ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

PTTKDLNB

CLASSIFICATION

Report to:

PGS.TS Lý Quốc Ngọc

LỚP CQ2016/2

1612221 Nguyễn Y Hợp

1612239 Hồ Thịnh Hưng

1512594 Trương Thanh Triết

Tp. Hồ Chí Minh, ngày 20 tháng 06 năm 2019

Nội dung

A- LÝ THUYẾT

1.Giới thiệu

Trong qua trình phân tích thống kê dữ liệu nhiều biến, đặc biệt trong giai đoạn hiện nay, dữ liệu ngày càng nhiều, ta cần có nhu cầu phân lớp dữ liệu để biế được sự khác biệt dữ liệu nhằm dữ đoán chính xác đặc trưng của dữ liệu. Đặc biệt trong Computer Vision, Marching learning, đây là bài toán quan trong được khai thác mạnh trong những năm gần đây. Cùng với sự phát triển của phần cứng, sự ra đời của deep Learning, mạng neural ra đời, phân lớp dữ liệu có tầm ảnh hường ngày càng quan trong. Vậy Classification là gì?

2.Định nghĩa

Classification là một kĩ thuật đa biến nhằm mục phân tích các đối tượng, dữ liệu (observation) và sau đó phân bố các đối tượng mới (observation) cho các class đã được xác định trước đó

Mục đích: Để sắp xếp các đối tượng (observation) vào 2 hay nhiều lớp đã được dán nhãn. Trọng tâm của vấn đề này là tạo ra một phương pháp tối ưu để dán nhãn đối tượng cho các lớp đã được dán nhãn

3.Phân tích bài toán phân lớp 2 population

3.1 Separation and Classification for Two Populations

Trong phần này chúng ta nhiệm vụ phân tích bài toán phân 2 lớp, nhiệm vụ xác định sao cho với một dữ liệu quan sát được là **x** thì sẽ được phân vào class nào.

Phát biểu bài toán

Đây là bài toánSupervised Learning (**Học có giám sát**) với dữ liệu đã được phân chia class sẵn

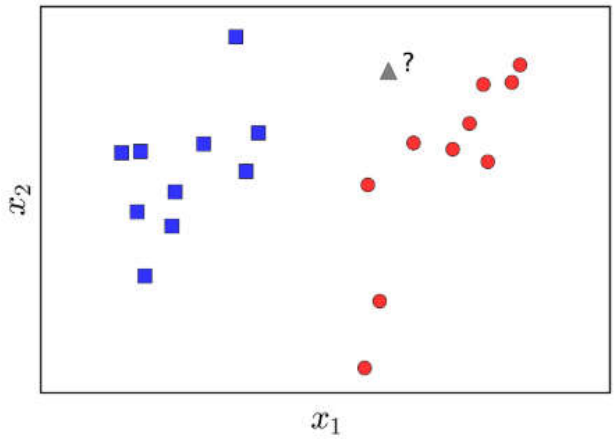
Population R_1 có label π_1, hàm mật độ xác suất $f_1(x)$, prior probability p_1

Population R_2 có label π_2, hàm mật độ xác suất $f_2(x)$ prior probability p_2

Input:

Cho trước vector $\mathbf{X}^T=[X_1, X_2, X_3 \dots X_p]$

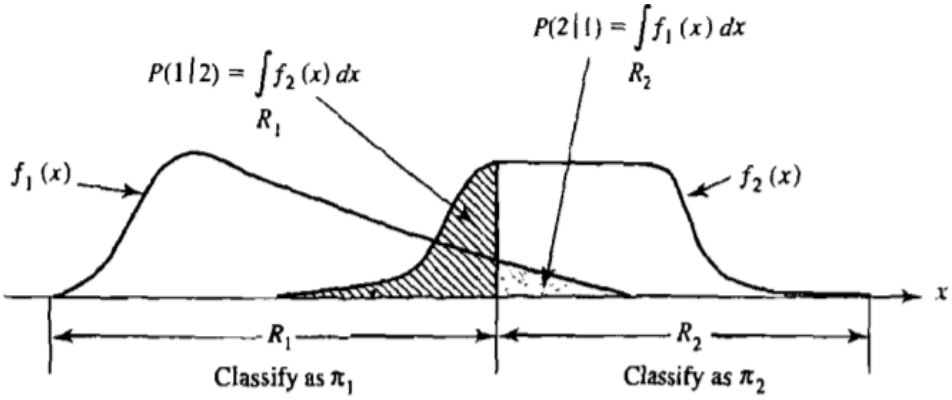
Output: X được dán nhãn label π_1 hoặc π_2



Hình 1: Mô tả bài toán phân 2 lớp

Trong Hình 1, ta nhận thấy 2 population đã được phân lớp thành màu đỏ và xanh. Bây giờ nếu có đối tượng"tam giác xám" thì đối tượng đó được phân lớp như thế nào?

Bây giờ chúng ta xét số chiều dữ liệu với p=2 khi đó ta có được các thông tin một cách trực quan sau :



Hình 2: Mô tả trực quan kí hiệu

Với 2 population thì ta có được $R_1 \cup R_2 = \Omega$

***P*(2|1) :**xác suất có điều kiên, phân lớp 1 đối tươngx vào π_2 trong khi nó thuộc về π_1,

$$P(2|1) = \mathrm{P}(\mathrm{X} \in R_2|\pi_1) = \int_{R_2} f_1(\mathbf{x})d\mathbf{x}$$

***P*(1|2) :**xác suất có điều kiên, phân lớp 1 đối tương x vào π_1 trong khi nó thuộc về π_2,

$$P(1|2) = \mathrm{P}(\mathrm{X} \in R_1|\pi_2) = \int_{R_1} f_2(\mathbf{x})d\mathbf{x}$$

Khi đó ta có được các xác suất có điều kiện sau:

P(Phân lớp sai X vào π_1)= P(X $\in R_1|\pi_2$) $P(\pi_2)$=$P(1|2)p_2$

P(Phân lớp sai X vào π_2)= P(X $\in R_2|\pi_1$) $P(\pi_1)$=$P(2|1)\ p_1$

Gọi c(i|j) là chi phí phân lớp sai đối tượng tử i vào j với i, j=1,2

Ta có thể biểu diễn chi phi phân lớp sai bằng vào cost matrix dưới đây

		Classify as	
		π_1	π_2
True population	π_1	0	c(2|1)
	π_2	c(1|2)	0

Để phân lớp đối tương một cách hiệu quả, ta đề xuất một hàm tính độ lỗi sao cho, khi phân lớp một đối tượng **x** thì độ lỗi sẽ nhỏ nhất. Hàm độ nỗi này được gọi là **ECM-excepted cost of misclassification**

$$ECM = c(2|1)P(2|1)p_1 + c(1|2)P(1|2)p_2$$

Mục tiêu của chúng ta là làm cực tiểu hàm này để việc phân lớp đạt hiểu quả tốt nhất. Dưới đây tôi sẽ trình bày điều kiện nào để cực tiểu hóa ECM:

Ta có:

$$P(2|1) = \text{P}(\text{X} \in R_2|\pi_1) = \int_{R_2} f_1(\mathbf{x})d\mathbf{x}$$

$$P(1|2) = \text{P}(\text{X} \in R_1|\pi_2) = \int_{R_1} f_2(\mathbf{x})d\mathbf{x}$$

Khi đó ECM tương đương như sau:

$$ECM = c(2|1)p_1 \int_{R_2} f_1(\mathbf{x})d\mathbf{x} + c(1|2)p_2 \int_{R_1} f_2(\mathbf{x})d\mathbf{x}$$

Mặt khác $R_1 \cup R_2 = \Omega$

$$\int_{R_1} f_1(\mathbf{x})d\mathbf{x} + \int_{R_2} f_1(\mathbf{x})d\mathbf{x} = \int_{\Omega} f_1(\mathbf{x})d\mathbf{x} = 1$$

ECM có thể viết lại:

$$ECM = c(2|1)p_1 \left[1 - \int_{R_1} f_1(\mathbf{x})d\mathbf{x}\right] + c(1|2)p_2 \int_{R_1} f_2(\mathbf{x})d\mathbf{x}$$

$$ECM = \int_{R_1} [c(1|2)p_2 f_2(\mathbf{x}) - c(2|1)p_1 f_1(\mathbf{x})]d\mathbf{x} + c(2|1)p_1$$

Ta nhận thấy c(1|2), c(2|1), p_1, p_2 không âm. Mặt khác $f_1(\mathbf{x})$, $f_2(\mathbf{x})$ không âm với mọi giá trị của **x.** Vậy để ECM minimum thì tích phân trong R_1 phải âm**:**

$$\int_{R_1} [c(1|2)p_2 f_2(\mathbf{x}) - c(2|1)p_1 f_1(\mathbf{x})]d\mathbf{x} \leq 0$$

Suy ra:

$$c(1|2)p_2 f_2(\mathbf{x}) - c(2|1)p_1 f_1(\mathbf{x}) \leq 0$$

Ta được kết luận sau:

Giá trị x thuộc về R_1 và được phân lớp π_1 khi:

$$R_1 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{c(1|2)}{c(2|1)} \frac{p_2}{p_1}$$

Giá trị x thuộc về R_2và được phân lớpπ_2 khi

$$R_2 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} < \frac{c(1|2)}{c(2|1)} \frac{p_2}{p_1}$$

Đây là một kết quả quan trong để áp dụng cho các bài phân tích sau này trong bài toán phân lớp. Từ kết quả trên ta có một số trường hợp đặc biệt sau:

Với $\mathbf{p_1/p_2=1}$

$$R_1 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{c(1|2)}{c(2|1)} R_2 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} < \frac{c(1|2)}{c(2|1)}$$

Với **c(1|2)/c(2|1)=1**

$$R_1 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{p_2}{p_1} \quad R_2 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} < \frac{p_2}{p_1}$$

Với $\mathbf{p_2/p_1=c(1|2)/c(2|1)=1}$ hoặc $\mathbf{p_2/p_1=1/(c(1|2)/c(2|1))}$

$$R_1 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq 1 \quad R_2 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} < 1$$

3.2 Classification With Two Multivariate Normal Population

Bây giờ chúng ta giả định có 2 hàm mật độ xác suất chuẩn $f_1(\mathbf{x})$ và $f_2(\mathbf{x})$. Hàm đầu tiện có mean vector $\boldsymbol{\mu_1}$ và covariance matrix $\boldsymbol{\Sigma_1}$, hàm thứ 2 có mean vector $\boldsymbol{\mu_2}$ và covariance matrix $\boldsymbol{\Sigma_2}$. Trong trường hợp đặc biệt thì covariance matrix bằng nhau. Sau đây chúng ta xét trường hợp đó:

