

Biometrický systém na báze ľudskej tváre

Zadanie č.4

Biometria

Obsah

- Zadanie č.4
 - Inštrukcie
 - Model riešenia a základné pojmy
 - Bodovanie

Inštrukcie 1/2

- Znenie zadania:

- Implementujte **dvoj-fázový** biometrický systém, ktorý rozpoznáva osobu pomocou obrazu tváre
- Dvoj-fázový systém:

Fáza č. 1 Extrakcia charakteristických znakov tváre pomocou PCA metódy

Fáza č. 2 Klasifikácia extrahovaných znakov pomocou 3 techník

Inštrukcie 2/2

- Znenie zadania:
 - Klasifikačné techniky, ktoré bude váš softvér vykonávať:
 1. **SVM** (Support Vector Machine)
 2. **Euklidova** vzdialenosť
 3. **Mahalanobisova** vzdialenosť
 - Cieľom je porovnanie týchto klasifikačných techník pomocou **percentuálnej úspešnosti klasifikácie v závislosti od rastúceho počtu hlavných komponentov PCA**
 - Zvoľte si postupne zvyšujúci sa počet hlavných komponentov, napr. 1, 2 až 100)
 - Pre každú konfiguráciu PCA metódy vykonajte **aspoň 3 kolá cross-validácie**

Model riešenia a základné pojmy

- Databáza obrazov tváre A
 - Názov databázy je ***faces*** (nahraná v AIS)
 - Obsahuje 15 osôb (10 obrazov tváre/osoba)
 - Rozmery sú Š(92) x V(112)
 - Formát názvu **XX_YY**

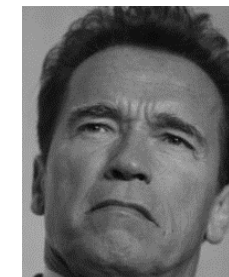
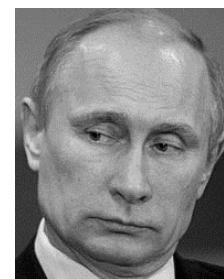
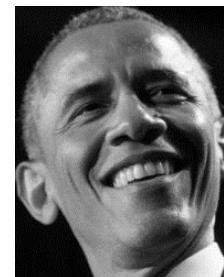
ID osoby ↑
ID fotky ↑



Model riešenia a základné pojmy

- Databáza obrazov tváre B
 - Názov databázy je *famous_faces* (nahraná v AIS)
 - Obsahuje 6 známych osôb (12 obrazov tváre/osoba)
 - Rozmery sú Š(200) x V(250)
 - Formát názvu *XX_YY*

meno osoby ↑
ID fotky ↑



Model riešenia a základné pojmy

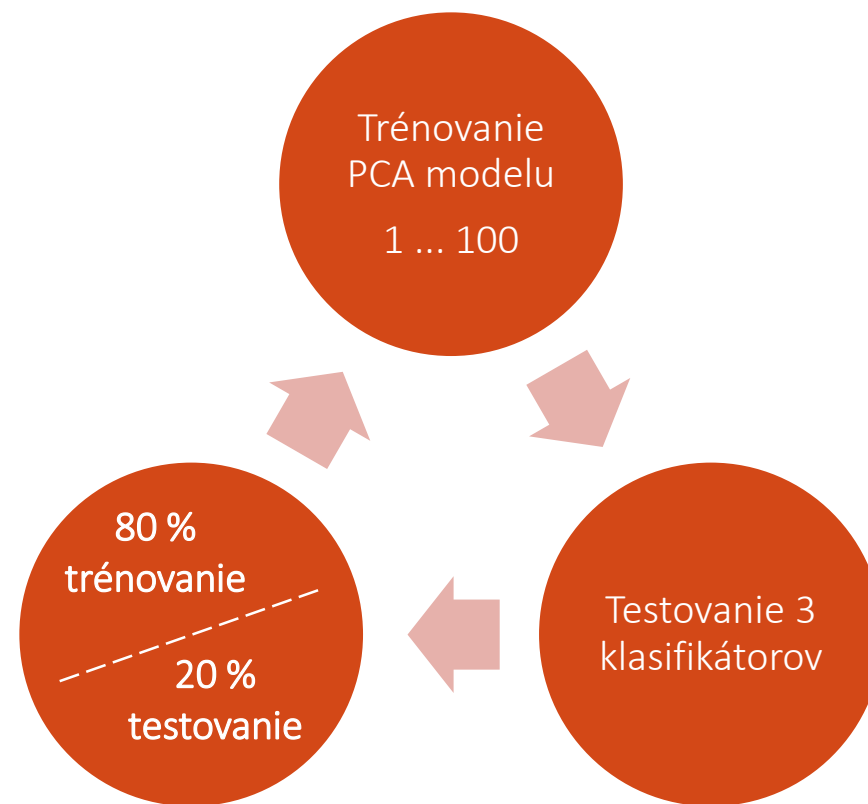
- Fáza č.1 – extrakcia znakov použitím PCA metódy
 - Použite dostupné implementácie PCA (napr. OpenCV, Matlab ...)
 - V zadaní si vytvorte konfiguráciu PCA modelu podľa počtu „eigenvektorov“ resp. hlavných komponentov

```
cv::PCA::PCA ( InputArray data,  
               InputArray mean,  
               int      flags,  
               int      maxComponents = 0  
             )
```

PCA v knižnici OpenCV

Model riešenia a základné pojmy

- Fáza č.1 – extrakcia znakov použitím PCA metódy
 - Pre každú konfiguráciu PCA modelu vykonajte aspoň 3 kolá cross-validácie



Cyklus zopakovať aspoň 3-krát

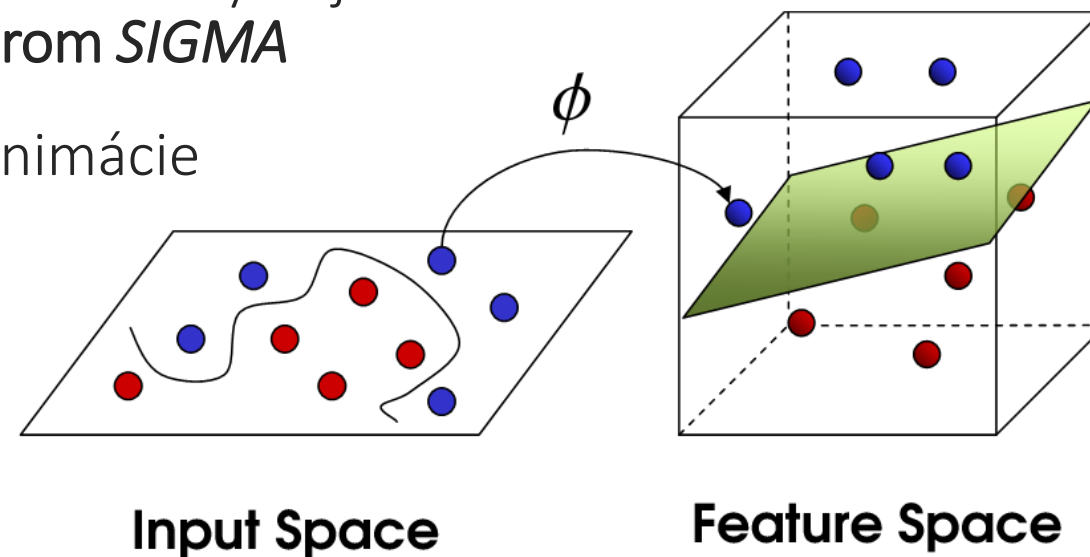
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia
 - Klasifikujú sa PCA koeficienty získané po projekcii pôvodných údajov do PCA priestoru
 - V prípade OpenCV je to vektor PCA koeficientov vrátený metódou *PCA::project()*
 - Klasifikujeme všetky obrazy tváre v testovacej množine

