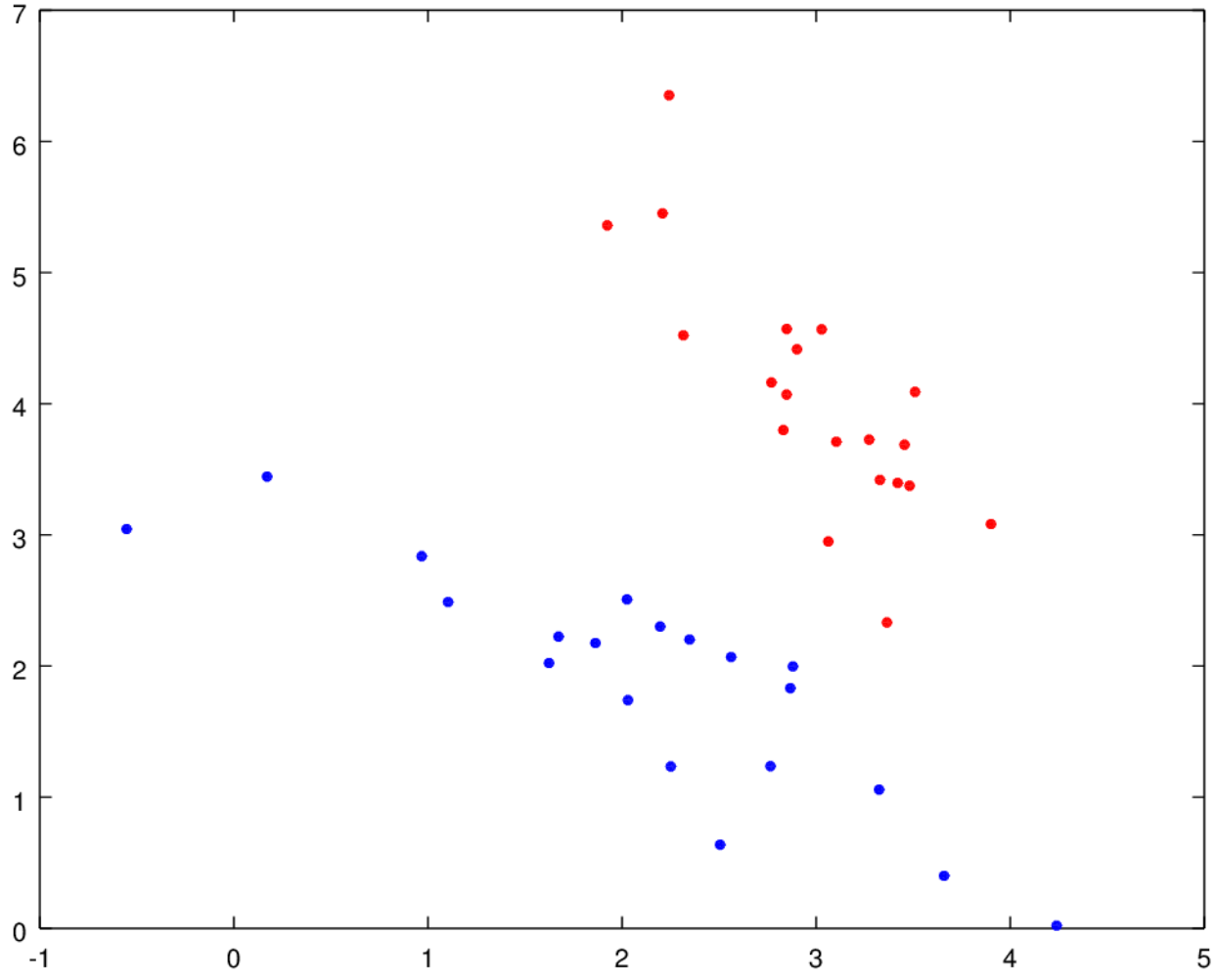


Linear Discriminant Analysis (LDA)

- Na naučenie sa funkcie priraďujúcej k LBPH správne ID sa dá použiť akákoľvek metóda strojového učenia
- Dobré výsledky dáva LDA
- V našom prípade LDA klasifikuje dáta v 256 rozmernom priestore do viacerých kategórii, avšak spôsob jej fungovania si priblížime na dvojrozmernom priestore a dvoch kategóriách
- Podstatou LDA je nájsť takú lineárnu transformáciu priestoru, ktorá najlepšie rozlišuje (diskriminuje) dané vzorky pri redukcii dimenzie priestoru.

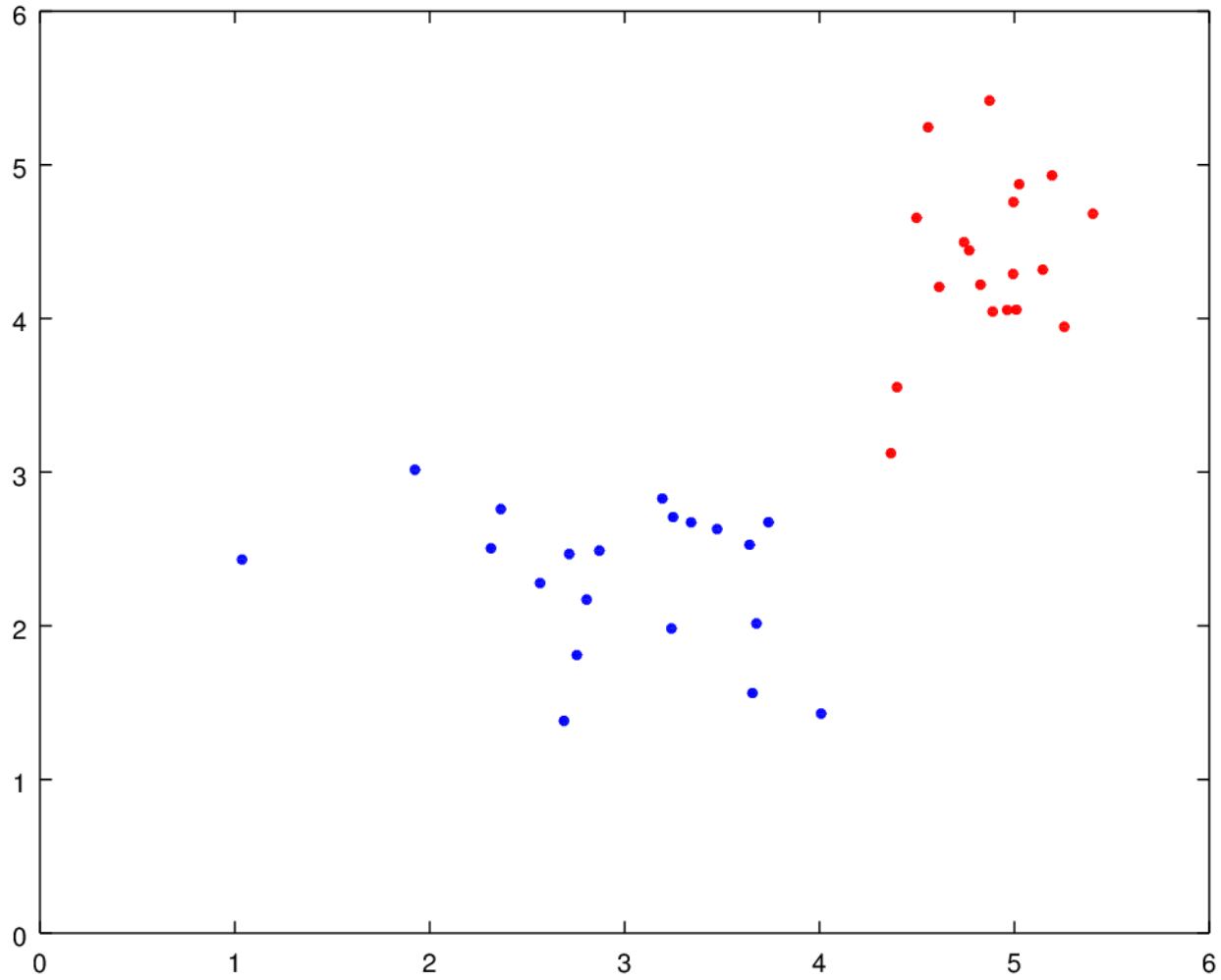
Linear Discriminant Analysis (LDA)