3.2.1 Classification of Normal Populations When $\Sigma_1=\Sigma_2=\Sigma$

Goi hàm mật độ xác suất của biến $\mathbf{X}^{\mathbf{T}}=[X_1,X_2,X_3\ldots X_p]$ của 2 population $\boldsymbol{\pi_1}$ và $\boldsymbol{\pi_2}$là:

$$f_i(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}}\, exp\left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_i})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_i})\right] \; với\; i = 1,2$$

Với những tham số$\boldsymbol{\Sigma}$, $\boldsymbol{\mu_1}$, $\boldsymbol{\mu_2}$đã biết **và** $(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}$ là hằng số nênđễ dàng có được:

$$\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} = \frac{exp\left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_1})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_1})\right]}{exp\left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_2})\right]}$$
$$= exp\left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_1})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_1}) + \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_2})\right]$$

Mặt khác

$$\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_1})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_2}) + \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_2})$$
$$= (\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}x - \frac{1}{2}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_1}+\boldsymbol{\mu_2})$$

Để cực tiểu hóa ECM trong từng vùng R_1, R_2 thì:

$$R_1: \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}$$

$$R_2: \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} < \frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}$$

Từ đó ta có được:

$$R_1: (\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}x - \frac{1}{2}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_1}+\boldsymbol{\mu_2}) \geq ln\left[\frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}\right]$$

$$R_2: (\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}x - \frac{1}{2}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_1}+\boldsymbol{\mu_2}) < ln\left[\frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}\right]$$

Trong thực tế các giá trị $\boldsymbol{\mu_1}, \boldsymbol{\mu_2}, \boldsymbol{\Sigma}$không xác định. Tuy nhiên nếu có được n_1 biến $\mathbf{X}$ với $\mathbf{X}^T=[X_1, X_2, X_3,\ldots,X_p]$, p là số chiều của dữ liệu có phân lớp$\boldsymbol{\pi_1}$và n_2 biến X có phân lớp $\boldsymbol{\pi_2}$. Với điều kiên $n_1+n_2-2 \geq p$. Ta có matrix data sau:

$$\mathbf{X_1} = \begin{bmatrix} \mathbf{x_{11}^T} \\ \mathbf{x_{12}^T} \\ \vdots \\ \mathbf{x_{1n_1}^T} \end{bmatrix} \text{ có thước } (n_1 \text{ x p})$$

$$\mathbf{X_2} = \begin{bmatrix} \mathbf{x_{21}^T} \\ \mathbf{x_{22}^T} \\ \vdots \\ \mathbf{x_{2n_2}^T} \end{bmatrix} \text{ có thước } (n_2 \text{ x p})$$

Vậy ta có thể tính được sample mean vector $\overline{\mathbf{x_1}}, \overline{\mathbf{x_2}}$và covariance matrix $\mathbf{S_1}, \mathbf{S_2}$ của $\mathbf{X_1}, \mathbf{X_2}$

Do 2 population có covariance matrix tương đồng nhau nên ta có thể tính$\mathbf{S_{pooled}}$

$$\boldsymbol{S_{pooled}} = \frac{\boldsymbol{n_1 - 1}}{\boldsymbol{n_1 + n_2 - 2}}\boldsymbol{S_1} + \frac{\boldsymbol{n_2 - 1}}{\boldsymbol{n_1 + n_2 - 2}}\boldsymbol{S_2}$$

Khi đó ta có kết quả sau:

$$R_1: (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x} - \frac{1}{2}(\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}(\overline{\mathbf{x_1}} + \overline{\mathbf{x_2}}) \geq ln\left[\frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}\right]$$

$$R_2: (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x} - \frac{1}{2}(\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}(\overline{\mathbf{x_1}} + \overline{\mathbf{x_2}}) < ln\left[\frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}\right]$$

Ta có nhân xét sau:

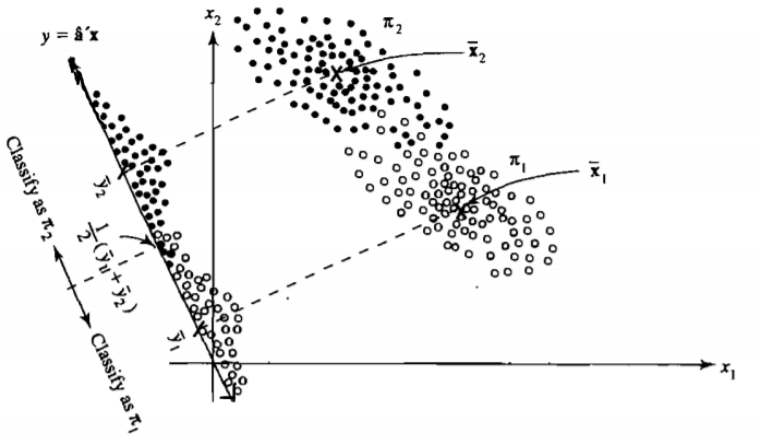
$$\hat{y} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x} = \hat{\mathbf{a}}^T\mathbf{x} \text{ (Tổ hợp tuyến tính)}$$

$$\hat{\boldsymbol{m}} = \frac{1}{2}(\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}(\overline{\mathbf{x_1}} + \overline{\mathbf{x_2}}) = \frac{1}{2}(\overline{\mathbf{y_1}} + \overline{\mathbf{y_2}})$$

Trong đó

$$\overline{\mathbf{y_1}} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\overline{\mathbf{x_1}} = \hat{\mathbf{a}}^T\overline{\mathbf{x_1}}$$

$$\overline{\mathbf{y_2}} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\overline{\mathbf{x_1}} = \hat{\mathbf{a}}^T\overline{\mathbf{x_2}}$$



Xét trường hợp đặc biệt:

$$ln = \left(\frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}\right) = 0$$

Ta có thể nhận thấy đẻ x_0 được phân lớp vào π_1:

$$y = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x_0} = \hat{\mathbf{a}}^T\mathbf{x_0} \geq \hat{\boldsymbol{m}}$$

Và được phân lớp vàoπ_2 trong trường hợp ngược lại.

3.2.2 Fisher's Approach To Classification with Two Population

Phân này chủ yếu tập trung vào tính chất được ứng dụng phân lớp đối tượng:

$$\hat{y} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x} = \hat{\mathbf{a}}^T\mathbf{x}$$

Giả sử y_{11}, y_{12}, y_{13},…,y_{1n_1} là tổ hớp tuyến tính của của các giá trị **x**thuộc populationπ_1 và y_{21}, y_{22}, y_{23},…,y_{2n_2} tổ hớp tuyến tính của của các giá trị **x**thuộc populationπ_2

Khi đó ta định nghĩa sự tách biệt của 2 tập dữ liệu chính là sự khác biệt giữa $\overline{y_1}$,$\overline{y_2}$, được thể hiện bằng đơn vị độ lệch chuẩn. Điều đó được biểu diễn bằng công thức sau

$$seperation = \frac{|\overline{y_1} - \overline{y_2}|}{S_y}$$

Tròn đó:

$$S_y^2 = \frac{\sum_{j=1}^{n_1}(y_{1j} - \overline{y_1})^2 + \sum_{j=1}^{n_2}(y_{2j} - \overline{y_2})^2}{n_1 + n_2 - 2}$$

Vậy để $\boldsymbol{seperation}$đạt cực đại thì

$$seperation = \frac{|\overline{y_1} - \overline{y_2}|}{S_y} = \frac{(\overline{y_1} - \overline{y_2})^2}{{S_y}^2} = \frac{(\hat{\mathbf{a}}^T\overline{\mathbf{x_1}} - \hat{\mathbf{a}}^T\overline{\mathbf{x_2}})^2}{\hat{\mathbf{a}}^T\boldsymbol{S_{pooled}}\hat{\mathbf{a}}} = \frac{(\hat{\mathbf{a}}^T\mathbf{d})^2}{\hat{\mathbf{a}}^T\boldsymbol{S_{pooled}}\hat{\mathbf{a}}}$$

với d=$(\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})$

Một tính chất tường đượng sử dụng đã được chứng minh. Để $seperation$ thì

$$\boldsymbol{D^2} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T {\boldsymbol{S_{pooled}}}^{-1} (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})$$

Kết quả này được ứng đụng để phân lớp những quan sát mới

3.2.3 Classification of Normal When Σ_1!=Σ_2

Trong phần này chúng ta xét giả thuyết như **3.2.1** tuy nhiên$\boldsymbol{\Sigma_2}$!=$\boldsymbol{\Sigma_2}$.

Hàm mật độ xác suất$f_i(\mathbf{x})$ của $\boldsymbol{\pi_i}$với i=1,2 như sau:

$$f_i(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma_i}|^{1/2}} exp\left[\frac{-1}{2}(\mathbf{x} - \boldsymbol{\mu_i})^T {\boldsymbol{\Sigma_i}}^{-1}(\boldsymbol{x} - \boldsymbol{\mu_i}\right] \; với \; i = 1,2$$

Áp dụng với kết quả sau

$$R_1 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}$$

$$R_2 : \frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} < \frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}$$

Ta được kết quả:

$$R_1 : \frac{-1}{2}x^T(\boldsymbol{\Sigma_1^{-1}} - \boldsymbol{\Sigma_2^{-1}})\mathbf{x} + (\boldsymbol{\mu_1}^T\boldsymbol{\Sigma_1^{-1}} - \boldsymbol{\mu_2}^T\boldsymbol{\Sigma_2^{-1}})\,\mathbf{x} - k \geq \frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}$$

$$R_2 : \frac{-1}{2}x^T(\boldsymbol{\Sigma_1^{-1}} - \boldsymbol{\Sigma_2^{-1}})\mathbf{x} + (\boldsymbol{\mu_1}^T\boldsymbol{\Sigma_1^{-1}} - \boldsymbol{\mu_2}^T\boldsymbol{\Sigma_2^{-1}})\,\mathbf{x} - k < \frac{c(1|2)}{c(2|1)}\frac{p_2}{p_1}$$

Trong đó

$$k = \frac{1}{2}\,ln\left(\frac{|\boldsymbol{\Sigma_1}|}{|\boldsymbol{\Sigma_2}|}\right) + \frac{1}{2}(\boldsymbol{\mu_1}^T\boldsymbol{\Sigma_1^{-1}}\boldsymbol{\mu_1} - \boldsymbol{\mu_2}^T\boldsymbol{\Sigma_2^{-1}}\boldsymbol{\mu_2})$$

Vậy với một giá trị $\mathbf{x_0}$**nếu thỏa điều kiên** $\mathbf{R_1}$ **thì được phân lớp vào** $\boldsymbol{\pi_1}$ và phân lớp$\boldsymbol{\pi_2}$trong trường hợp ngược lại

Đến đây là kết thúc lý thuyết về phân phân lớp cho 2 population. Tiếp theo chúng ta tìm hiểu về phần phân lớp cho g populations với $g \geq 2$.

