۹۵۹۷۰۳۹ امیرحسین عباسکوهی

۵

۹

(+1) data → (1,1) , (2,1) , (1,2)

(-1) data → (2,2) , (3,2)

$$\mu_{+1} = \left(\frac{1+2+1}{3}, \frac{1+1+2}{3}\right) = \left(\frac{4}{3}, \frac{4}{3}\right)$$

$$\mu_{-1} = \left(\frac{2+3}{2}, \frac{2+2}{2}\right) = \left(\frac{5}{2}, \frac{4}{2}\right)$$

$$\mu = \left(\frac{1+2+1+2+3}{5}, \frac{1+1+2+2+2}{5}\right) = \left(\frac{9}{5}, \frac{8}{5}\right)$$

$$\mu_{+1} - \mu = (-0.47, -0.27)$$

$$\mu_{-1} - \mu = (0.7, 0.4)$$

$$S_b = \sum_{k=1}^{2} (m_k - m) N_k (m_k - m)^T \quad , \quad N_{+1} = 3 \, , \, N_{-1} = 2$$

$$(\mu_{+1} - \mu)(\mu_{+1} - \mu)^T = \begin{bmatrix} -0.47 \\ -0.27 \end{bmatrix} \begin{bmatrix} -0.47 & -0.27 \end{bmatrix} = \begin{bmatrix} 0.2209 & 0.1269 \\ 0.1269 & 0.0729 \end{bmatrix}$$

$$(\mu_{-1} - \mu)(\mu_{-1} - \mu)^T = \begin{bmatrix} 0.7 \\ 0.4 \end{bmatrix} \begin{bmatrix} 0.7 & 0.4 \end{bmatrix} = \begin{bmatrix} 0.49 & 0.28 \\ 0.28 & 0.16 \end{bmatrix}$$

$$S_b = \begin{bmatrix} 0.6627 & 0.3807 \\ 0.3807 & 0.2187 \end{bmatrix} + \begin{bmatrix} 0.98 & 0.56 \\ 0.56 & 0.32 \end{bmatrix} = \begin{bmatrix} 1.6427 & 0.9407 \\ 0.9407 & 0.5387 \end{bmatrix}$$

b

$\mu_{+1} = (4/3, 4/3)$ (+1) data => (1,1), (1,2), (2,1)

$\mu_{-1} = (5/2, 4/2)$ (-1) data => (2,2), (3,2)

$$S_i = \sum_{n=1}^{N_k} (x_{nk} - m_k)(x_{nk} - m_k)^T$$

$$S_{+1} = \left(\begin{bmatrix} -0.34 \\ -0.34 \end{bmatrix} [-0.34 \ -0.34] \right) + \left(\begin{bmatrix} -0.34 \\ 0.67 \end{bmatrix} [-0.34 \ \ 0.67] \right)$$

$$+ \left(\begin{bmatrix} 0.67 \\ -0.34 \end{bmatrix} [0.67 \ \ -0.34] \right)$$

$$= \left(\begin{bmatrix} 0.1156 & 0.1156 \\ 0.1156 & 0.1156 \end{bmatrix} \right) + \left(\begin{bmatrix} 0.1156 & -0.2278 \\ -0.2278 & 0.4489 \end{bmatrix} \right) + \left(\begin{bmatrix} 0.4489 & -0.2278 \\ -0.2278 & 0.1156 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 0.6801 & -0.34 \\ -0.34 & 0.6801 \end{bmatrix}$$

$$S_{-1} = \left(\begin{bmatrix} 0.5 \\ 0 \end{bmatrix} [0.5 \ \ 0] \right) + \left(\begin{bmatrix} -0.5 \\ 0 \end{bmatrix} [-0.5 \ \ 0] \right)$$

$$= \left(\begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} \right) + \left(\begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} \right) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0 \end{bmatrix}$$

$$S_w = S_{-1} + S_{+1} = \begin{bmatrix} 1.1801 & -0.34 \\ -0.34 & 0.6801 \end{bmatrix}$$