Uvažujme tieto vzorky dvoch kategórii



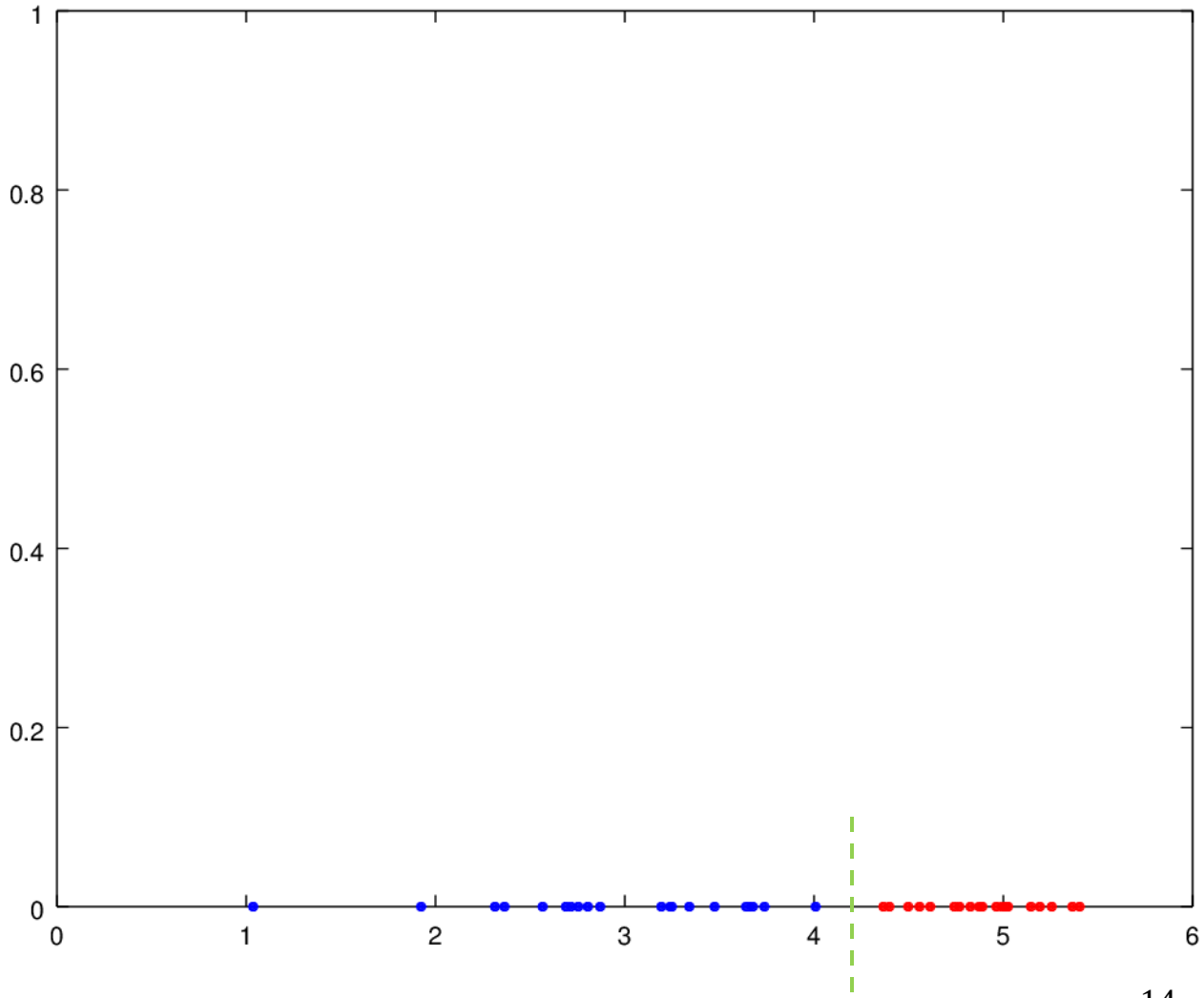
Linear Discriminant Analysis (LDA)

LDA ich dokáže transformovať ich súradnice tak ...



Linear Discriminant Analysis (LDA)

... že po redukcií dimenzie možno kategórie ľahko rozlíšiť



Linear Discriminant Analysis (LDA)

Pri transformácii priestoru dát LDA uvažuje len lineárne transformácie, pri dvojrozmerných dátach ide teda o stĺpcový vektor (označme *W*)

$$\left(x_1, x_2\right) \times \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \left(x^{'}\right)$$

čo znamená, že

$$x^{'} = w_1 x_1 + w_2 x_2$$

- Každú zmenená súradnica je určitou lineárnou kombináciou pôvodných súradníc

Linear Discriminant Analysis (LDA)

Ktorá z možných *W* je najlepšia pre dané vzorky?
Tá, ktorá kategórie (*a* a *b*) najlepšie diskriminuje.

LDA hľadá takú *W*, pre ktorú nadobúda maximum:

$$J(W) = \frac{(\mu_a^W - \mu_b^W)^2}{(\sigma_a^W)^2 + (\sigma_b^W)^2}$$

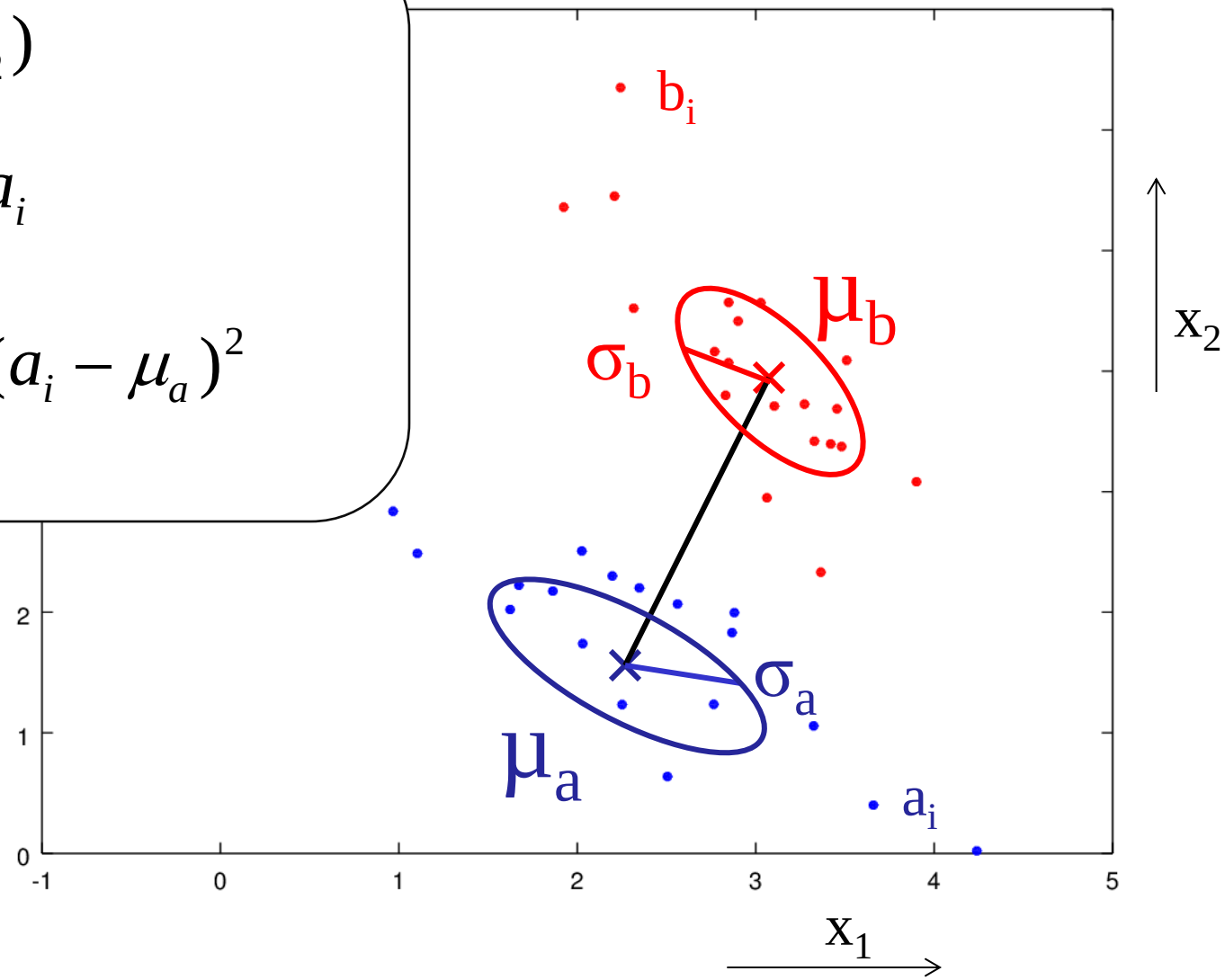
čo je približne vtedy, keď je vzdialenosť priemerov kategórii μ čo najväčšia a rozptyl vzoriek σ^2 čo najmenší

Linear Discriminant Analysis (LDA)

$$a_i = (x_1, x_2)$$

$$\mu_a = \frac{1}{n}\sum_i a_i$$

$$\sigma_a^2 = \frac{1}{n}\sum_i (a_i - \mu_a)^2$$

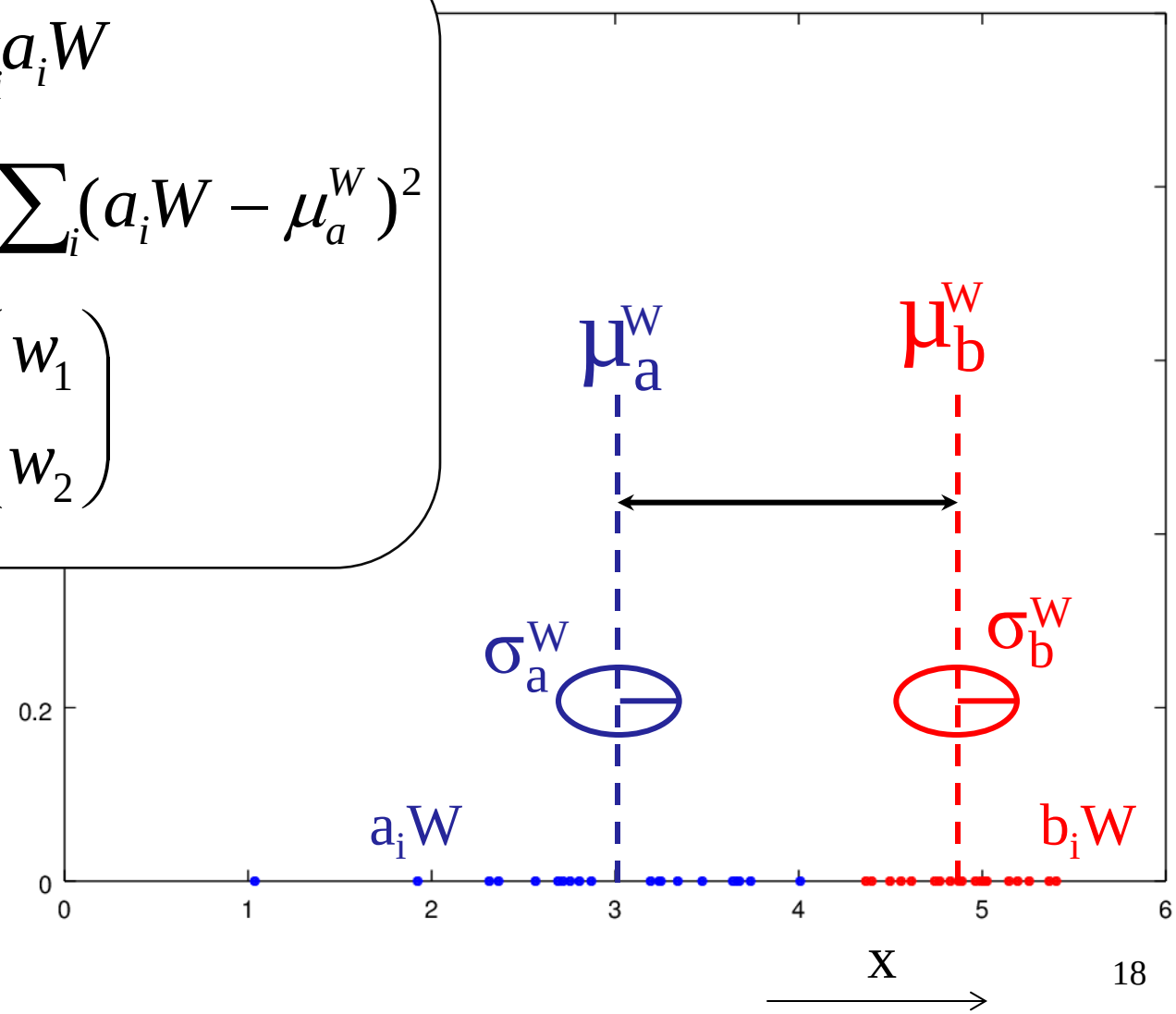


Linear Discriminant Analysis (LDA)

$$\mu_a^W = \frac{1}{n}\sum_i a_i W$$

$$(\sigma_a^W)^2 = \frac{1}{n}\sum_i (a_i W - \mu_a^W)^2$$

$$W = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$



Linear Discriminant Analysis (LDA)

Ako nájsť pre ktorú *W* nadobúda *J(W)* maximum? J(W) je diferencovateľná, možno si ju predstaviť ako kopec na vrchole ktorého je vodorovný v podľa každej zo svojich priestorových súradníc w_1, w_2

Riešime teda sústavu dvoch rovníc o dvoch neznámych:

$$\frac{dJ(W)}{dw_i} = 0$$

To sa v princípe vždy nejako dá, ale v našom špeciálnom prípade sa dá k riešeniu dostať aj (výpočtovo) ľahšie.