4. Đánh giá quy tắc phân loại

Để đánh giá hiệu suất của quy trình phân loại mẫu, ta cần tính xác suất hoặc tỉ lệ phân loại sai của nó

Một thước đo hiệu suất có thể được tính cho bất kỳ quy trình phân loại nào là**the apparent error rate**(APER)

Ta định nghĩa n_{1M} và n_{2M} là số lượng đối tượng bị phân lỗi sai trên $\boldsymbol{\pi_1}$ **và** $\boldsymbol{\pi_2}$khi đó độ lỗi được xác định như sau:

$$APER = \frac{n_{1M} + n_{2M}}{n_1 + n_2}$$

(APER) tình toán dễ dàng một cách trực quan. Tuy nhiên nó có xu hướng đánh giá thấp trong thực tế khi phân loại đối tượng mới. Vì vậy người ta đề xuấtAER.

Ý tưởng là xem xét một mẫu thử nghiệm độc lập và các đối tượng mới mà ta biết nhãn lớp thực sự. Điều này có nghĩa là ta chia mẫu ban đầu trong một mẫu đào tạo và mẫu thử nghiệm

AER sau đó được ước tính bằng tỷ lệ của các đối tượng phân loại sai trong mẫu thử trong khi mẫu đào tạo được sử dụng để xây dựng quy tắc phân loại. Tuy nhiên, có hai nhược điểm với phương pháp này:

- Nó đòi hỏi mẫu lớn.
- Quy tắc phân loại ít chính xác hơn vì chúng tôi không sử dụng thông tin từ mẫu thử để xây dựng bộ phân loại.

Quy trình thực hiện như sau:

- Để một đối tượng ra khỏi mẫu và xây dựng quy tắc phân loại dựa trên n - 1 đối tượng còn lại trong mẫu.
- Phân loại quan sát được lấy ra bằng cách sử dụng quy tắc phân loại được xây dựng ở bước 1
- Lặp lại hai bước trước cho từng đối tượng trong mẫu. Gọi n_{1M}^{CV}, n_{2M}^{CV}số lượng các quan sát được phân loại sai trong lớp 1 và 2 tương ứng.

$$\widehat{\boldsymbol{AER}} = \frac{n_{1M}^{CV} + n_{2M}^{CV}}{n_1 + n_2}$$

5. Phân tích bài toán Classification with several population

5.1 Classification with several population

Chương này trình bày về phân lớp cho g population với g ≥ 2.Trước hết ta cần định nghĩa kí hiệu cũng như ý nghĩa của kí hiệu đó được dùng trong chương này

Gọi $f_i(x)$ là hàm mật độ xác suất của population $\boldsymbol{\pi_i}$, i=1,2,3,…,g. $f_i(x)$là hàm mật độ phân chuẩn ta có các giải thuyết sau:

- p_i : prior probability của population $\boldsymbol{\pi_i}$, i=1,2,3,…,g
- c(k,i): chi phi cho việc phân lớp 1 item vào $\boldsymbol{\pi_k}$, Trong khi nó thuộc về$\boldsymbol{\pi_i}$, *với* i,k=1,2,3,…,g. Với k=i, c(k|i)=0
- Với các giá trị x thuộc về R_k được phân lớp vào $\boldsymbol{\pi_k}$ **và ta có:**

P(k|i) = P(Phân lớp item vào lớp $\boldsymbol{\pi_k}|\boldsymbol{\pi_i}$)=$\int_{R_k} f_i(\mathbf{x})d\mathbf{x}$ với i,k=1,2,3,…,g và

$P(i|i) = 1 - \sum_{i=1,k!=i}^{g} P(k|i)$

Để phân lớp đối tượng, ta cần một độ lỗi sao cho khi phân lớp đối tượng **x** sao cho độ lỗi này nhỏ nhất. Trong chương này, ta dùng **ECM**đã được dùng ở chương **3.1**

Xét chi phi phân lớp sai có điều kiên một giá trị **x**từ$\boldsymbol{\pi_1}$ **vào** $\boldsymbol{\pi_2, \pi_3, \pi_4 \ldots \pi_g}$là:

$$ECM(1)=c(2|1)P(2|1)+c(3|1)P(3|1)+\ldots+c(g|1)P(g|1)=\sum_{k=2}^{g} P(k|1)c(k|1)$$

Tương tự ta có được kì vọng chi phí phân lớp sai ECM(2), ECM(3), ECM(4),…,ECM(g).

Với mỗi ECM(i) ta có được prior probability p_i, nên ta có được ECM tổng thể:

$$ECM=p_1 ECM(1)+ p_2\, ECM(2) + p_3\, ECM(3) +\ldots+ p_g\, ECM(g)$$

$$= p_1\left(\sum_{k=2}^{g} P(k|1)c(k|1)\right) + p_2\left(\sum_{k=1,k!=2}^{g} P(k|2)c(k|2)\right) + \cdots$$

$$+ p_g\left(\sum_{k=1}^{g-1} P(k|g)c(k|g)\right) = \sum_{k=1}^{g} p_1\left(\sum_{k=1,k!=i}^{g-1} P(k|i)c(k|i)\right)$$

Để cực tiểu hóa ECM thì cần phải phân lớp x vào population $\boldsymbol{\pi_k}$với k=1,2,3,…g thì

$\sum_{i=1,i!=k}^{g} p_i\, f_i(x)\, c(k|i)$ nhỏ nhất

Giả sử, chi phí phân lớp sai c(k|i) như nhau cho tất cả các population khi đó ta được: $\sum_{i=1,i!=k}^{g} p_i\, f_i(x)$ nhỏ nhất để ECM được cực tiểu hóa

Tuy nhiên nếu chi phi phân lớp sai c(k|i) như nhau, khi đó ta mục tiêu để tối ưu quá trình phân lớp chỉ còn phụ thuộc vào p_i , $f_i(x)$. Vậy bậy giờ ta xem $\boldsymbol{p_k\, f_k(\mathbf{x})}$ như một "kì vọng" để x được phân vào $\boldsymbol{\pi_k}$**.** Từ đây mục tiêu của ta sẽ dễ dàng hơn nhiều, đó là làm sao cho "kì vọng" lớn nhấttrong tất cả các kì vọng để phân vào lớp $\boldsymbol{\pi_k}$

Do đó để phân lớp x_0 vào lớp $\boldsymbol{\pi_k}$ **thì:**

$$\boldsymbol{p_k f_k(\mathbf{x_0}) > p_i f_i(\mathbf{x_0})\ với\ i}\ i! = k$$

Tương đượng với

$$\boldsymbol{ln(\ p_k f_k(\mathbf{x_0})) > ln(\ p_i f_i(\mathbf{x_0})\ ) với\ i}\ i! = k$$

5.2 Classification with Normal Populations

$$f_i(x) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma_i}|^{1/2}} exp\left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_i})^T \boldsymbol{\Sigma_i}^{-1}(\mathbf{x}-\boldsymbol{\mu_i})\right]\ với\ i = 1,2,3\ldots,g$$

Vậy để phân lớp x_0 vào lớp $\boldsymbol{\pi_k}$thì:

$$ln(\ p_k f_k(\mathrm{x})) = \boldsymbol{ln}(\ p_k) - \frac{p}{2}ln(2\pi) - \frac{1}{2}ln(\boldsymbol{\Sigma_k}) - \left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_k})^T \boldsymbol{\Sigma_k}^{-1}(\mathbf{x}-\boldsymbol{\mu_k})\right]\ lớn\ nhất$$

Do $\frac{p}{2}ln(2\pi)$ là hằng số nên

$$ln(\ p_k f_k(\mathrm{x})) = -\frac{1}{2}ln(\boldsymbol{\Sigma_k}) - \left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_k})^T \boldsymbol{\Sigma_k}^{-1}(\mathbf{x}-\boldsymbol{\mu_k})\right] + \boldsymbol{ln}(\ p_k)\ lớn\ nhất$$

$$max\left(-\frac{1}{2}ln(\boldsymbol{\Sigma_i}) - \left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_i})^T \boldsymbol{\Sigma_i}^{-1}(\mathbf{x}-\boldsymbol{\mu_i})\right] + \boldsymbol{ln}(\ p_i)\right)\ với\ i = 1,2,3\ldots,g$$

Đặt:

$$\boldsymbol{d_i^Q(\mathbf{x})} = -\frac{1}{2}ln(\boldsymbol{\Sigma_i}) - \left[\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu_i})^T \boldsymbol{\Sigma_i}^{-1}(\mathbf{x}-\boldsymbol{\mu_i})\right] + \boldsymbol{ln}(\ p_i)$$

$\boldsymbol{d_i^Q(\mathbf{x})}$bao gồm sự tham gia của phương sai tổng quát$|\boldsymbol{\Sigma_i}|$**,** prior probability và khoảng cách từ **x** đến trưng bình mẫu$\boldsymbol{\mu_i}$**.** Chúng ta có thể dùngdiscrimination score trong bài toán phân lớp. Khi đó x_0 được phân lớp vào $\boldsymbol{\pi_k}$nếu **:**

$$\boldsymbol{the\ quadratic\ score\ d_k^Q(x_0) = largest\ trong\ d_1^Q(x_0), d_2^Q(x_0), d_3^Q(x_0)\ldots, d_g^Q(x_0)}$$

Xét trường hớp $\boldsymbol{\mu_i}$ **và** $\boldsymbol{\Sigma_i}$chưa biết. Tuy nhiên ta có tập traning set $\mathbf{X_i}$tương ứng với populations$\boldsymbol{\pi_i}$với i=1,2,3,..g. Khi đó ta có được các thông tinh sau:

$$\overline{\mathbf{x_i}} = \boldsymbol{sample\ mean\ vector}$$

$$\boldsymbol{S_i} = \boldsymbol{sample\ covariance\ matrix}$$

Với

$$\boldsymbol{n_i} = \boldsymbol{kích\ thước\ sample}$$

Xét trường hợp $\boldsymbol{\Sigma_i} = \boldsymbol{\Sigma}$

$$\boldsymbol{d_i^Q}(\mathbf{x}) = -\frac{1}{2} ln(\boldsymbol{\Sigma}) - \frac{1}{2}\boldsymbol{x^T\Sigma^{-1}x} + \boldsymbol{\mu_i}^T\boldsymbol{\Sigma^{-1}x} - \frac{1}{2}\boldsymbol{\mu_i}^T\boldsymbol{\Sigma^{-1}\mu_i} + \boldsymbol{ln}(\,p_i) \text{ với } i = 1,2,3,\ldots,g$$