Model riešenia a základné pojmy

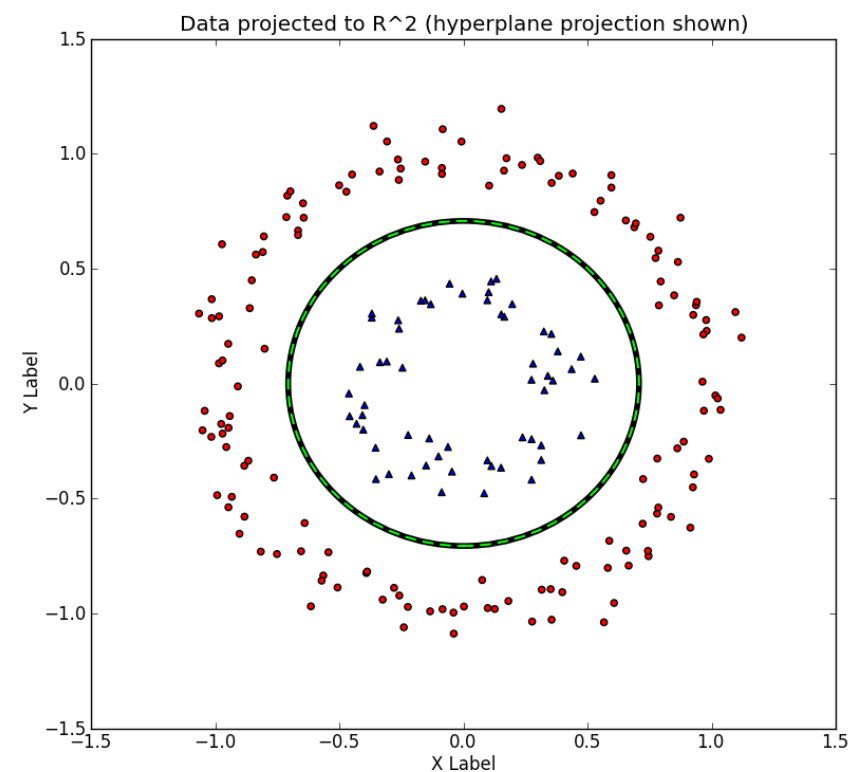
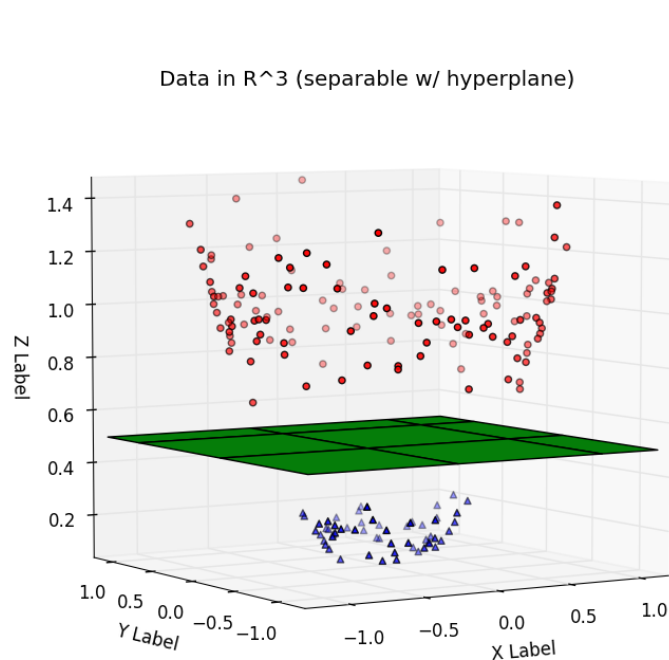
- Fáza č.2 – klasifikácia pomocou SVM

- Použite **RBF ako kernel funkciu** (kernel = funkcia vyjadrujúca podobnosť 2 vzoriek v priestore vyššej dimenzie), **experimentujte s parametrom *SIGMA***
- Vysvetlenie “kernel” triku vo forme animácie (<http://y2u.be/BE1M1xxr03s>)



Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou SVM



Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou SVM
 - Snažíme sa vypočítať $K(x, y) = \Phi(x) \cdot \Phi(y)$, čo môže byť výpočtovo náročné
 - Príklad:
 - mapovanie je $\Phi(x) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$
 - výpočtovo jednoduchší je tzv. „kernel trick“ vo forme $K(x, y) = (x \cdot y)^2$
 - Všeobecný polynomiálny kernel $K(x, y) = ((x \cdot y) + \theta)^d$
 - Všeobecný RBF kernel $K(x, y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}$

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou SVM

- Implementácia SVM pre používateľov **Matlabu**:

- <http://www.mathworks.com/matlabcentral/fileexchange/33170-multi-class-support-vector-machine> (používa RBF ako kernel funkciu)

- Implementácia SVM pre používateľov **OpenCV 3+** :