Linear Discriminant Analysis (LDA)

Dá sa totiž ukázať, že riešením tejto sústavy rovníc je matica zložená z tzv. vlastných vektorov určitej matice. A vlastné vektory vieme veľmi efektívne spočítať. Ako na to prídeme?

Skúmajme najprv ako sa μ_a k μ_a^W a σ_a ku σ_a^W, t.j. čo robí s priemerom a rozptylom vzoriek hľadaná transformácia W:

$$\mu_a^W = \frac{1}{n}\sum_i a_i W = \mu_a W$$

$$\mu_b^W = \mu_b W$$

$$(\mu_a^W - \mu_b^W)^2 = W^T \overbrace{(\mu_a - \mu_b)^T(\mu_a - \mu_b)}^{S} W$$

určuje rozptyl medzi kategóriami

Linear Discriminant Analysis (LDA)

$$(\sigma_a^W)^2 = \frac{1}{n_a}\sum_i (a_i W - \mu_a^W)^2 = \frac{1}{n_a}\sum_i (a_i W - \mu_a W)^2$$

$$= \frac{1}{n_a}\sum_i (a_i W - \mu_a W)^2 = \frac{1}{n_a}\sum_i W^T (a_i - \mu_a)^T (a_i - \mu_a) W$$

$$= \frac{1}{n_a} W^T \underbrace{\left(\sum_i (a_i - \mu_a)^T (a_i - \mu_a)\right)}_{S_a} W = \frac{1}{n_a} W^T S_a W$$

$S_a \longleftarrow$ kovariančná matica, určuje rozptyl vzoriek v rámci kategórie

$$(\sigma_b^W)^2 = \frac{1}{n_b} W^T S_b W$$

Linear Discriminant Analysis (LDA)

$$J(W) = \frac{(\mu_a^W - \mu_b^W)^2}{(\sigma_a^W)^2 + (\sigma_b^W)^2} = \frac{W^T S W}{\frac{1}{n_a} W^T S_a W + \frac{1}{n_b} W^T S_b W}$$

$$= \frac{W^T S W}{W^T \underbrace{(S_a / n_a + S_b / n_b)}_{S_{ab}} W} = \frac{W^T S W}{W^T S_{ab} W}$$

Linear Discriminant Analysis (LDA)

Riešime: $$\frac{dJ(W)}{dW} = 0 \qquad \frac{d}{dW}\left(\frac{W^T S W}{W^T S_{ab} W}\right) = 0$$

$$W^T S_{ab} W \frac{d}{dW}\left(W^T S\, W\right) - W^T S\, W \frac{d}{dW}\left(W^T S_{ab} W\right) = 0$$

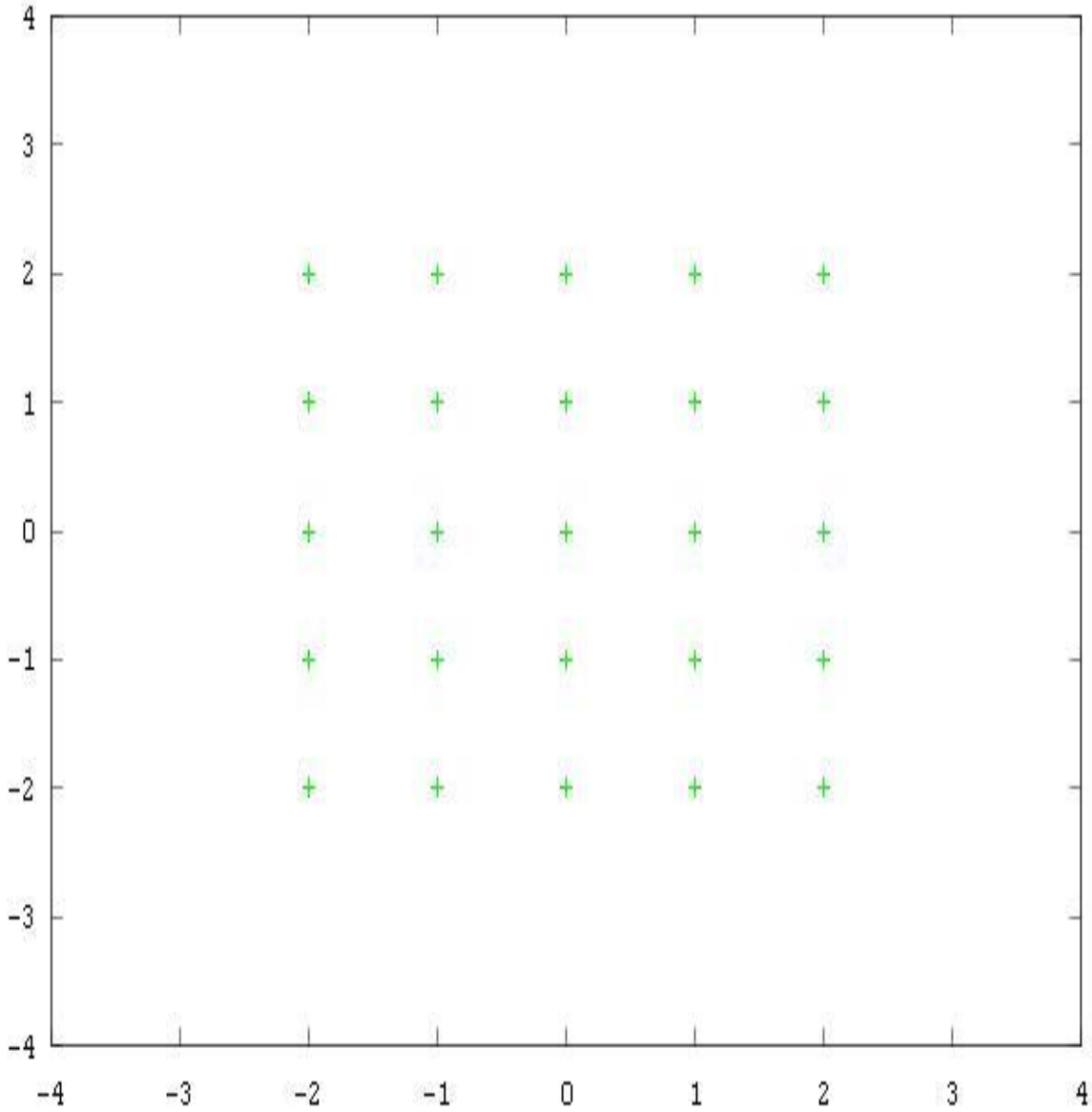
$$W^T S_{ab} W\left(2 S\, W\right) - W^T S\, W\left(2 S_{ab} W\right) = 0$$