Nhận thấy $-\frac{1}{2} ln(\boldsymbol{\Sigma}) - \frac{1}{2}\boldsymbol{x^T\Sigma^{-1}x}$ **giống nhau** $\boldsymbol{trong\ d_1^Q}(\mathbf{x}), \boldsymbol{d_2^Q}(\boldsymbol{x}), \boldsymbol{d_3^Q}(\boldsymbol{x})\ldots, \boldsymbol{d_g^Q}(\boldsymbol{x})$**.** Khi đó ta định nghĩa linear discriminant score là:

$$d_i(x) = \boldsymbol{\mu_i}^T\boldsymbol{\Sigma^{-1}x} - \frac{1}{2}\boldsymbol{\mu_i}^T\boldsymbol{\Sigma^{-1}\mu_i} + \boldsymbol{ln}(\,p_i)$$

Tương ứng với $\boldsymbol{\mu_i}$ **và** $\boldsymbol{\Sigma_i}$chưa biết thì ta được:

$$\widehat{d_i}(x) = \overline{\mathbf{x_i}}^T\boldsymbol{S_{pooled}}^{-1}\mathbf{x} - \frac{1}{2}\overline{\mathbf{x_i}}^T\boldsymbol{S_{pooled}}^{-1}\overline{\mathbf{x_i}} + \boldsymbol{ln}(\,p_i)$$

Với *i*=1,2,3,…,g

Trong đó

$$\boldsymbol{S_{pooled}} = \frac{1}{n_1 + n_2 + \ldots + n_g - g}((n_1 - 1)S_1 + (n_2 - 1)S_2 + \ldots + (n_g - 1)S_g)$$

Vậy để $\mathbf{x_0}$ được phân lớp vào $\boldsymbol{\pi_k}$ **thì :**

the linear discriminant $\widehat{d_k}(x_0)$ = lớn nhất trong $\widehat{d_1}(x_0), \widehat{d_2}(x_0), \ldots, \widehat{d_g}(x_0)$

6. Linear Discriminant Analysis LDA

6.1 Phát biểu bài toán

Bài toán đặt mục tiêu làm cho classes được gọi là *discriminative* nếu các class đó cách xa nhau (between-class variance lớn) và dữ liệu trong mỗi class có xu hướng giống nhau (within-class variance nhỏ). Linear Discriminant Analysis là thuật toán đi tìm một phép chiếu sao cho tỉ lệ giữa *between-class variance* và *within-class variance* lớn nhất có thể.

6.2 Phân tích bài toán LDA cho 2 class

Trong phần này xét 2 class

Giả sử ta có 2 class quan sát được gọi là C_1 và C_2 với tổng số phần tử là N có kì vong là μ_1 và μ_2

Số chiều dữ liệu đang xét là p chiều. C_1 có N_1 phần từ được biểu diễn bằng matrix X_1 với kích thước (N_1 x p). C_2 có N_2 phần tử được biểu diễn bằng matrix với kích thước (N_2 x p)

Phép chiếu dữ liệu xuống 1 đường thẳng có thể được mô tả bằng một vector hệ số **w**, giá trị tương ứng của mỗi điểm dữ liệu mới được cho bởi công thức sau:

$$y_n = w^T x_n \; với \; 1 \leq n \leq \; N$$

Gọi m_1 và m_2 là vector kì vọng của class C_1 và C_2 sau khi được chiếu xuống đường thẳng được mô tả bằng hệ số w

Khi đó kì vọng của tổ hợp tuyến tính của mỗi class được tính như sau

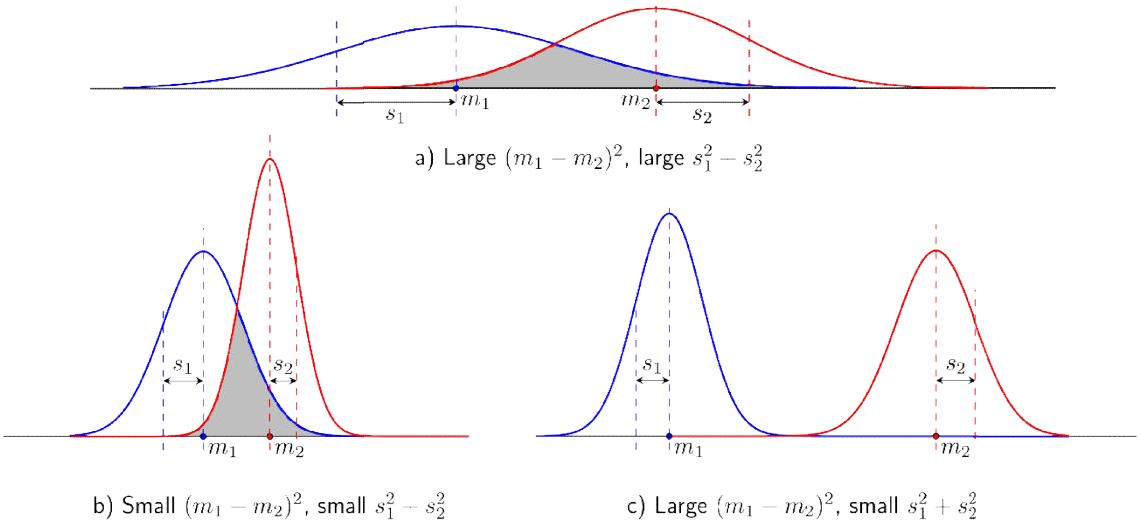
$$m_i = \frac{1}{N_i} \sum_{n=1}^{N_i} y_n = \boldsymbol{w^T \mu_i} \; với \; i = 1{,}2$$

Khi đó

$$m_1 - m_2 = \boldsymbol{w^T(\mu_1 - \mu_2)}$$

Ta định nghĩa Các **within-class variances** là

$$s_k^2 = \sum_{n \in C_K} (y_n - m_k)^2 \; với \; k = 1{,}2$$



Dưa theo hình trên ta thấy khoảng cách giữa các kỳ vọng và tổng các phương sai ảnh hưởng tới độ *discriminant* của dữ liệu. Nên mục tiêu của ta là:

tìm giá trị lớn nhất của hàm mục tiêu:

$$F_{fisher}(w) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$

Tiếp theo, chúng ta sẽ đi tìm biểu thức phụ thuộc giữa tử số và mẫu số trong vế phải của phương trình F_{fisher}

$$(m_1 - m_2)^2 = \boldsymbol{w^T(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T w = w^T S_B w}$$

Với:

$$\boldsymbol{S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T}$$

$\mathbf{S_B}$ còn được gọi là **between-class covariance matrix**. Đây là một ma trận đối xứng nửa xác định dương.

Mặt khác ta lại có:

$$s_1^2 + s_2^2 = \sum_{k=1}^{2} \sum_{n \in C_K} \boldsymbol{(w^T(x_n - \mu_k)^2 = w^T} \sum_{k=1}^{2} \sum_{n \in C_K} \boldsymbol{(x_n - \mu_k)(x_n - \mu_k)^T w}$$

Đặt S_w là **within-class covariance matrix:**

Khi đó:

$$s_1^2 + s_2^2 = \boldsymbol{w^T S_w w}$$

Vậy

$$F_{fisher}(w) = \frac{\boldsymbol{w^T S_B w}}{\boldsymbol{w^T S_w w}}$$

Lấy đạo hàm $F_{fisher}(w) = 0$ theo **w** ta được

$$\nabla_w F_{fisher}(w) = 0$$

$$\nabla_w \left(\frac{\boldsymbol{w^T S_B w}}{\boldsymbol{w^T S_w w}}\right) = 0$$

$$(\boldsymbol{w^T S_w w})(2\boldsymbol{S_B w}) - (\boldsymbol{w^T S_B w})(2\boldsymbol{S_w w}) = 0$$

$$\boldsymbol{S_B w} = \frac{\boldsymbol{w^T S_B w}}{\boldsymbol{w^T S_w w}}(\boldsymbol{S_w w})$$

$$\boldsymbol{S_w^{-1} S_B w} = F_{fisher}(w)\boldsymbol{w}$$

Vì $F_{fisher}(w)$ vô hướng nên ta suy ra w phải là một vector riêng của$\boldsymbol{S_w^{-1} S_B}$**.** Vậy, để hàm mục tiêu là lớn nhất thì$F_{fisher}(w)$chính là trị riêng lớn nhất của$\boldsymbol{S_w^{-1} S_B}$

Vậy ta có thể chọn **w** sao cho$(\boldsymbol{\mu_1 - \mu_2})(\boldsymbol{\mu_1 - \mu_2})^T \boldsymbol{w}$ =$F_{fisher}(w) = L$ là trị riêng lớn nhất của$\boldsymbol{S_w^{-1} S_B}$

Khi đó

$$\boldsymbol{Lw = S_w^{-1}(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T w = LS_w^{-1}(\mu_1 - \mu_2)}$$

Vậy

$$\boldsymbol{w = \alpha S_w^{-1}(\mu_1 - \mu_2)}$$

7. Phân tích bài toán LDA multi-class

7.1 Phân tích bài toán

Giả sử rằng số chiều của dữ liệu *D* lớn số lượng class*C*

Mục tiêu của ta mong muốn giảm số chiều về D'<D ứng với mỗi điểm dữ liệu **x**ta có**:**

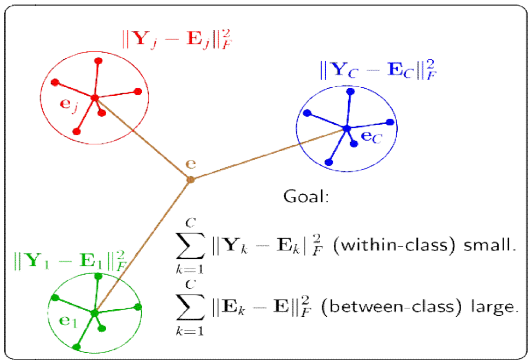
$$\boldsymbol{y = W^T x}$$

Với$\boldsymbol{W \in R^{DxD'}}$

Dưới đây là kí hiệu được dùng trong phần này:

- $\boldsymbol{X_k, Y_k}$lần lượt là matrix dữ liệu trong không gian ban đầu và trong không gian mới với số chiều nhỏ hơn với $\boldsymbol{Y_k = W^T X_k}$

- $\boldsymbol{m_k} = \frac{1}{N_k}\sum_{n\in C_k} \boldsymbol{x_n}$ là vector kì vọng của **class k** trong không gian ban đầu
- $\boldsymbol{e_k} = \frac{1}{N_k}\sum_{n\in C_k} \boldsymbol{y_n} = \boldsymbol{W^T m_k}$là kì vọng của class k trong không gian mới
- **m** là vector kỳ vọng của toàn bộ dữ liệu trong không gian ban đầu và **e** là vector kỳ vọng trong không gian mới.