$$S_w^{-1} S_b \rightarrow S_w^{-1} = \frac{1}{0.68} \begin{pmatrix} 0.6801 & 0.34 \\ 0.34 & 1.1801 \end{pmatrix}$$

$$\cong \begin{pmatrix} 0.99 & 0.5 \\ 0.5 & 1.72 \end{pmatrix}$$

$$S_w^{-1} S_b = \begin{pmatrix} 0.99 & 0.5 \\ 0.5 & 1.72 \end{pmatrix} \begin{pmatrix} 1.6427 & 0.3407 \\ 0.3407 & 0.5387 \end{pmatrix}$$

$$= \begin{pmatrix} 2.1 & 1.2 \\ 2.44 & 1.4 \end{pmatrix}$$

$$A - \lambda I = \begin{pmatrix} 2.1-\lambda & 1.2 \\ 2.44 & 1.4-\lambda \end{pmatrix} \xrightarrow{\det} (2.1-\lambda)(1.4-\lambda) - (1.2)(2.44)$$

$$= 2.94 + \lambda^2 - 3.5\lambda - 2.928$$

$$= 0.012 + \lambda^2 - 3.5\lambda$$

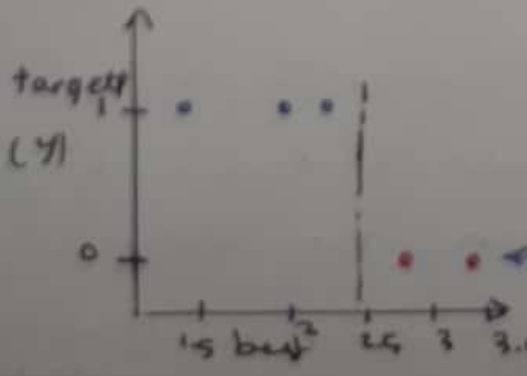
$$\xrightarrow{\approx} \lambda_1 = 3.5 \,, \; \lambda_2 = 3.43 \quad \text{eigenvalues}$$

eigenvector $\rightarrow$ $v_1 = (0.65 \quad -0.5)$
(normalized)

$v_2 = (0.76 \quad 0.87)$

$w = (0.65 \,, 0.76)$

$X\text{-transformed} = X \; w = [\, 1.41 \,, 2.06 \,, 2.16 \,, 2.82 \,, 3.47 \,]$



$\rightarrow$ بله target

۱)

a) انواع روش های جستجو در انتخاب فیچر:
(۱) exhaustive
(۲) heuristic
(۳) randomized

روش های ارزیابی:
(۱) filter methods
(۲) wrapper methods

تفاوت ها

(۱) در filter methods ارزیابی مستقل از الگوریتم طبقه‌بندی است
اما در wrapper از حوزه ارزیابی مرتبط با الگوریتم طبقه بندی استفاده می‌شود

(۲) در filter هدف ارزیابی زیرمجموعه ویژگی ها بر اساس information content (مثل dependence, interclass distance)
اما در wrapper، تابع هدف یک pattern classifier است که زیرمجموعه ویژگی ها را براساس predective accuracy ارزیابی می‌کند

(۳) روش filter سریع و جامع است اما wrapper کند است اما دقیق است (جامع هم نیست)

b) PCA و LDA در دو محاسبات ماتریسی سنگینی دارند مثل ماتریس کوواریانس، eigen value و eigen vector ها و موارد اینچنینی. تمام این موارد نیاز به load شدن تمام موارد در memory دارند و این علاوه بر مشکل زمانی می‌تواند با مشکل حافظه هم روبه رو کند به خصوص که ابعاد dimension در این موارد کاملاً متحدد و غیر خطی است. همچنین در این موارد به Sparse matrix هم برمی خوریم و ممکن است در محاسبات با det=0

ه)

رو به رو شدیم که برای مثال در LDA محاسبات را معتل می‌کند.