$$S\, W - \frac{W^T S\, W}{W^T S_{ab} W} S_{ab} W = 0$$

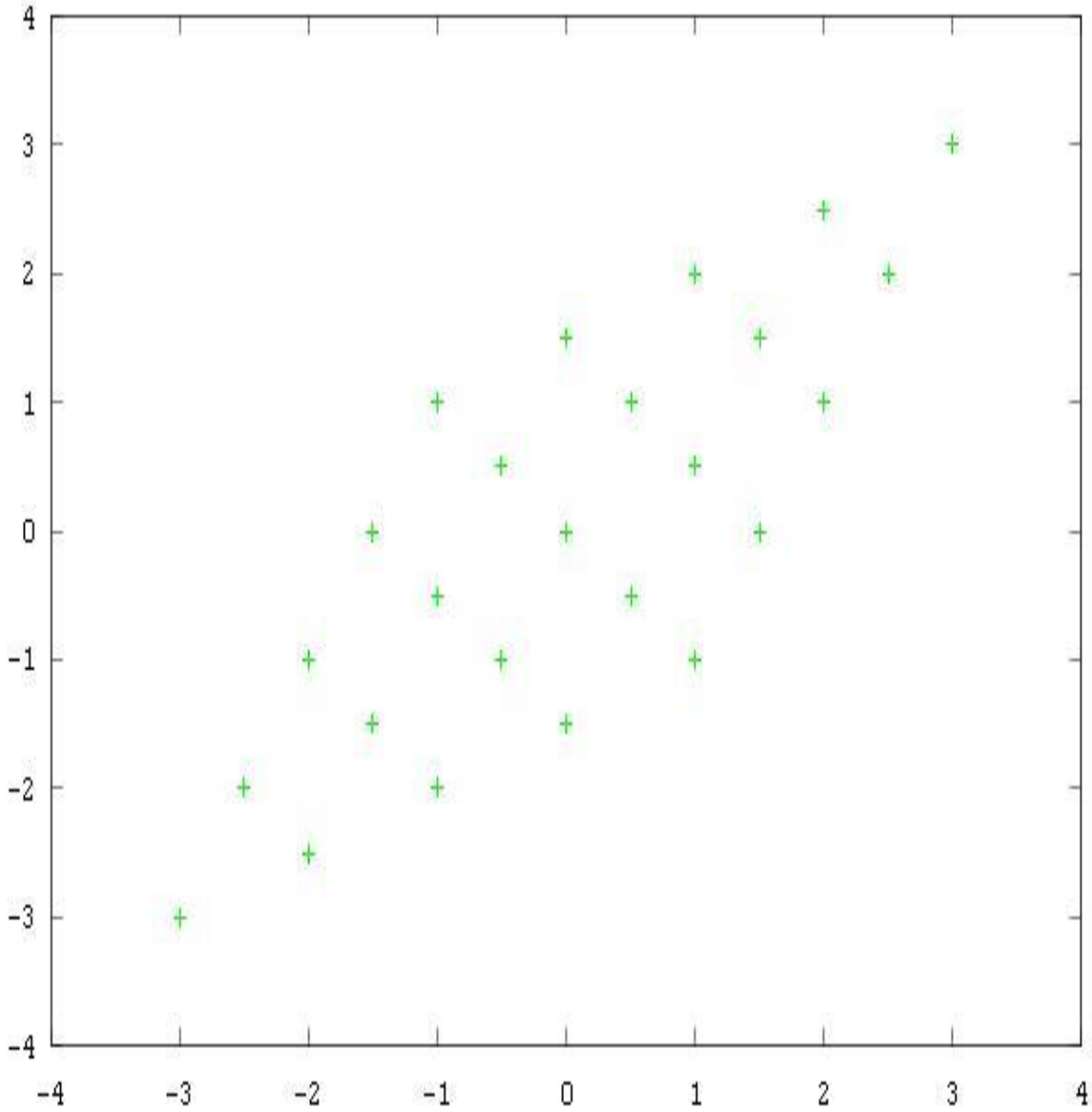
vlastná hodnota M

$$S\, W = J(W) S_{ab} W \qquad \underbrace{S_{ab}^{-1} S}_{M}\, W = J(W) W$$

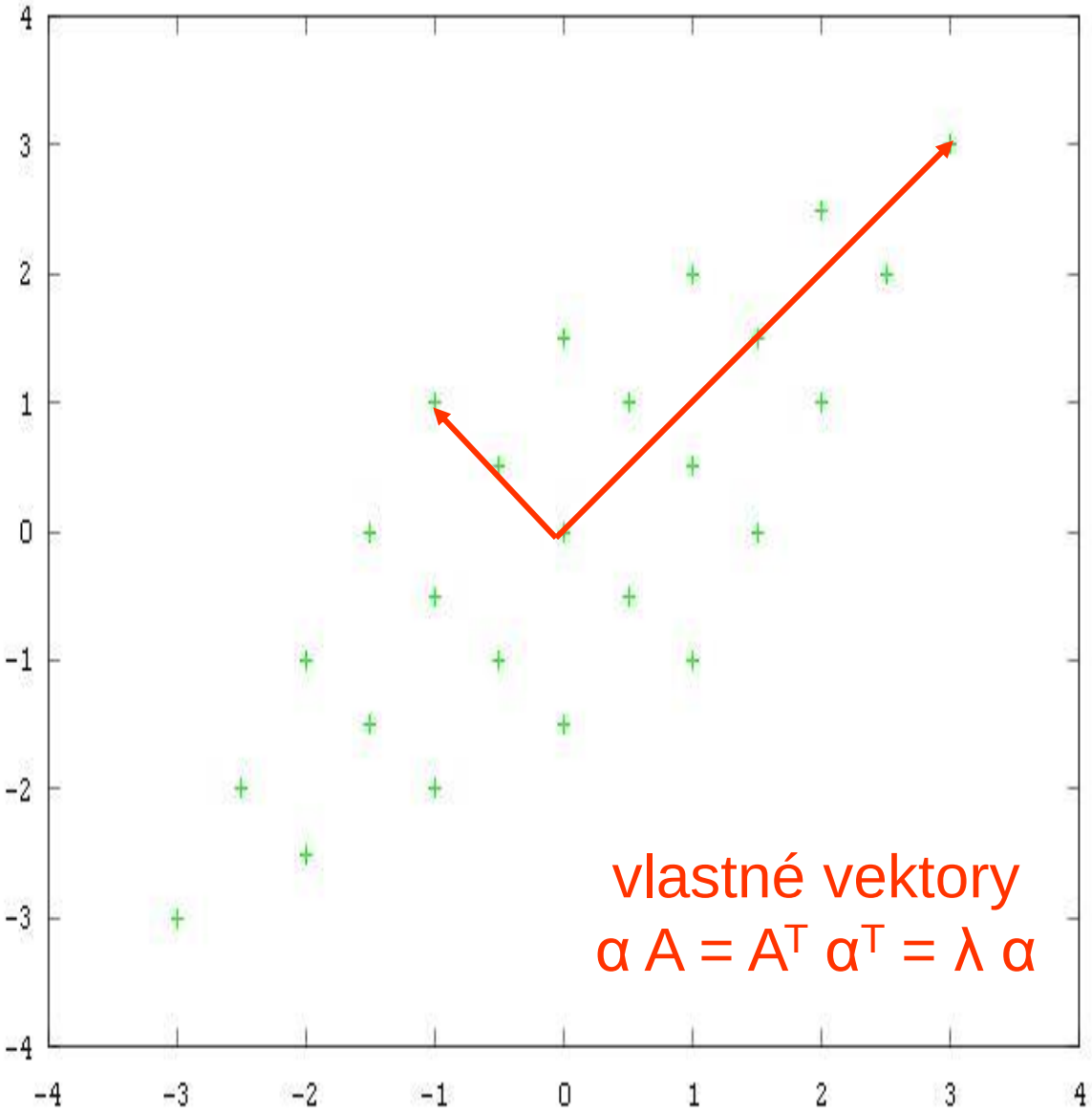
vlastný vektor M



Keď transformujeme tieto body pomocou nejakej matice A

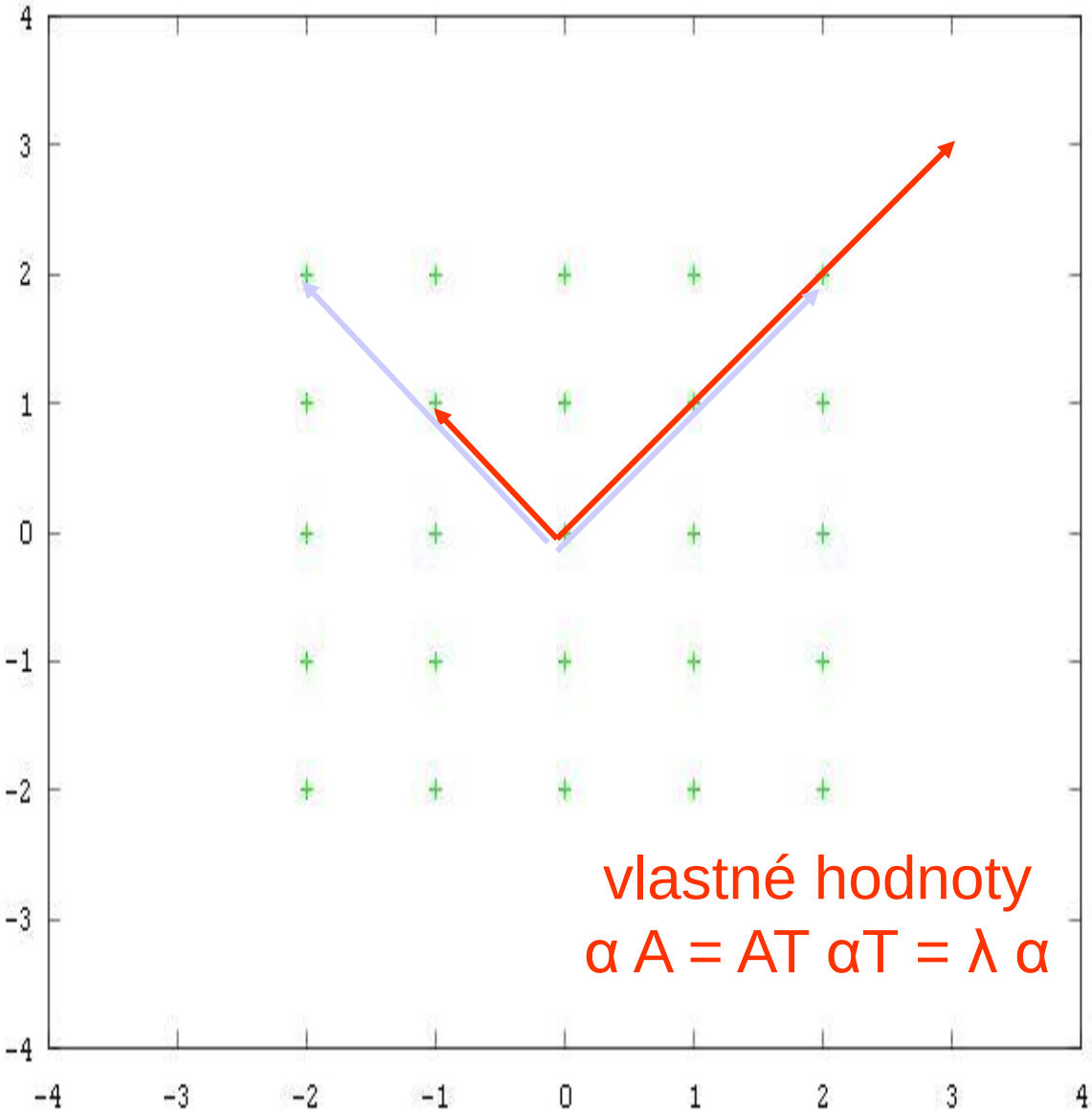


Keď transformujeme tieto body pomocou nejakej matice A



tak na svoj násobok sa zobrazia tzv. vlastné vektory

$$\frac{|\leftarrow|}{|\leftarrow|} = 1/2$$



$$\frac{|\leftarrow|}{|\leftarrow|} = 3/2$$

vlastné hodnoty udávajú ich zväčšovanie či zmenšovanie

vlastné vektory vieme efektívne vypočítať

Pre dimenziu 1,2,3 a 4 priamo vzorcom

Pre 5 a viac iteračným algoritmom, v najjednoduchšej verzii vezme akýkoľvek vektor a zobrazuje ho maticou, kým sa nezobrazí (skoro) na seba, potom sa v už len z na neho kolmých vektorov hľadá ďalší taký, až sa nájdu všetky. Ale sú k dispozícii algoritmy o mnoho rafinovanejšie a rýchlejšie.

Pomocou vlastných vektorov sa počíta všeličo, napr. aj rovnice piateho a vyšších stupňov, ...