Mục đích cũng là sự khác nhau giữa các thành phần trong 1 class (within-class) là nhỏ và sự khác nhau giữa các classes là lớn. Các điểm dữ liệu có màu khác nhau thể hiện các class khác nhau được miểu tả ở hình trên

7.2 Within-class nhỏ

Within class k có thể được tính như sau:

$$\sigma_k^2 = \sum_{n\in C_k} \left|\left|\boldsymbol{y_n} - \boldsymbol{e_k}\right|\right|^2 = \left|\left|\boldsymbol{Y_k} - \boldsymbol{E_k}\right|\right|^2 = \left|\left|\boldsymbol{W^T}(\boldsymbol{X_k} - \mathbf{M_k})\right|\right|^2$$

$$= trace[\boldsymbol{W^T}(\boldsymbol{X_k} - \mathbf{M_k})(\boldsymbol{X_k} - \mathbf{M_k})^T\boldsymbol{W}]$$

Trong đó$\boldsymbol{E_k}$ là một matrix có các cột giống hệt nhau và bằng với kì vọng$\boldsymbol{e_k}$trong không gian mới Ta có thể nhận thấy được $\boldsymbol{E_k} = \boldsymbol{W^T}\mathbf{M_k}$ với $\mathbf{M_k}$ là matrix có các cột giống như nhau bằng với vector kì vọng $\boldsymbol{m_k}$ trong không gian ban đầu

Vậy within-class trong multi-class LDA được tính bằng công thức sau:

$$s_w = \sum_{k=1}^{C} \sigma_k^2 = \sum_{k=1}^{C} trace[\boldsymbol{W^T}(\boldsymbol{X_k} - \mathbf{M_k})(\boldsymbol{X_k} - \mathbf{M_k})^T\boldsymbol{W}] = trace(\boldsymbol{W^T S_w W})$$

Trong đó:

$$\boldsymbol{S_w} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{C}} ||\boldsymbol{X_k} - \boldsymbol{M_k}||^2 = \sum_{\boldsymbol{k=1}}^{\boldsymbol{C}} \sum_{\boldsymbol{n\in C_k}} (\boldsymbol{x_n} - \boldsymbol{m_k})(\boldsymbol{x_n} - \boldsymbol{m_k})^T$$

Được gọi làwithin-class covariance matrix của multi-class LDA

7.3 Between class lớn

Để đạt được between class lớn thì tất cả những điểm trong không gian sẽ cách xa vector kì vonge chung của dữ liệu. Việc này cũng có thể đạt được nếu các vector kỳ vọng của mỗi class xa các vector kỳ vọng chung **e**(trong không gian mới

$$\boldsymbol{s_B = \sum_{k=1}^{C} N_k \,||\, e_k - e||^2 = \sum_{k=1}^{C} ||\, E_k - E||^2}$$

Với$\boldsymbol{N_k}$là trọng số vì những class sẽ có số lượng phần tử khác nhau

$$\boldsymbol{s_B = \sum_{k=1}^{C} trace(W^T(M_k - M)(M_k - M)^T W) = \; trace(\, W^T S_B W)}$$

Trong đó:

$$S_B = \sum_{k=1}^{C} (\boldsymbol{M_k - M})(\boldsymbol{M_k - M})^{\boldsymbol{T}} = \sum_{k=1}^{C} \boldsymbol{N_k (m_k - m)(m_k - m)^T}$$

7.4 Loss function

Với cách định nghĩa và ý tưởng về within-class nhỏ và between-class lớn như trên, ta có thể xây dựng bài toán tối ưu nhưu sau:

$$F_{fisher}(\boldsymbol{W}) = \frac{trace(\boldsymbol{W^T S_B W})}{trace(\boldsymbol{W^T S_w W})}$$

Tìm W sao cho $F_{fisher}(\boldsymbol{W})$ đã cực đại

Lấy đạo hàm $F_{fisher}(\boldsymbol{W}) = 0$ theo **W** ta được

$$\nabla_W F_{fisher}(W) = 0$$

$$\Leftrightarrow \frac{2(\boldsymbol{S_B W}\; trace(\boldsymbol{W^T S_B W}) - trace(\boldsymbol{W^T S_w W})\boldsymbol{S_w W})}{(trace(\boldsymbol{W^T S_w W}))^2} = 0$$

$$\Leftrightarrow (\boldsymbol{S_B W}\; trace(\boldsymbol{W^T S_B W}) - trace(\boldsymbol{W^T S_w W})\boldsymbol{S_w W}) = 0$$

$$\Leftrightarrow \boldsymbol{S_w^{-1} S_B W = JW}$$

Suy ra mỗi một cột của **W** là một vector riêng của $\boldsymbol{S_w^{-1} S_B}$ ứng với trị riêng lớn nhất của matrix này

8. So sánh LDA và PCA

LDA	PCA
Giảm số chiều của sữ liệu bằng cách mapping dữ liệu với số chiều lớn sang một không gian ít chiều hơn.	
Supervised	Unsupervised
Xây dựng không gian mới với số chiều nhỏ hơn ban đầu sao cho mỗi class phải đảm bảo tính chất **between class lớn,within class nhỏ**	Xây dựng không gian mới với số chiều nhỏ hơn ban đầu, các trục tọa độ trong không gian mới phải đảm bảo tính chất ***maximize the variability.***

9. Ứng dụng

- Face recognition
- Handwritten digit recognition
- Text mining
- Image retrieval
- Microarray data analysis
- Protein classification

B-THỰC HÀNH

1. Phân lớp 2 quần thể

Source: Data courtesy of K.A. Jensen and B.Van Alen of the State of Alaska Department of Fish and Game

Áp dụng tính chất để phân lớp

$$\hat{y} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x}=\hat{\mathbf{a}}^{\mathrm{T}}\mathbf{x} \text{ (Tổ hợp tuyến tính)}$$

$$\hat{\boldsymbol{m}} = \frac{1}{2}(\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}(\overline{\mathbf{x_1}} + \overline{\mathbf{x_2}}) = \frac{1}{2}(\overline{\mathbf{y_1}} + \overline{\mathbf{y_2}})$$

Trong đó

$$\overline{\mathbf{y_1}} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\overline{\mathbf{x_1}} = \hat{\mathbf{a}}^T\overline{\mathbf{x_1}}$$

$$\overline{\mathbf{y_2}} = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\overline{\mathbf{x_1}} = \hat{\mathbf{a}}^T\overline{\mathbf{x_2}}$$

Để x_0 được phân lớp vào π_1:

$$y = (\overline{\mathbf{x_1}} - \overline{\mathbf{x_2}})^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x_0}=\hat{\mathbf{a}}^T\mathbf{x_0} \geq \hat{\boldsymbol{m}}$$

Và được phân lớp vàoπ_2 trong trường hợp ngược lại.

Code

```
import numpy as np
import matplotlib.pyplot as plt
import pandas
```

```
# Load datat từ file SalmonData.xlsx
data=pandas.read_excel("SalmonData.xlsx")
X_1=np.vstack((data["Freshwater_Alask"],data["Marine_Alask"])).T
X_2=np.vstack((data["Freshwater_Canadian"],data["Marine_Canadian"])).T
n_1=X_1.shape[0]
n_2=X_2.shape[0]
p_1=X_1.shape[1]
p_2=X_2.shape[1]
```

```
# Tính sample mean
mean_X_1=np.mean(X_1,axis=0).reshape((p_1,1))
mean_X_2=np.mean(X_2,axis=0).reshape((p_2,1))
# Tính covariance matrix
S_1=np.cov(X_1.T)
S_2=np.cov(X_2.T)
```

```
# Tính S_pooled
S_pooled=(1/(n_1+n_2-2))*((n_1-1)*S_1+(n_2-1)*S_2)
S_pooled_inv=np.linalg.inv(S_pooled)
a=(mean_X_1-mean_X_2).T.dot(S_pooled_inv)
w=a.dot(mean_X_1-mean_X_2)*0.5
```

```
def classification(X,a,w):
    class_id=[]
for i in X:
if a.dot(i.T)>=w:
            class_id.append(1)
else:
            class_id.append(0)
return np.array(class_id)
```

Test

```
#Load tập test
```

```
data_test=pandas.read_excel("Test.xlsx")
X_test=np.vstack((data_test["Freshwater"],data_test["Marine"])).T
y_test=data_test["Class"]
```

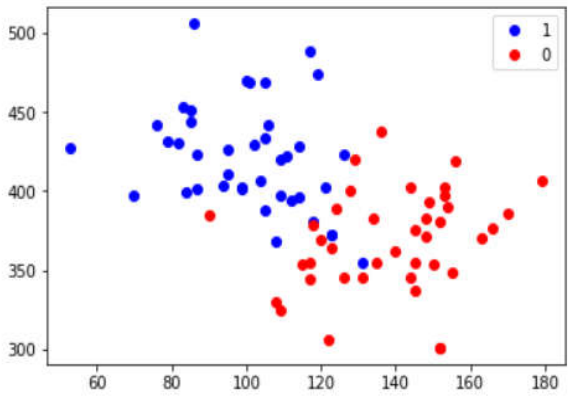
```
#predict test set
y_predict=classification(X_test,a,w)
accuracy_score_test=metrics.accuracy_score(y_predict,y_test)
print("accuracy score test: ",accuracy_score_test*100)
```

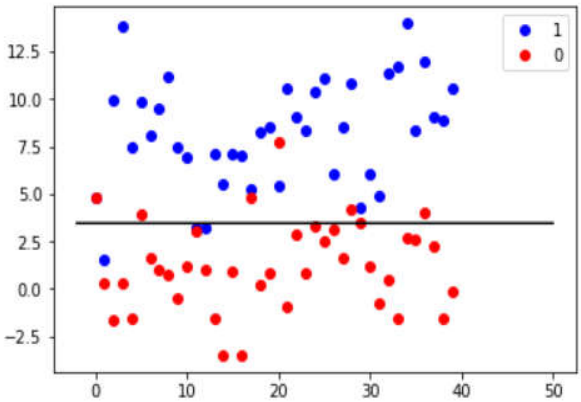
```
accuracy score test:  91.30434782608695
```