- http://docs.opencv.org/3.1.0/d1/d2d/classcv_1_1ml_1_1SVM.html#gsc.tab=0

- http://docs.opencv.org/3.0-beta/modules/ml/doc/support_vector_machines.html

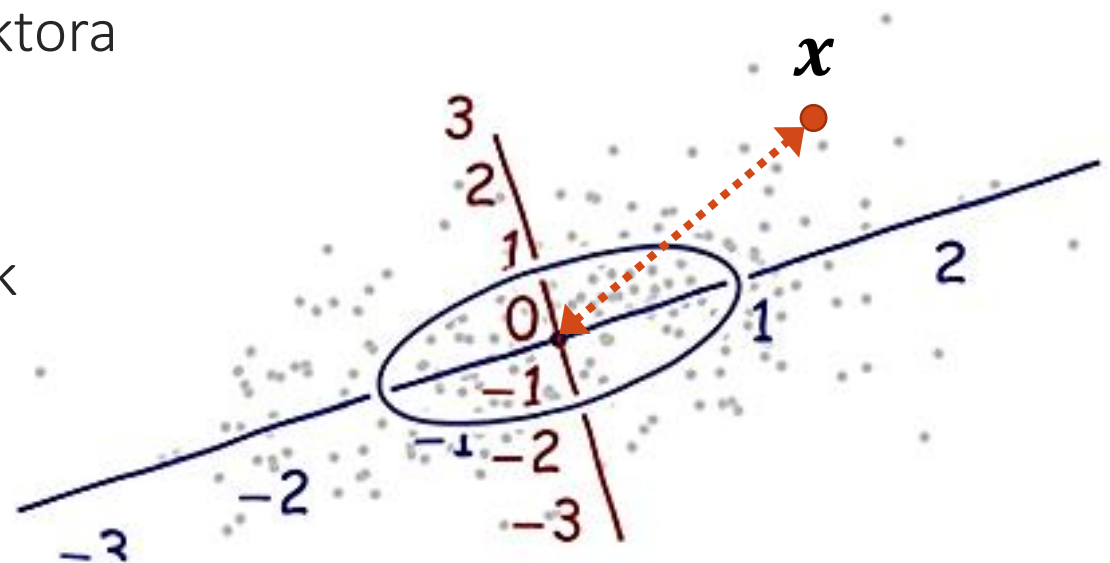
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Euklidovej vzdialenosti
 - Porovnávame 2 vektory PCA koeficientov pomocou Euklidovej vzdialenosti
 - Pri identifikácii hľadáme v trénovacej databáze vektor s najmenšou Euklidovou vzdialenosťou od neznámeho testovacieho vektora
 - Euklidovu vzdialenosť vektorov x a y vypočítame nasledovne:

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2}$$

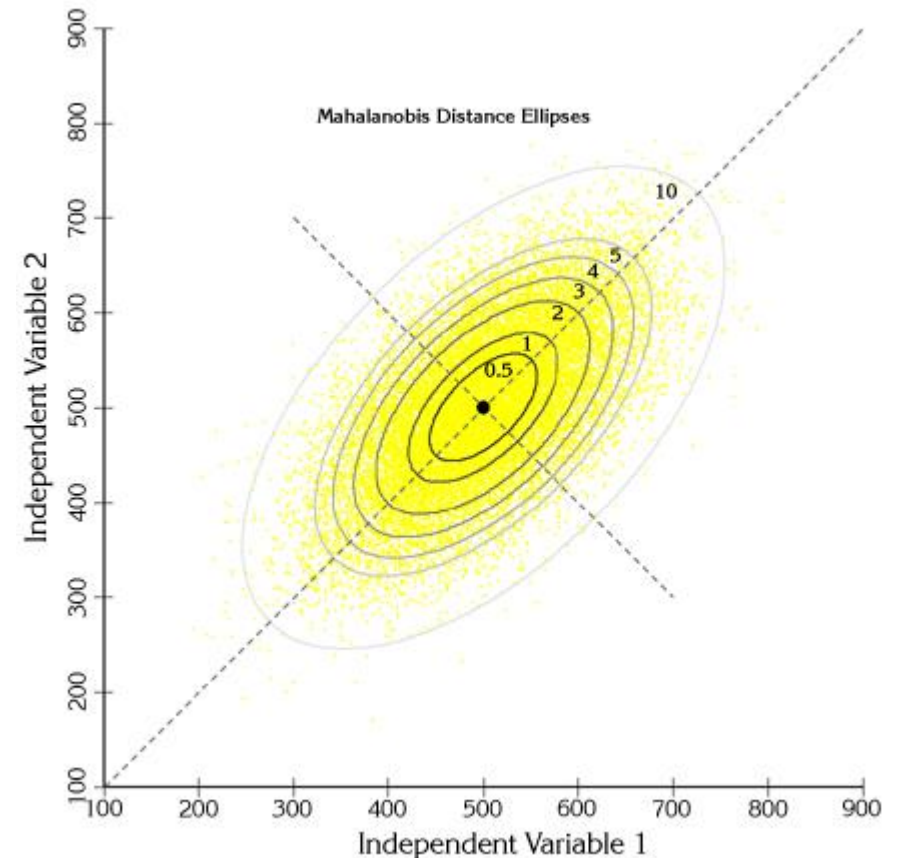
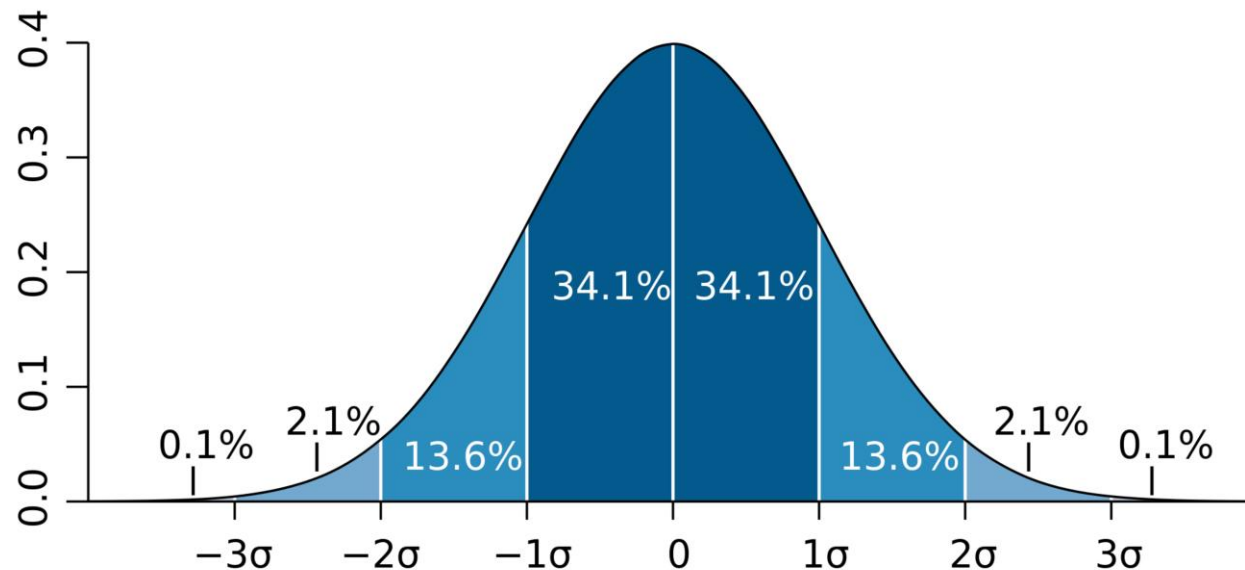
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti
 - Je to forma vyjadrenia vzdialenosti vektora x od stredu pravdepodobnostného rozdelenia P
 - Meria sa počet štandardných odchýlok medzi x a P



Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti

- Výpočet:

$$d(x) = \sqrt{(x - \mu_K) S^{-1} (x - \mu_K)^T}$$

x	testovací vektor vyjadrený PCA koeficientami
μ_K	stredná hodnota PCA koeficientov fotiek v konkrétnej triede K
S^{-1}	inverzná matica ku kovariančnej matici trénovacích vektorov v triede K (<i>každý trénovací vektor je vyjadrený PCA koeficientami</i>)

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti
 - Názorná ukážka v Matlabe (Mahalanobisova vzdialenosť x od stredu rozdelenia náhodne zvolených údajov)
 - Manuálny výpočet

```
x =  
  
      5      9      7
```

```
train_sample =  
  
      8      1      6  
      3      5      7  
      4      9      2  
      8      5      2  
      9      6      3  
      1      4      7
```

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti

1

$\mathbf{x} =$
5 9 7

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti

1

x =		
5	9	7

2

train_sample =		
8	1	6
3	5	7
4	9	2
8	5	2
9	6	3
1	4	7

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti

1

```
x =  
  
      5      9      7
```

2

```
train_sample =  
  
      8      1      6  
      3      5      7  
      4      9      2  
      8      5      2  
      9      6      3  
      1      4      7
```

3

```
>> mean_value = mean(train_sample)  
  
mean_value =  
  
      5.5000      5.0000      4.5000
```

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti

1

```
x =  
  
      5      9      7
```

2

```
train_sample =  
  
      8      1      6  
      3      5      7  
      4      9      2  
      8      5      2  
      9      6      3  
      1      4      7
```

3

```
>> mean_value = mean(train_sample)  
  
mean_value =  
  
      5.5000      5.0000      4.5000
```

4

```
>> sqrt((x-mean_value)*inv(cov(train_sample))*(x-mean_value)')  
  
ans =  
  
      5.6088
```

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti

1

```
x =  
  
      5      9      7
```

2

```
train_sample =  
  
      8      1      6  
      3      5      7  
      4      9      2  
      8      5      2  
      9      6      3  
      1      4      7
```

3

```
>> mean_value = mean(train_sample)  
  
mean_value =  
  
      5.5000      5.0000      4.5000
```

4

```
>> sqrt((x-mean_value)*inv(cov(train_sample))*(x-mean_value)')  
  
ans =  
  
      5.6088
```

$$d(x) = \sqrt{(x - \mu_K) S^{-1} (x - \mu_K)^T}$$

Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti
 - Použitie knižničnej funkcie

```
>> sqrt(mahal(x,train_sample))
```

```
ans =
```

```
5.6088
```

```
x =
```

```
5
```

```
9
```

```
7
```

```
train_sample =
```

```
8      1      6
```

```
3      5      7
```

```
4      9      2
```

```
8      5      2
```

```
9      6      3
```

```
1      4      7
```


Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti
 - Stručný návod ako použiť Mahalanobisovu vzdialenosť na klasifikovanie obrazu tváre T_1 :
 1. Vytvoríme PCA model z tréningových dát
 2. Po projekcii získame PCA koeficienty tváre T_1 (napr. pomocou *project()* v OpenCV)
 3. Vypočítame Mahalanobisovu vzdialenosť PCA koeficientov tváre T_1 postupne od stredu pravdepodobnostného rozdelenia PCA koeficientov každej triedy v tréningovej množine

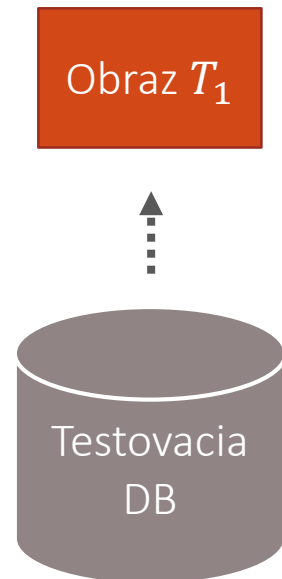
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



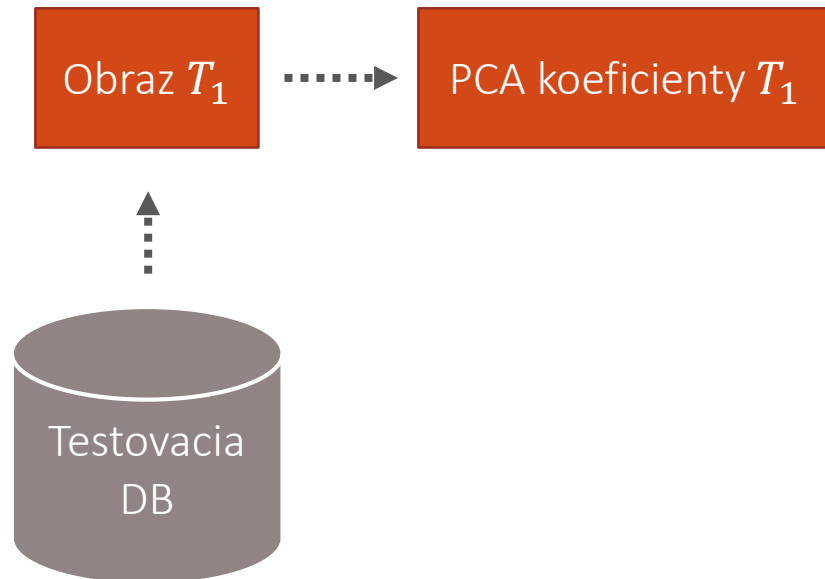
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