A keďže ho používa LDA, fáza trénovania zo vzoriek je pri LDA – v porovnaní s inými metódami strojového učenia - neobvykle rýchla.

```
na=20;
a=stdnormal_rnd([na,2]);
a=a.*[1,0.3];
t=pi/3;
R = [cos(t) -sin(t); sin(t) cos(t)];
a=a*R;
a=a.+[3,4];
```

```
nb=20;
b=stdnormal_rnd([nb,2]);
b=b.*[1.2,0.4];
t=pi/4;
R = [cos(t) -sin(t); sin(t) cos(t)];
b=b*R;
b=b.+[2,2];
```

```
plot(a(:,1),a(:,2),"r.",b(:,1),b(:,2),"b.");
axis([0,6,0,6]);
```

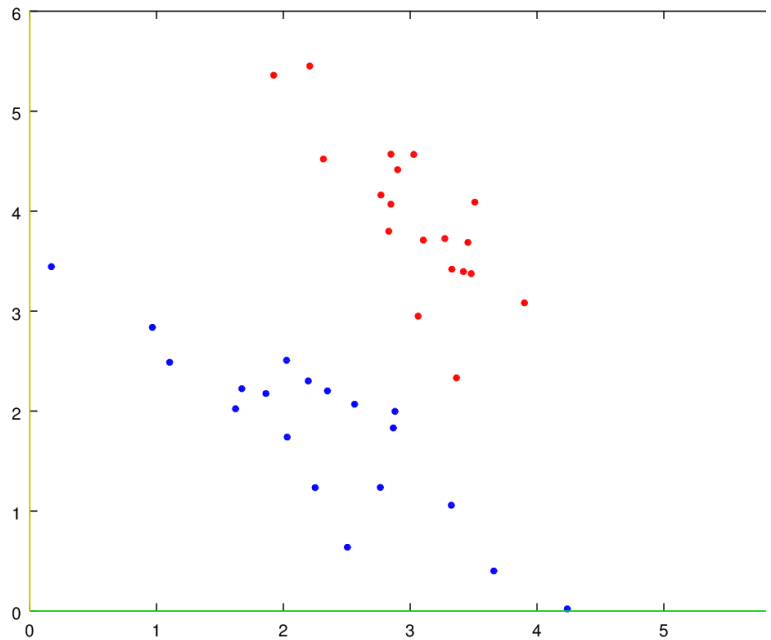
```
ma=mean(a);
mb=mean(b);
sa=cov(a);
sb=cov(b);
s=sa/na+sb/nb;
z=inv(s)*sb;
[w,lambda] = eig(z);
```

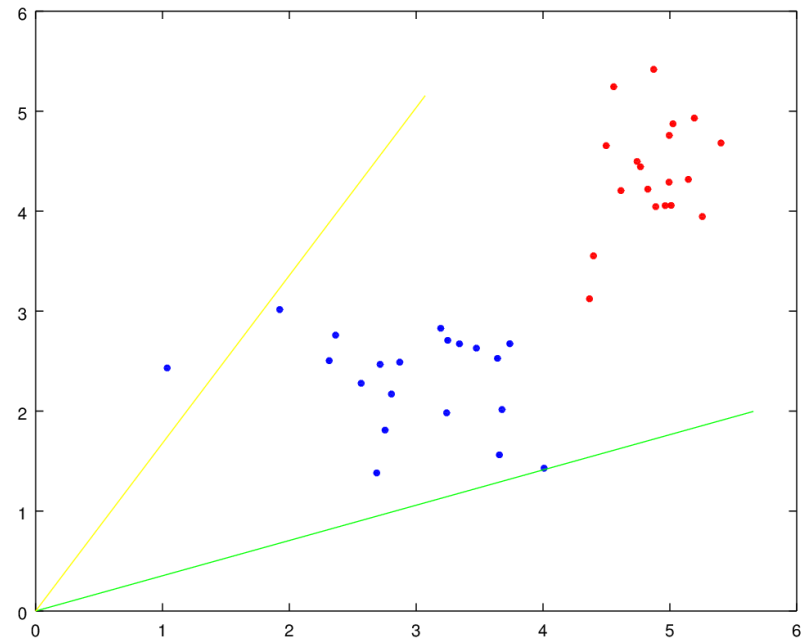
Pri LDA teda zo vzoriek spočítame správnu maticu, nájdeme jej vlastné vektory a ten s väčšou vlastnou hodnotou definuje tú správnu transformáciu

```
wa=a*w';
wb=b*w';
plot(wa(:,1),wa(:,2),"r.",wb(:,1),wb(:,2),"b.");
axis([0,6,0,6]);
```

```
plot(wa(:,1),zeros(na,1),"r.",wb(:,1),zeros(nb,1),"b.");
axis([0,6,0,1]);
```

Linear Discriminant Analysis (LDA)





Pomocou tohto triku LDA dokáže rozlišovať kategórie ktoré definuje pomocou vzoriek a to aj pri väčšej dimenzii priestoru a väčšom počte kategórií. Samozrejme len natoľko dokonale, ako to povaha vzoriek umožňuje.

LDA – dimenzia > 2

Pri väčšej dimenzii postupujeme analogicky. Nájdeme vlastné vektory a hodnoty príslušnej matice a tie vlastné, ktoré majú významne veľké vlastné hodnoty, zložíme po stĺpcoch do matice, ktorá bude maticou prechodu k priestoru s redukovanou dimenziou.

Prax ukazuje, že výrazných vlastných hodnôt je oveľa menej než je pôvodná dimenzia, takže redukcia dimenzie je výrazná.

LDA – viac kategórii

Pri viacerých kategóriách sa dá vymyslieť podobný postup.

Minimalizujeme pri ňom

$$J(W) = \frac{\sum_i (\mu_i^W - \mu^W)^2}{\sum_i (\sigma_i^W)^2}$$

kde i je kategória a μ^W je priemer všetkých vzoriek

Trénovanie

3F 2E ... FF	2A 2E ... 00	3E 2E ... F0	3F 2A ... EF	33 22 ... FF	3F 20 ... FE	3F 23 ... EF	00 01 ... AA	01 00 ... AE	01 00 ... AE
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
0	0	0	0	0	0	0	1	1	1

Sady fotiek roztriedené podľa osoby na fotke spracujeme na skupiny LBPH, každej skupine priradíme určité ID a pomocou LDA natrénujeme model, ktorý tieto osoby rozlišuje.