Plot

```
# plot
plt.plot(X_1[:,0], X_1[:,1], "o", color="blue",label="1")
plt.plot(X_2[:,0], X_2[:,1], "o", color="red",label="0")
plt.legend()
plt.show()
```

```
# plot
plt.plot(X_1[:,0], X_1[:,1], "o", color="blue",label="1")
plt.plot(X_2[:,0], X_2[:,1], "o", color="red",label="0")
plt.legend()
plt.show()temp_1=a.dot(X_1.T).reshape(-1,1)
temp_2=a.dot(X_2.T).reshape(-1,1)
plt.plot(temp_1,"o",color="blue",label="1")
plt.plot(temp_2,"o",color="red",label="0")
xx = np.arange(-2, 50, 0.01).reshape(-1, 1)
y=w+xx*0
plt.plot(xx,y,color="black")
plt.legend()
plt.show()
```





2. Phân lớp 3 quần thể

Soure: Data for Graduate School of Business

Tính chất áp dụng:

$$\widehat{d}_i(x) = \overline{\mathbf{x_i}}^T \boldsymbol{S_{pooled}}^{-1}\mathbf{x} - \frac{1}{2}\overline{\mathbf{x_i}}^T \boldsymbol{S_{pooled}}^{-1}\overline{\mathbf{x_i}} + \boldsymbol{ln}(p_i)$$

Vậy để $\mathbf{x_0}$ được phân lớp vào $\boldsymbol{\pi_k}$ **thì :**

$$the\ linear \text{ discriminant } \widehat{d_k}(x_0) = \text{ lớn nhất trong } \widehat{d_1}(x_0), \widehat{d_2}(x_0), \ldots, \widehat{d_g}(x_0)$$

Code

```
import numpy as np
import matplotlib.pyplot as plt
import pandas
```

```
# Load dữ liệu
data=pandas.read_excel("Data_Graduate.xlsx")
X_1=np.vstack((data["GPA_Admit"],data["GMAT_Admit"])).T
X_2=np.vstack((data["GPA_not_admit"],data["GMAT_not_admit"])).T
X_3=np.vstack((data["GPA_Borderline"],data["GMAT_Borderline"])).T

#Loai bo NaN
X_1=X_1[~np.isnan(X_1).any(axis=1)]
X_2=X_2[~np.isnan(X_2).any(axis=1)]
X_3=X_3[~np.isnan(X_3).any(axis=1)]
```

```
#Lấy thông tin kích thước dữ liệu và số chiều dữ liệu
n_1=X_1.shape[0]
n_2=X_2.shape[0]
n_3=X_3.shape[0]
p_1=X_1.shape[1]
p_2=X_2.shape[1]
p_3=X_3.shape[1]
```

```
#tính priori probability
temp=n_1+n_2+n_3
priori_prob_1=1.0*n_1/temp
priori_prob_2=1.0*n_2/temp
priori_prob_3=1.0*n_3/temp
priori_prob=np.array([priori_prob_1,priori_prob_2,priori_prob_3])
```

```
# Tính sample mean
mean_X_1=np.mean(X_1,axis=0).reshape((p_1,1))
mean_X_2=np.mean(X_2,axis=0).reshape((p_2,1))
mean_X_3=np.mean(X_3,axis=0).reshape((p_3,1))
```

```
mean=np.array([mean_X_1,mean_X_2,mean_X_3])
```

```
# Tính sample covariance matrix
S_1=np.cov(X_1.T)
S_2=np.cov(X_2.T)
S_3=np.cov(X_3.T)
```

```
# Tính S_pooled
S_pooled=(1/(n_1+n_2+n_3-3))*((n_1-1)*S_1+(n_2-1)*S_2+(n_3-1)*S_3)
# nghịch đảo S_pooled
S_pooled_inv=np.linalg.inv(S_pooled)
```

```
#Hệ số của tổ hợp tuyến tính
a_1=mean_X_1.T.dot(S_pooled_inv)[0]
a_2=mean_X_2.T.dot(S_pooled_inv)[0]
a_3=mean_X_3.T.dot(S_pooled_inv)[0]
a=np.array([a_1,a_2,a_3])
```

$$\widehat{d}_i(x) = \overline{\mathbf{x_i}}^{\boldsymbol{T}} \boldsymbol{S_{pooled}}^{-1} \mathbf{x} - \frac{1}{2} \overline{\mathbf{x_i}}^{T} \boldsymbol{S_{pooled}}^{-1} \overline{\mathbf{x_i}} + \boldsymbol{ln}(\, p_i)$$

```
def discriminant_score(X,priori_prob,a,mean_X):
return np.log(priori_prob)+a.dot(X)-0.5*a.dot(mean_X)
```

```
#Hàm phân lớp
def classification(X,priori_prob,a,mean):
    class_id=[]
for i in X:
        d_1=discriminant_score(i,priori_prob[0],a[0],mean[0])
        d_2=discriminant_score(i,priori_prob[1],a[1],mean[1])
        d_3=discriminant_score(i,priori_prob[2],a[2],mean[2])
if d_1>d_2 and d_1>d_3:
            class_id.append(1)
if d_2>d_1 and d_2>d_3:
            class_id.append(2)
if d_3>d_1 and d_3>d_2:
            class_id.append(3)
return np.array(class_id)
```

```
#Load data test
data=pandas.read_excel("Test1.xlsx")
X_test=np.vstack((data["GPA"],data["GMAT"])).T
y_test=data["Class"]
```

```
#Predict test set
```

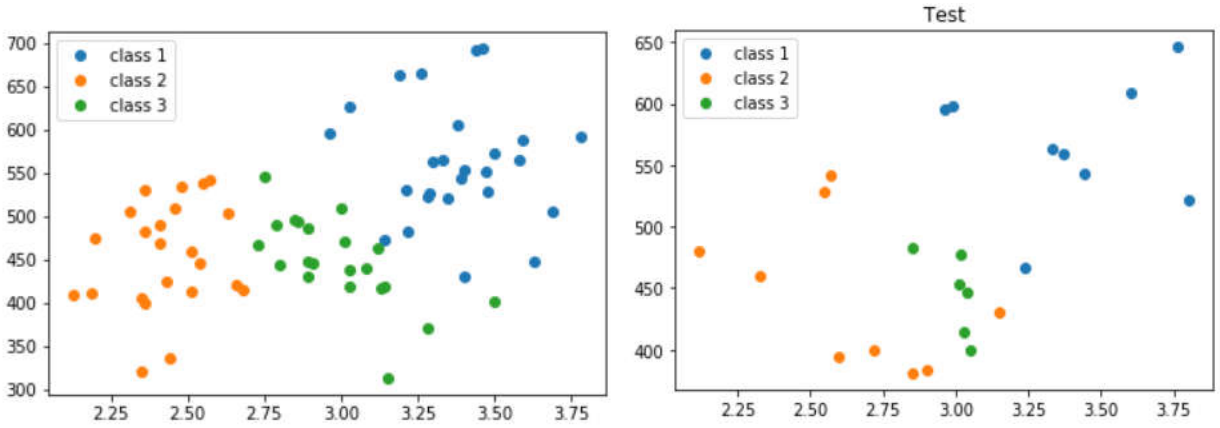
```
y_predict=classification(X_test,priori_prob,a,mean)
accuracy_score_test=metrics.accuracy_score(y_test, y_predict)#tính độ chính xác
print("accuracy score test :",accuracy_score_test*100)
```

```
accuracy score test : 83.33333333333334
```

Plot

```
#plot
plt.plot(X_1[:,0], X_1[:,1], "o",label="class 1")
plt.plot(X_2[:,0], X_2[:,1], "o",label="class 2")
plt.plot(X_3[:,0], X_3[:,1], "o",label="class 3")
plt.legend()
plt.show()
```

```
# Plot
class_1=X_test[y_test==1]
class_2=X_test[y_test==2]
class_3=X_test[y_test==3]
plt.plot(class_1[:,0], class_1[:,1], "o",label="class 1")
plt.plot(class_2[:,0], class_2[:,1], "o",label="class 2")
plt.plot(class_3[:,0], class_3[:,1], "o",label="class 3")
plt.title("Test")
plt.legend()
plt.show()
```



3. LDA with 2 class

```
#Những thư viên cần thiết
import numpy as np
import pickle
import gzip
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
```

```
from sklearn.model_selection import train_test_split
from sklearn import metrics, datasets
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
```

Tự code các hàm của LDA

```
# Dataset
np.random.seed(22)
mean0 = np.array([0, 5])
mean1= np.array([5, 0])
cov0 = np.array([[4, 3], [3, 4]])
cov1 = np.array([[3, 1], [1, 1]])
N0 = 50#số lượng phần tử của class0
N1 = 40#số lượng phần tử của class1
X0=np.random.multivariate_normal(mean0, cov0, N0)
X1=np.random.multivariate_normal(mean1, cov1, N1)
X = np.concatenate((X0, X1))# xếp chồng X0 và x1
y = np.array([0]*N0 + [1]*N1)#gán nhãn cho X0 và X1
```

```
# Build S_B (between-class)
m0 = np.mean(X0.T, axis = 1, keepdims = True)
m1 = np.mean(X1.T, axis = 1, keepdims = True)
a = (m0 - m1)
S_B = a.dot(a.T)
```

```
# Build S_W (within-class)
A = X0.T - np.tile(m0, (1, N0))
B = X1.T - np.tile(m1, (1, N1))
S_W = A.dot(A.T) + B.dot(B.T)
```

```
# Tìm vector riêng ứng với trị riêng lớn nhất
_, W = np.linalg.eig(np.linalg.inv(S_W).dot(S_B))
w = W[:,0]
print(w)
```

```
W: [ 0.75091074 -0.66040371]
```

Dùng hàm có sẵn trong thư viện sklearn

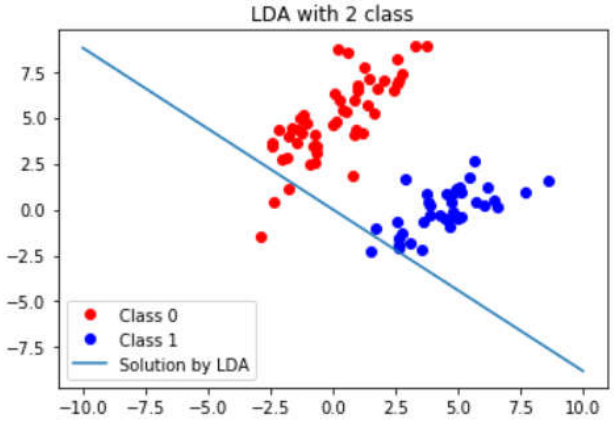
```
# LDA trogn thư viện scikit-learn
clf=LDA()
clf.fit(X,y)#fit tập Dataset
w=clf.coef_/np.linalg.norm(clf.coef_)#vector w đã được chuẩn hóa
Y_predict=clf.predict(X)# predict X
accuracy_score=metrics.accuracy_score(Y_predict,y)#tính độ chính xác của tập
traning
print("accuracy_score: ",accuracy_score)
```

```
print("w: ",w)
```

```
accuracy_score:  1.0
w:  [[ 0.75091074 -0.66040371]]
```