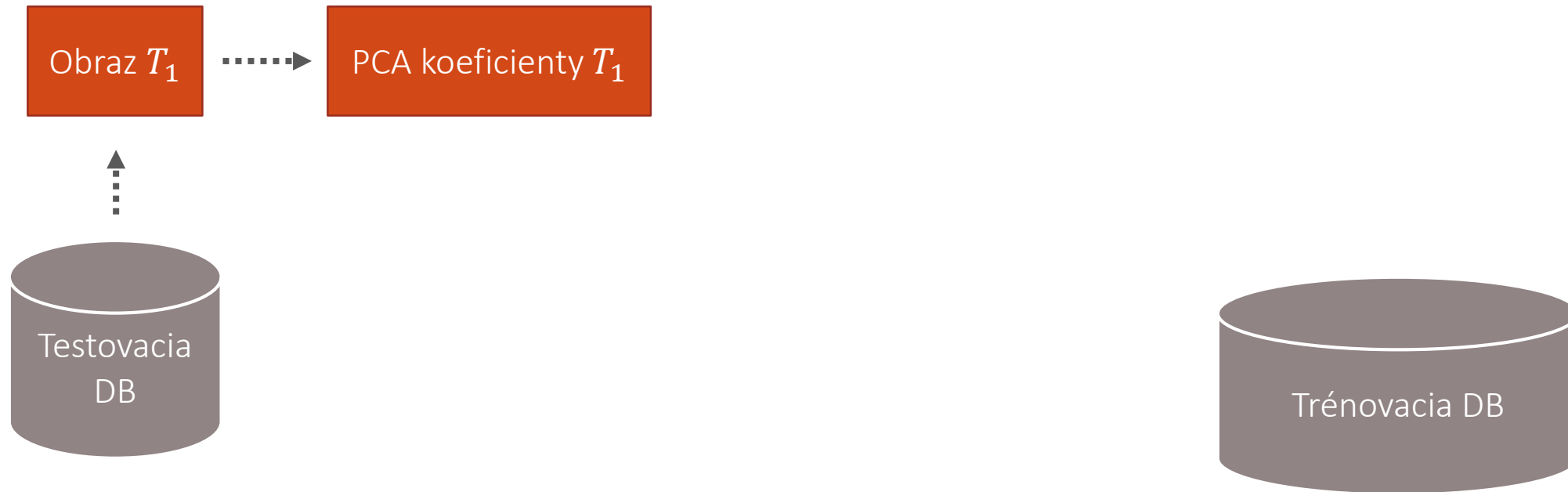
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



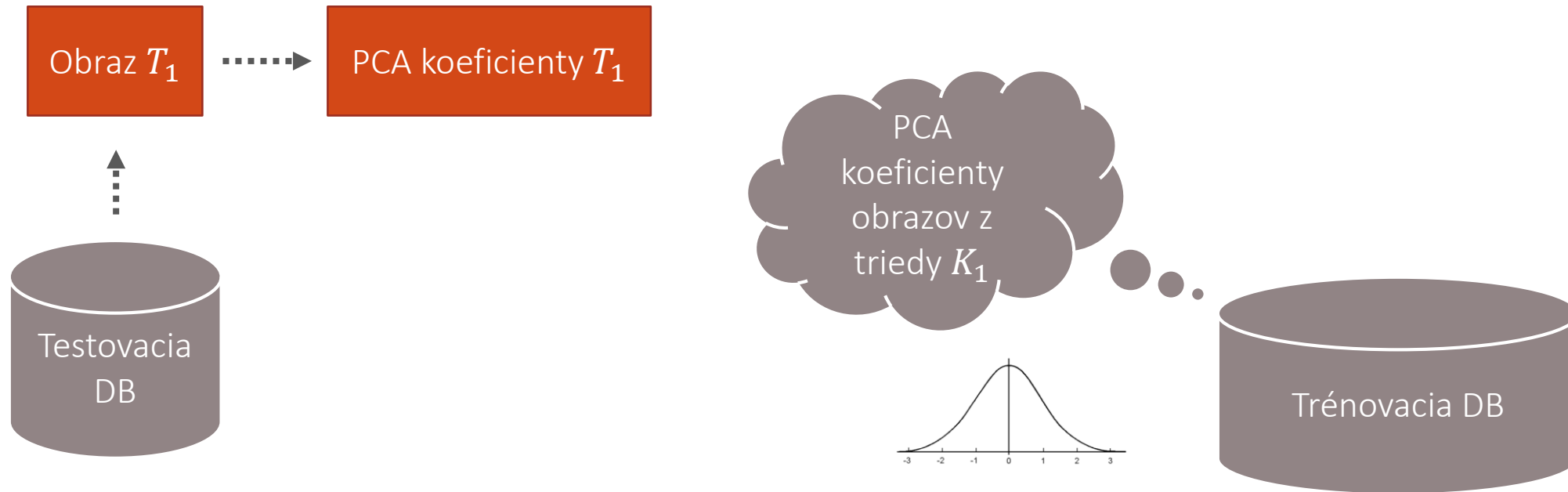
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



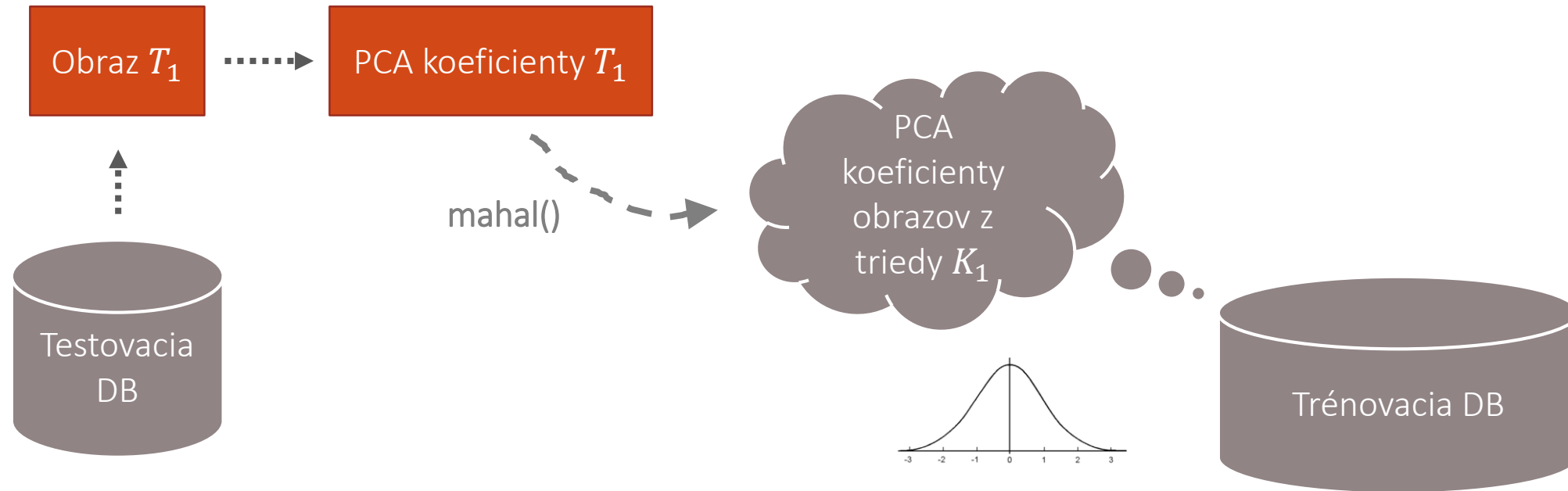
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