---

۲)

a) فرض کنید x_{ij} ، j امین داده از کلاس i ام باشد که کلاس i ام n_i داده دارد. حال داریم:

$$S_T = \sum_{ij} (x_{ij} - \bar{x})^2$$

$$S_w = \sum_{ij} (x_{ij} - \bar{x}_i)^2$$

حال ثابت می‌کنیم $S_B = S_T - S_w$

$$(x_{ij} - \bar{x}) = (\bar{x}_i - \bar{x}) + (x_{ij} - \bar{x}_i)$$

$$(x_{ij} - \bar{x})^2 = (\bar{x}_i - \bar{x})^2 + (x_{ij} - \bar{x}_i)^2 + 2(x_{ij} - \bar{x}_i)(\bar{x}_i - \bar{x})$$

$$\underbrace{\sum_{ij} (x_{ij} - \bar{x})^2}_{S_T} = \sum_{ij} (\bar{x}_i - \bar{x})^2 + \underbrace{\sum_{ij} (x_{ij} - \bar{x}_i)^2}_{S_w} + 2\sum_i \left[(\bar{x}_i - \bar{x}) \sum_j (x_{ij} - \bar{x}_i) \right]$$

$$S_T - S_w = \sum_{ij} (\bar{x}_i - \bar{x})^2 + 2\sum_i \left[(\bar{x}_i - \bar{x}) \sum_j (x_{ij} - \bar{x}_i) \right]$$

$$= \sum_i n_i (\bar{x}_i - \bar{x})^2 = S_B$$

پس اثبات برای حالت چندکلاسه انجام شده. البته در اینجا x تک فیچر در نظر گرفته شده اما برای حالت جامع باید جای تمام x^2 ها $x x^T$ قرار دهیم.

۲

b-

رنک ماتریس S_B حداکثر $k-1$ است. (k تعداد کلاس‌ها است)

$$S_B = \sum_{i=1}^{k} n_i (\mu_i - \mu)(\mu_i - \mu)^T$$

در رابطه بالا می‌بینیم که S_B مجموع k ماتریس با رنک ۱ است

(چون $rank(AA^T) = rank(A)$ که $rank(A)$ در اینجا یک است)

در واقع مجموع k ماتریس مرتبط از طریق μ با رنک یک می‌باشد

پس رنک ماتریس S_B حداکثر $k-1$ است.

↓ اثبات دقیق‌تر

$S_b \rightarrow$ sum of C rank_one matrices so: $rank(S_b) \leq C$

The C terms are dependant so in fact: $rank(S_b) \leq C-1$

↓

(via μ)

Algorithm: EigenFaces

Input: Face sets probe, gallery, training

Output: Displays each face in probe along with its best matching face in gallery

---

Steps

```
Setup (gallery, training)
for each Γ ∈ probe
    match = Recognize(Γ)
    print Γ, match[1][1]
```

7

Algorithm : Setup (gallery, training)

Input : gallery, training

output : facespace (2D)

Steps

let $M = |training|$

Ψ = Average (training)

for each Γ_i in training

$\Phi_i = \Gamma_i - \Psi$

let $A = [\Phi_1 \dots \Phi_M]$

let $C = A^T A$

$L[i] = \lambda_i$ such that $|C - \lambda_i I| = 0$ for $i = 1 \dots M$

$V[i] = v_i$ such that $Cv_i = \lambda_i v_i$ for $i = 1 \dots M$

for $i = 1$ to $|gallery|$ do :

let $\Gamma \in$ gallery

gallery = gallery $- \{\Gamma\}$

facespace[i][1] = Γ

facespace[i][2] = Project (Γ)

return facespace

8.

Algo: Project

Input: Face Image Γ

Output: face image

Steps

$\Omega[i] = V[i]^T [\Gamma - \Psi]$ for $i = 1 \dots M$

return Ω

Algo: Recognize (Γ)

Input: Face Image Γ

Output: a copy of facespace (closest to Γ's projected face)

Steps

O = project(Γ)

let match = facespace

Sort match by row based on dist(O, match[i][2])

retur match

* برای تشخیص اینکه چهره هست یا نه ← در recognize باید وقتی Sort
شد میزان شباهت تشخیص داد و اگر میزان شباهت از
حدی کمتر بود ← face نیست