Plot

```
#plot
x = np.linspace(-10, 10, 1000)
plt.plot(X0[:,0],X0[:,1],"o",color='red',label='Class 0')
plt.plot(X1[:,0],X1[:,1],"o",color='blue',label='Class 1')
plt.plot(x,x*w[0,1]/w[0,0],label="Solution by LDA")
plt.title("LDA with 2 class")
plt.legend()
plt.show()
```



4. LDA with multi class

```
# data set
np.random.seed(22)
mean0 = np.array([0, 5])
mean1= np.array([5, 0])
mean2= np.array([4,-5])
cov0 = np.array([[4, 3], [3, 4]])
cov1 = np.array([[3, 1], [1, 1]])
cov2 = np.array([[5, 1], [1, 2]])
N0 = 50#số lượng phần tử của class0
N1 = 40#số lượng phần tử của class1
N2 = 70#số lượng phần tử của class2
X0=np.random.multivariate_normal(mean0, cov0, N0)
X1=np.random.multivariate_normal(mean1, cov1, N1)
X2=np.random.multivariate_normal(mean2, cov2, N2)
X = np.concatenate((X0, X1,X2))# xếp chồng X0 và x1
y = np.array([0]*N0 + [1]*N1+[2]*N2)#gán nhãn cho X0 và X1
```

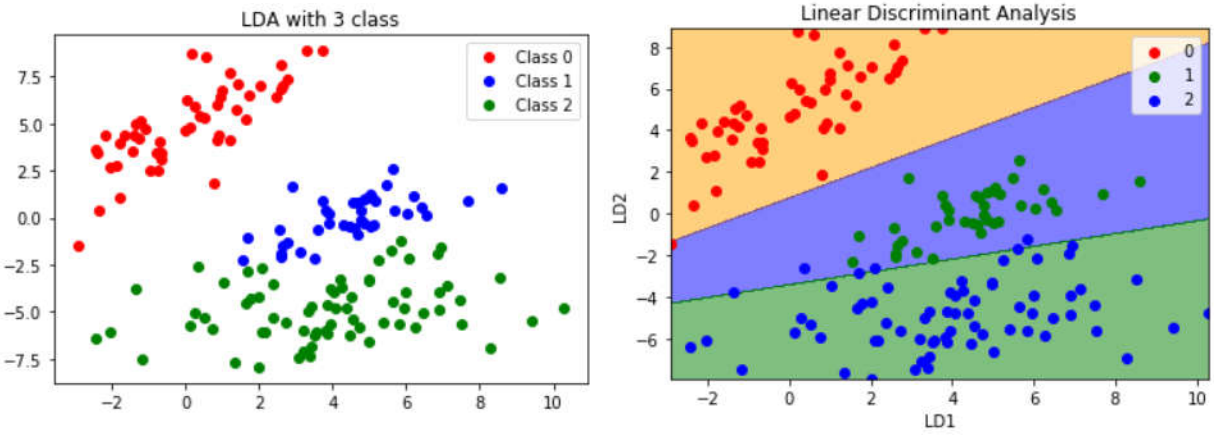
```
# LDA trong thư viện scikit-learn
clf=LDA()
clf.fit(X,y)#fit tập Dataset
w=clf.coef_/np.linalg.norm(clf.coef_)#vector w đã được chuẩn hóa
Y_predict=clf.predict(X)# predict X
accuracy_score=metrics.accuracy_score(Y_predict,y)#tính độ chính xác của tập
traning
print("accuracy_score: ",accuracy_score*100)
print("w: ",w)
```

```
accuracy_score:  96.25
w:  [[-0.42013262  0.70714195]
 [ 0.0930827  -0.00228981]
 [ 0.24690462 -0.50379293]]
```

Plot

```
x = np.linspace(-15, 15, 1000)
plt.plot(X0[:,0],X0[:,1],"o",color='red',label='Class 0')
plt.plot(X1[:,0],X1[:,1],"o",color='blue',label='Class 1')
plt.plot(X2[:,0],X2[:,1],"o",color='green',label='Class 2')
plt.title("LDA with 3 class")
plt.legend()
plt.show()
```

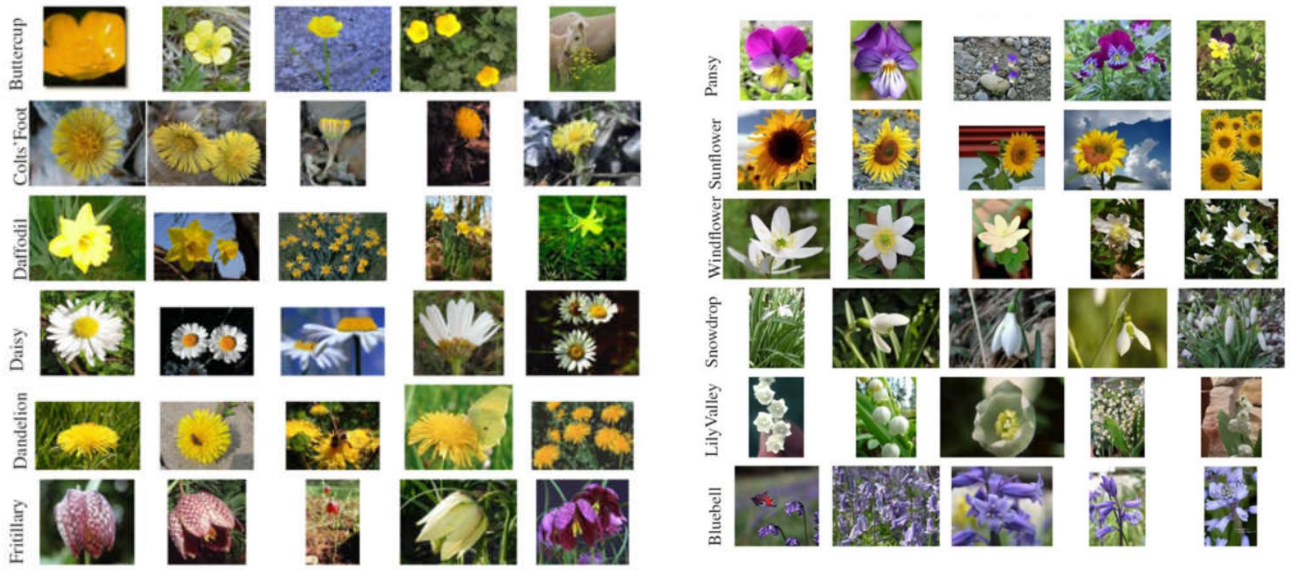
```
X_set, y_set = X, y

aranged_pc1 = np.arange(start=X_set[:, 0].min(), stop=X_set[:, 0].max(), step=0.01)
aranged_pc2 = np.arange(start=X_set[:, 1].min(), stop=X_set[:, 1].max(), step=0.01)

X1, X2 = np.meshgrid(aranged_pc1, aranged_pc2)
plt.contourf(X1, X2, clf.predict(np.array([X1.ravel(),
X2.ravel()]).T).reshape(X1.shape),
alpha=0.5, cmap=ListedColormap(('orange', 'blue', 'green')))

plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
c=ListedColormap(('red', 'green', 'blue'))(i), label=j)
plt.title('Linear Discriminant Analysis')
plt.xlabel('LD1')
plt.ylabel('LD2')
plt.legend()
plt.show()
```



5. Phân loại ảnh hoa dùng LDA

Data set:http://www.robots.ox.ac.uk/~vgg/data/flowers/17/

Data set bao gồm hình ảnh của 17 loại hoa. Mỗi loại 80 ảnh:



Ở đây mình chỉ thực hiện phân lớp 5 loại hoa:

- Coltsfoot
- Snowdrop
- Sunflower
- Tigerlily
- windflower

```
import cv2
import os
import numpy as np
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
from sklearn.externals import joblib
from scipy.cluster.vq import *
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
import time
import sys
```

Training

```
#lấy thông tin ảnh lưu trong dataset
def get_image(train_path):
    image_paths = []#list dùng để lưu đường dẫn đến tập tin ảnh
image_classes = []# list dùng để lưu class được phân lớp sẵn
class_id = 0#biến để đưa class về dạng số
    # lấy đường dẫn các ảnh trong dataset/train
train_labels = os.listdir(train_path)  # lấy tên labels trong  training set
for training_name in train_labels:#Duyệt từng thư mục (class)
dir = os.path.join(train_path, training_name)  # nối path train path+labels
class_path = [os.path.join(dir, f) for f in os.listdir(dir)]#lấy đường dẫn ảnh
trong class
image_paths += class_path# thêm vào list
image_classes += [class_id] * len(class_path)#đánh dấu id của từng ảnh
class_id += 1
return image_paths, image_classes, class_id,train_labels
```

```
#hàm trích xuất đặc trưng bằng thuật toán SURF
def SUFT_feature(image_paths):
    surf = cv2.xfeatures2d.SURF_create()#khởi tạo trích xuất đặc trưng SURF
des_list = []#list dùng để lưu tất cả các đặc trưng trích xuất từ ảnh
start = time.time()
for image_path in image_paths:#duyệt từng đừng dẫn đến ảnh trong tập data set
im = cv2.imread(image_path)# đọc ảnh
des = surf.detectAndCompute(im, None)[1]  # lấy được descriptor
des_list.append(des)#thêm vào list
end = time.time()
print("SURF : ", end - start)
return des_list
```

```
def get_Kmeans(des_list,k):
#Xếp chồng tất cả đặc trưng
descriptors = des_list[0]
for descriptor in des_list[1:]:
        descriptors = np.vstack((descriptors, descriptor))
    voc, variance = kmeans(descriptors, k, 1)# thực hiên phân cum, trả về voc là
tọa độ các điểm đại diện cho cluster
return voc
```

```
#Tính Histogram
def Histogram(des_list,voc,k):
    X_data=[]
for des in des_list:#duyệt đặc trưng của từng ảnh
words, distance = vq(des, voc)#thực hiện phan cum, trả về các vector mô tả về cụm được phân vào
X_data.append(np.bincount(words, minlength=k).reshape(1, -1).ravel())#đưa về histogram
im_features=np.array(X_data,"float32")#đưa về dạng matrix
stdSlr = StandardScaler().fit(im_features)#chuẩn hóa histogram
im_features = stdSlr.transform(im_features)
return im_features,stdSlr
```

```
#Hàm Training bằng LDA
def LDA_traning(im_features,image_classes):
    clf = LDA()
    clf.fit(im_features, np.array(image_classes))#thực hiện phân lớp và trả về clf
return clf
```

```
k=1500
print("Get image path...")
train_path="dataset/train/"
image_paths, image_classes, class_id,train_labels=get_image(train_path)#Thực hiện đọc train set và trả về đường dẫn, class id
print("Run SURF...")
des_list = SUFT_feature(image_paths)
print("Run Kmean...")
voc = get_Kmeans(des_list, k)#Thực hiện phân cụm
print("Run Histogram...")
im_features, stdSlr = Histogram(des_list, voc, k)#Đưa đề histogram
print("Traning...")
clf = LDA_traning(im_features, image_classes)#Thực hiện phân lớp
print("Finish")
```

```
Get image path...
Run SURF...
SURF :  8.612936735153198
Run Kmean...
Run Histogram...
Traning...
Finish
```