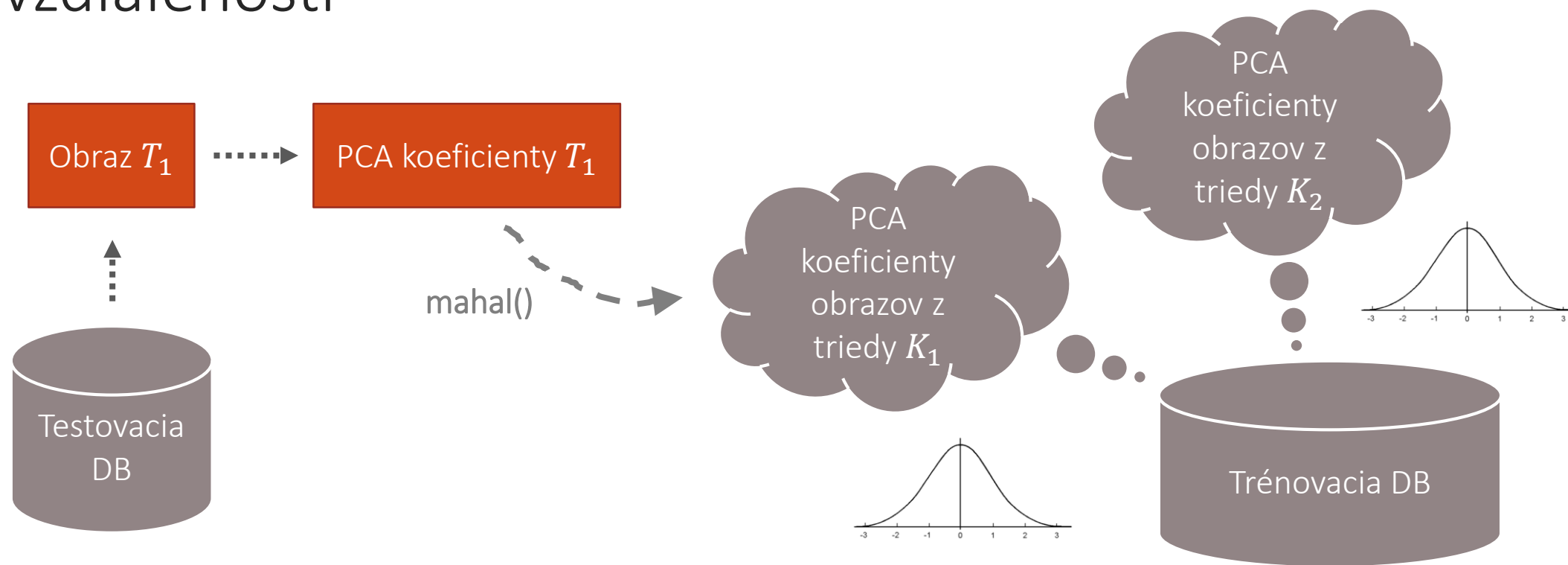
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



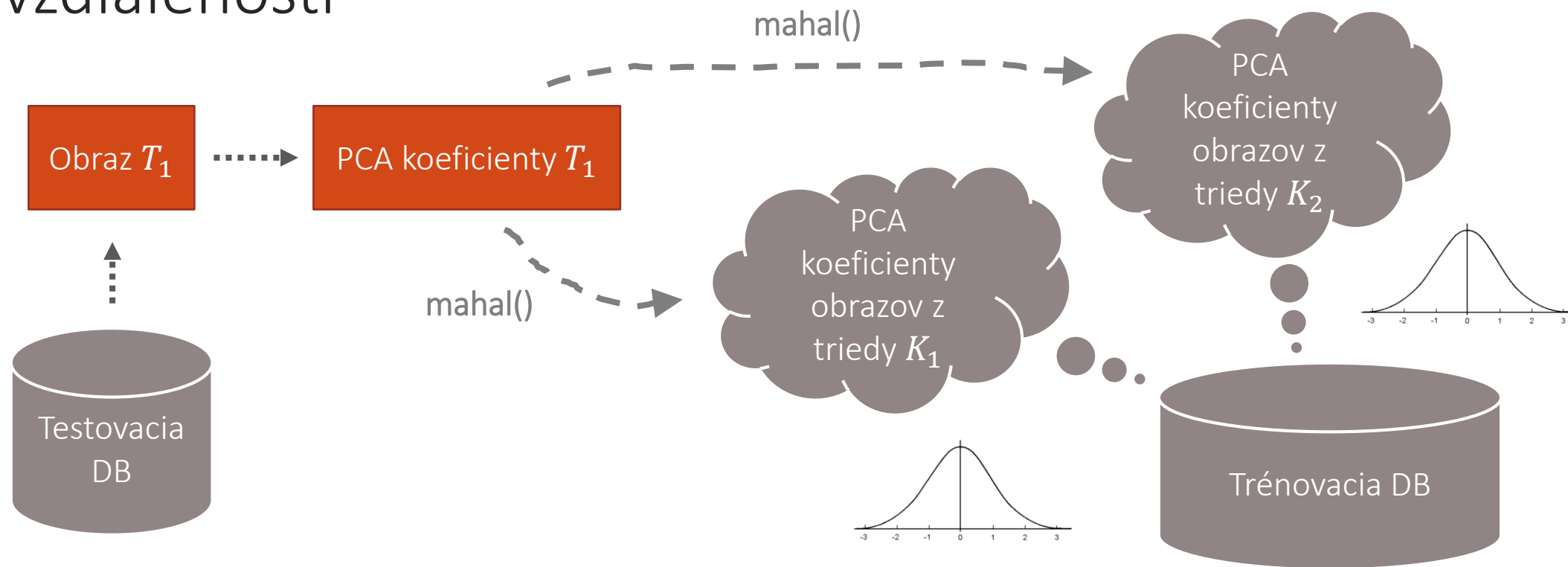
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