TrainRecognizer

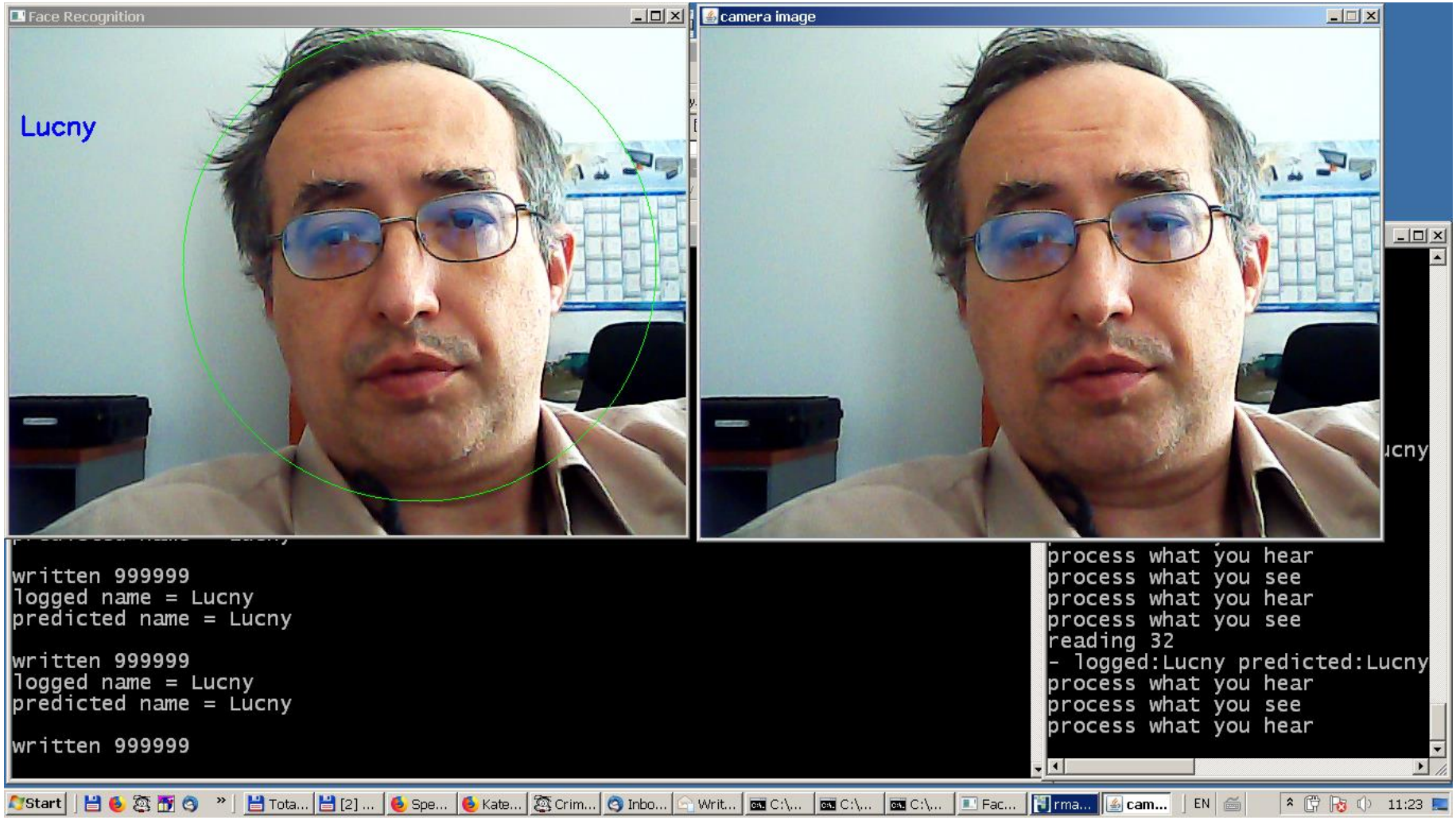
Testovanie

Tento model môžeme potom použiť tak, že v každom obrázku z kamery hľadáme tvár, k nej príslušný LBPH a ten vložíme do modelu nech povie jeho ID.

Pritom je vhodné stabilizovať facial landmarky (napr priemerovať viaceré posledné hodnoty) i predikované ID (napríklad medián z viacerých posledných hodnôt

TestRecognizer

Testovanie


Face Recognition
Lucny
camera image
written 999999
logged name = Lucny
predicted name = Lucny
written 999999
logged name = Lucny
predicted name = Lucny
written 999999
process what you hear
process what you see
process what you hear
process what you see
reading 32
- logged:Lucny predicted:Lucny
process what you hear
process what you see
process what you hear
Start
11:23

On-line rozpoznávanie

Keďže trénovanie je rýchle, môžeme opraviť chyby zistené pri testovaní, hneď ako ich zistíme, pretrénovať model a vylepšovať ho.

OnlineRecognizer

- Metódu LDA (pre dve kategórie) vymyslel Ronald Fisher v roku 1936

- LBPH navrhli T. Ojala, M. Pietikäinen, and D. Harwood v roku 1994

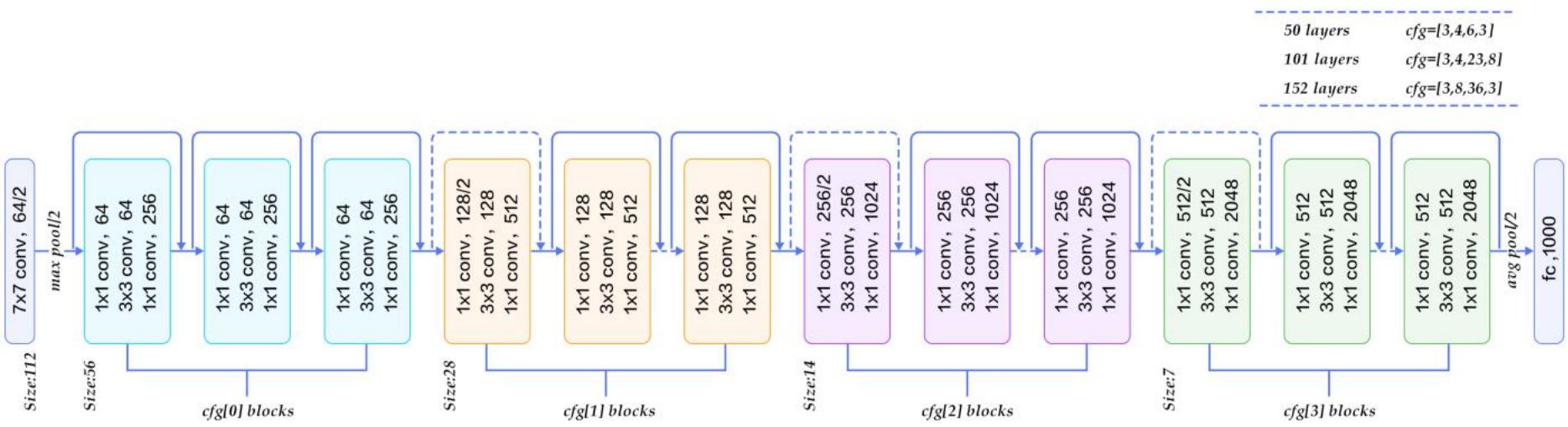
- Open source knižnica na strojové učenie a počítačové videnie
- Autor: Davis King
- Boost Software License
- Linux, Windows, Mac
- CUDA based
- Od roku 2002

- Open source knižnica na počítačové videnie
- Intel, Willow Garage, Itseez (Gary Bradski)
- BSD Licencia
- 2500 rôznych algoritmov
- C++, Java, Python
- Windows, Linux, Android
- Je to fakt rýchle spoľahlivé
- Podpora CUDA a OpenCL

Veľa pekných zdrojákov sa dá získať za registráciu zdarma z článkov na www.learnopencv.com a ich projekte na GitHub-e

Nabudúce...

- Poznáme aj iné metódy rozpoznávania tvárí, založené na hlbokom učení.
- Sieť ResNet bola naučená dať k obrázku jeho kód, pričom podobné obrázky majú podobné kódy a čím sa obrázky viac líšia, tým rozličnejšie kódy by mali dostať.



Ďakujeme za pozornosť

Seminár Robotika.SK

Ako naučiť robota rozpoznávať kto je kto pomocou knižnice Dlib

Andrej Lúčny

Katedra aplikovanej informatiky FMFI UK

lucny@fmph.uniba.sk

http://dai.fmph.uniba.sk/w/Andrej_Lucny