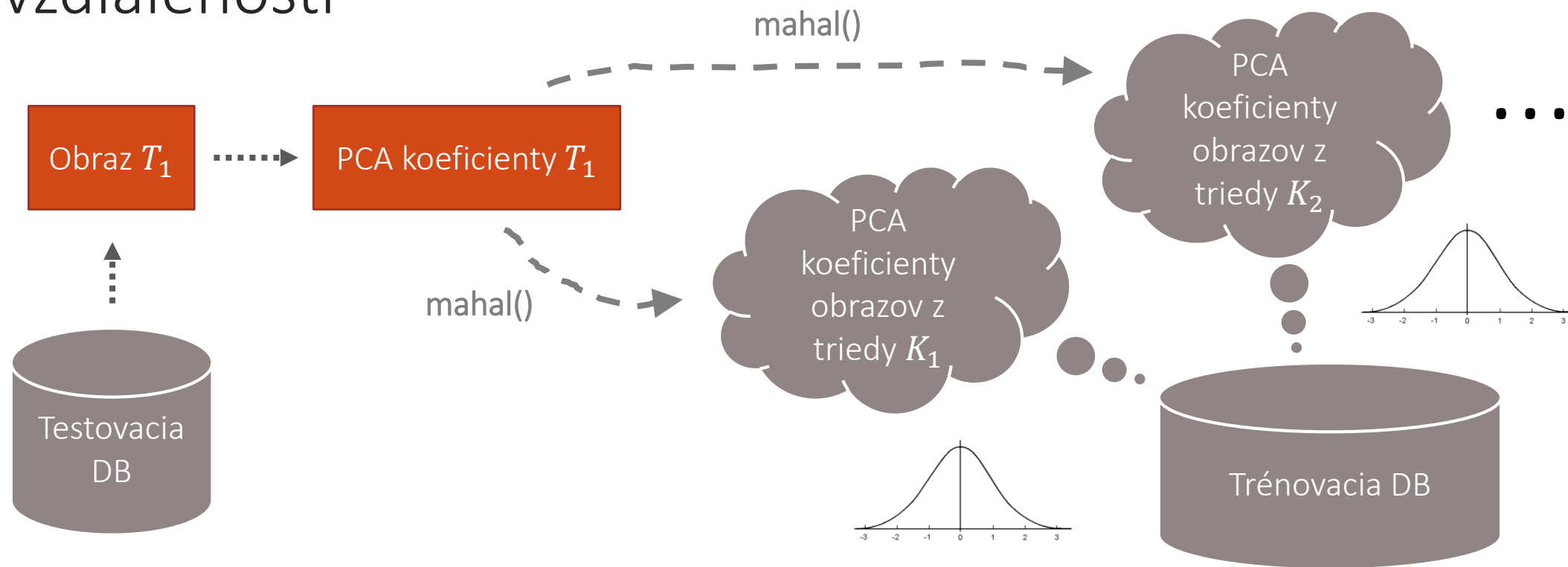
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



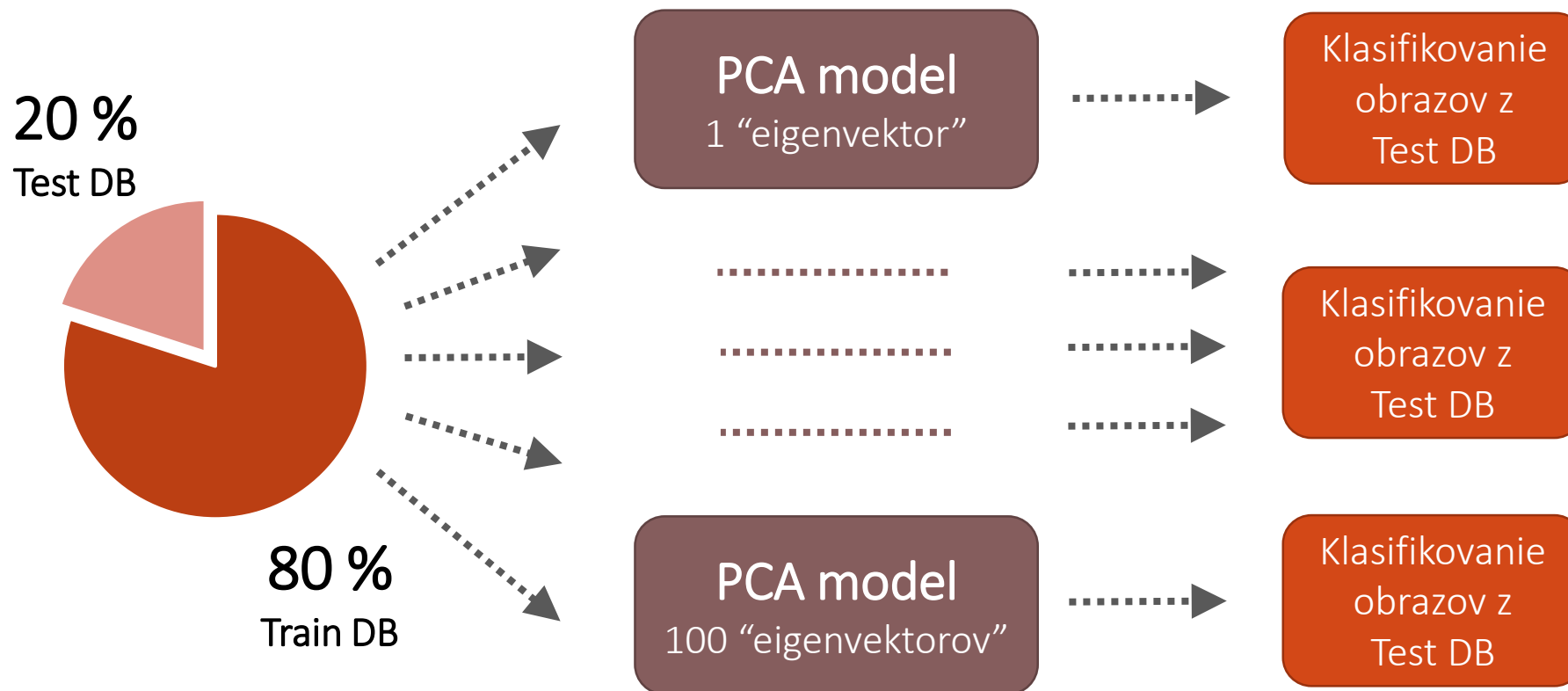
Model riešenia a základné pojmy

- Fáza č.2 – klasifikácia pomocou Mahalanobisovej vzdialenosti



Model riešenia a základné pojmy

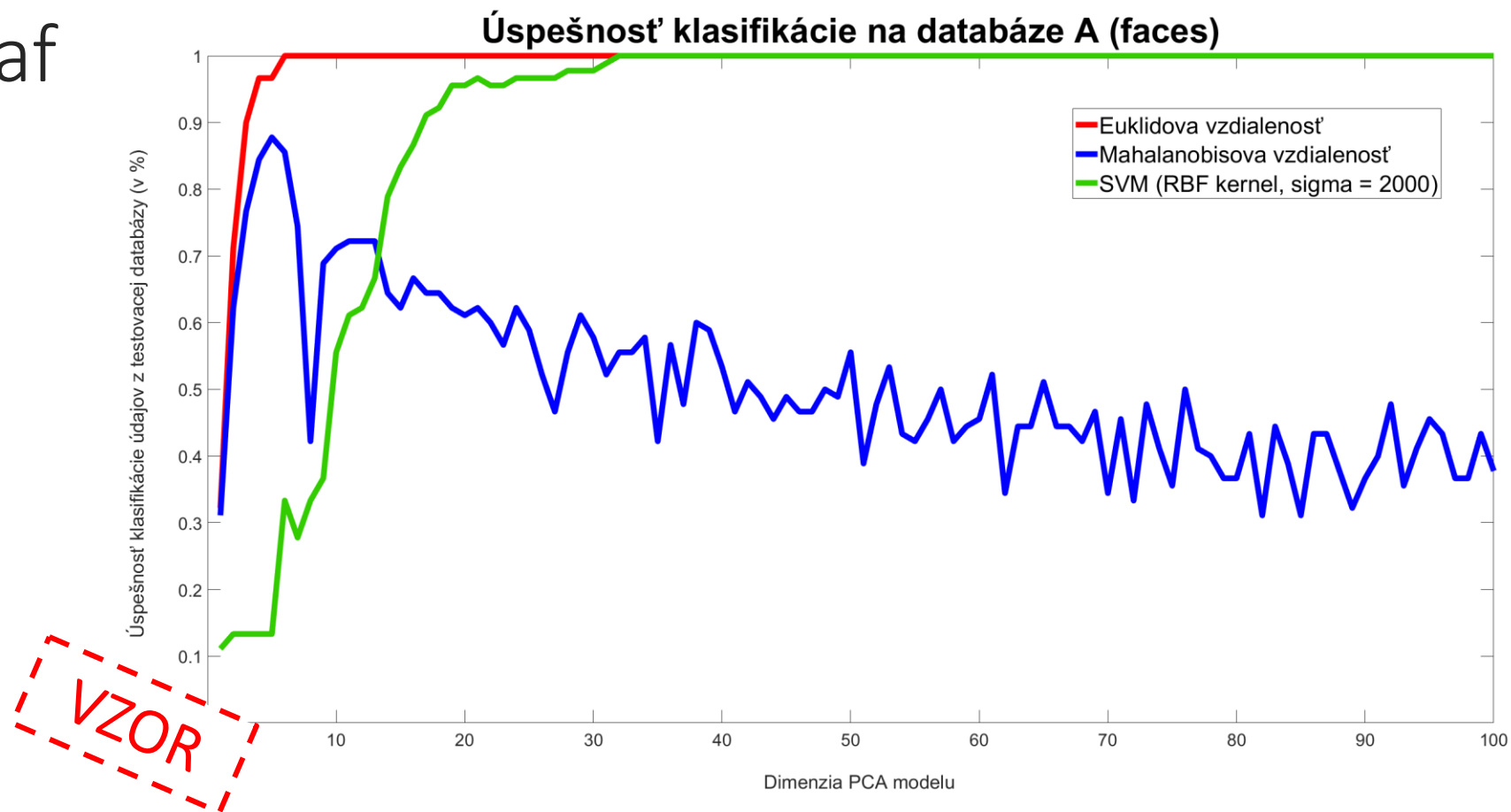
○ Sumarizácia



Model riešenia a základné pojmy

- Výstupný graf

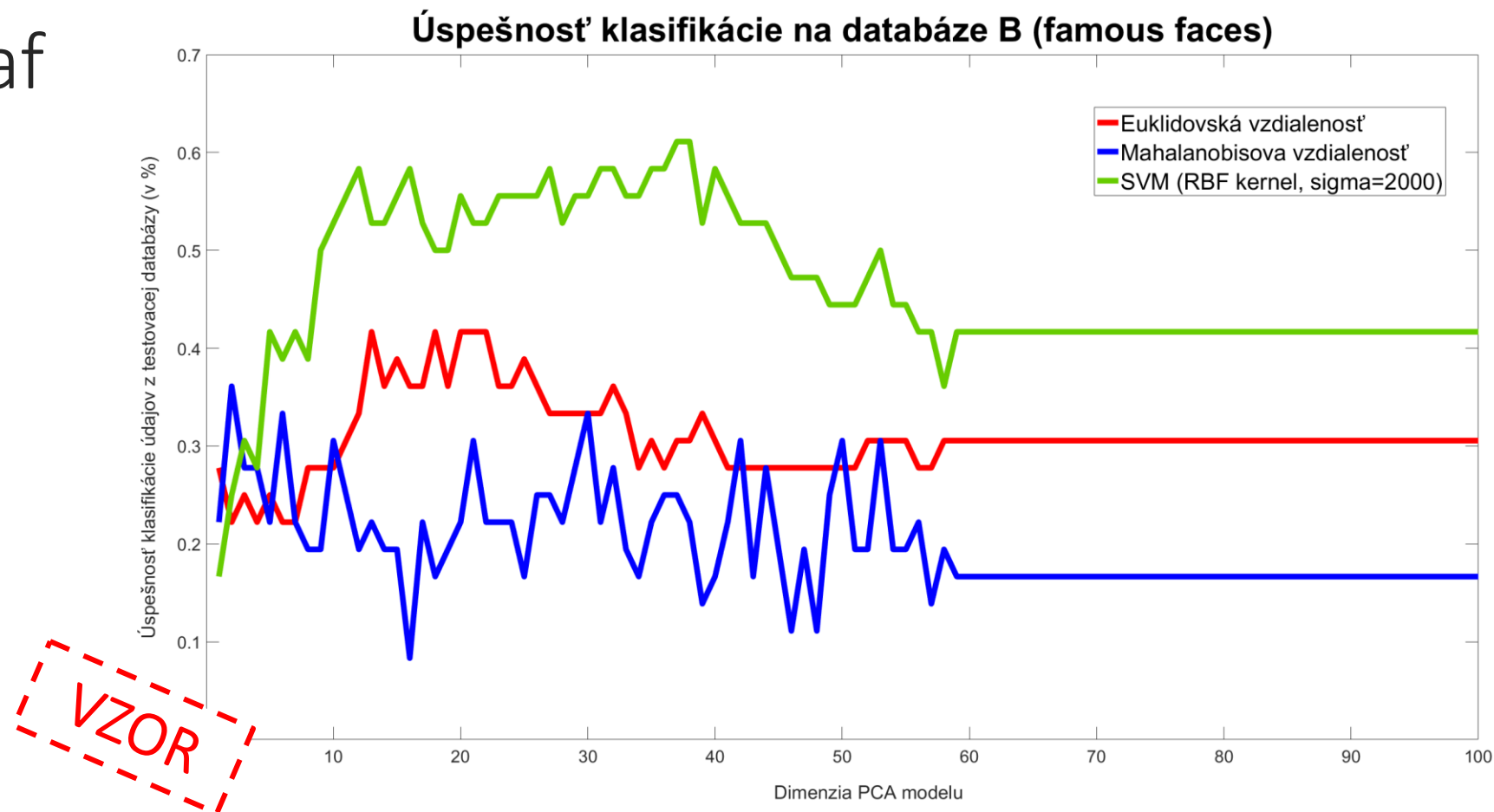
- Databáza A
- TrainSize = 8
- TestSize = 2



Model riešenia a základné pojmy

- Výstupný graf

- Databáza B
- TrainSize = 10
- TestSize = 2



Bodovanie

Použitá cross-validácia	1b
Správne použitie PCA metódy	1b
Odprezentovanie použitých klasifikátorov: <ul style="list-style-type: none">• SVM• Euklidova vzdialenosť• Mahalanobisova vzdialenosť	
	1b
	1b
	3b
Graf úspešnosti klasifikátorov	3b
Spolu	10b