Testing

```
#Tính Histogram
def Histogram_testing(des_list,voc,k,stdSlr):
    X_data=[]
for des in des_list:
        words, distance = vq(des, voc)#thực hiện phân cụm và trả về vector mô tả
X_data.append(np.bincount(words, minlength=k).reshape(1, -1).ravel())#thực hiện
```

```
histogram và thêm vào list
im_features=np.array(X_data,"float32")#chuyển thành matrix
im_features = stdSlr.transform(im_features)#thực hiện chuẩn hóa
return im_features
```

```
test_path="dataset/test"
image_paths_test, image_classes_test,
class_id_test,train_labels_test=get_image(test_path)
print("Run SURF...")
des_list = SUFT_feature(image_paths_test)# trích xuất đặc trưng
print("Calculate histogram...")
im_features_test=Histogram_testing(des_list,voc,k,stdSlr)#tính toán histogram và
chuẩn hóa
print("Predict...")
y_predict=clf.predict(im_features_test)# predict tập test
accuracy_score=metrics.accuracy_score(y_predict,np.array(image_classes_test))#tính
độ chính xác
print("accuracy_score: ",accuracy_score*100)
```

```
Run SURF...
SURF :  1.5868709087371826
Calculate histogram...
accuracy_score:  68.18181818181817
```

Kết quả:


snowdrop


snowdrop


sunflower


sunflower


tigerlily


tigerlily


windflower


windflower

6. Phân lớp chữ số viết tay dùng LDA

Link MNIST:http://yann.lecun.com/exdb/mnist/

Bộ dữ liệu được sử dụng là bộ MNIST. Mỗi mẫu (example) trong bộ MNIST gồm: input là ảnh chữ số viết tay grayscale có kích thước 28×28 (như vậy, véc-tơ input sẽ có số chiều là 28×28=784),"correct ouput" ∈ {0, 1, ..., 9}cho biết chữ số tương ứng của ảnh (như vậy, sẽ có tất cả 10 lớp). Dưới đây là một số mẫu trong bộ MNIST

Bạn download file dữ liệu "mnist.pkl.gz" đính kèm. Trong file dữ liệu này, các giá trị pixel đã được scale về[0, 1] bằng cách chia cho 255. Hơn nữa, người ta cũng đã chia cho bạn 3 tập:

- Tập training: gồm 50.000 mẫu
- Tập validation: gồm 10.000 mẫu
- Tập test: gồm 10.000 mẫu

Ở đây tôi chỉ dùng tập traning và test

Code:

```
#những thư viện cần thiết
import numpy as np
import pickle
import gzip
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
from sklearn.metrics import accuracy_score
```

```
# Hàm đọc file MNIST
def read_mnist(mnist_file):
    f = gzip.open(mnist_file, 'rb')
    train_data, val_data, test_data = pickle.load(f, encoding='latin1')
    f.close()

    train_X, train_Y = train_data
    val_X, val_Y = val_data
    test_X, test_Y = test_data
return train_X, train_Y, val_X, val_Y, test_X, test_Y
```

```
train_X, train_Y, _, _, test_X, test_Y = read_mnist('mnist.pkl.gz')#đọc dữ liệu từ
file MNIST
lda=LDA()#khơi tạo LDA
clf=lda.fit(train_X,train_Y)#fit tập train
```

```
Y_predict=clf.predict(train_X)#predict tập train
accuracy_train=metrics.accuracy_score(Y_predict,train_Y)#tính độ chính xác
print("accuracy score training :",accuracy_train*100)
```

```
accuracy score training : 87.048
```

```
Y_predict=clf.predict(test_X)#predict tập test
accuracy_test=metrics.accuracy_score(Y_predict,test_Y)#tính độ chính xác
print("accuracy score :",accuracy_test*100)
```

```
accuracy score : 87.28
```

7. Dùng LDA phân lớp tập IRIS

```
#load data từ có sẵn trong thư viện sklearn
Iris=datasets.load_iris()
```

```
X = Iris.data
```

```
y = Iris.target
#lấy 30 tập data set làm tập test
X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.3,
random_state=1)
```

```
lda=LDA(n_components=3)#khởi tạo LDA
clf=lda.fit(X_train,y_train)#fit tập train
```

```
Y_predict=clf.predict(X_train)#predict tập train
accuracy_train=metrics.accuracy_score(Y_predict,y_train)#tính độ chính xác
print("accuracy score training :",accuracy_train*100)
```

```
accuracy score training : 97.14285714285714
```

```
Y_predict=clf.predict(X_test)#predict tập test
accuracy_test=metrics.accuracy_score(Y_predict,y_test)#tính độ chính xác
print("accuracy score testing :",accuracy_test*100)
```

```
accuracy score testing : 100.0
```

C- Hướng Dẫn Chạy Code

1. Yêu cầu phần mềm

Source code được viết bằng python 3 trên chương trình Jupyter Notebook, vì vậy để sử dụng cần phải cài đặt:

Jupyter Notebook: https://jupyter.org/

Anaconda: https://www.anaconda.com/

Khuyến khích nên cài Anaconda bởi Anaconda bao gồm Jupyter Notebook và các thư viện cần thiết sẽ được cài đặt một cách tự động

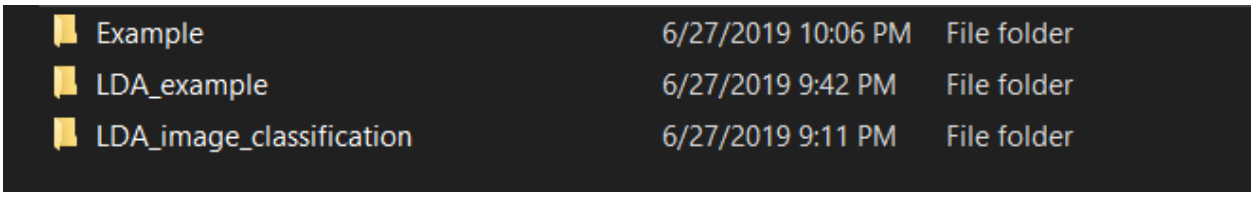
Khi cài đặt, nhớ check dòng “Add Anacondato my PATH environment variable”.

Hướng dẫn sử dụng Jupyter Notebook có thể tham khảo ở link : https://www.dataquest.io/blog/jupyter-notebook-tutorial/

2. Dataset và Source code

Dataset cần dùng tôi đã tải sẵn về trong thư mục chứa Source. Nếu cần cũng có thể tải theo link tôi đã để ở bên trên.

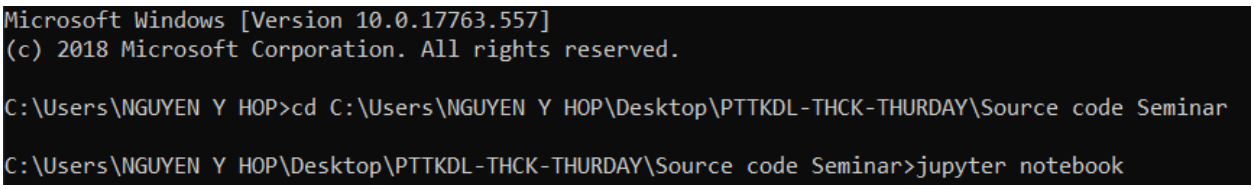
Source code gồm 3 thư mục chính và mỗi thư mục chưa source code + dataset



3. Thực thi code

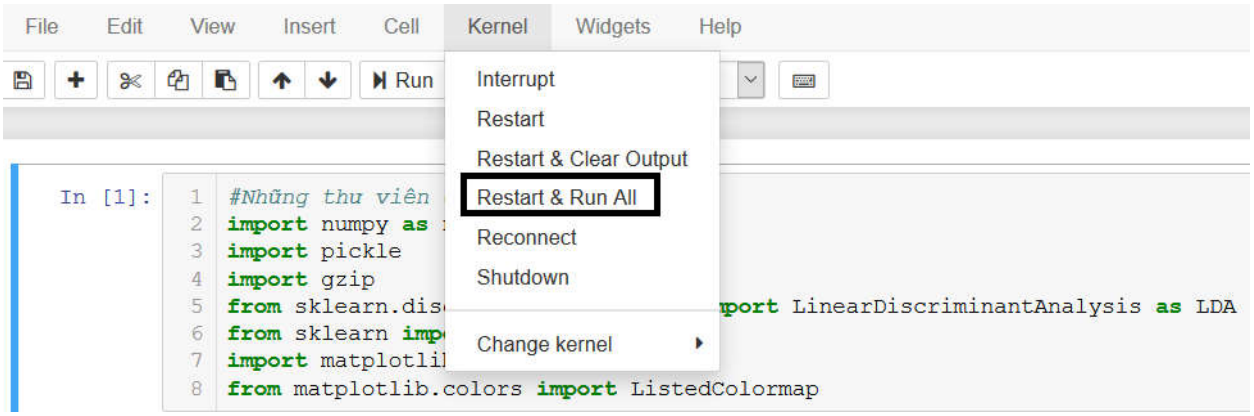
B1: Mở Source code bằng Jupyter Notebook

Giả sử Source code nằm trong thư mục: C:\Users\NGUYEN Y HOP\Desktop\PTTKDL-THCK-THURDAY\Source code Seminar



B2.Chọn File thực thi có đuôi .ipynb (nằm trong mỗi thư mục)

B3: Chọn Kernel-> Restart & Run All



D- TÀI LIỆU THAM KHẢO

[1] Applied Multivariate Statistical Analysis-LQN

[2] https://classifion.sicyon.com/References/discrim.pdf

[3] http://users.stat.umn.edu/~helwig/notes/discla-Notes.pdf

[4] An Introduction To Multivariate Statistical Analysis (Woley Series In probability and Statistics

[5] http://www.sci.utah.edu/~shireen/pdfs/tutorials/Elhabian_LDA09.pdf

[6] http://research.cs.tamu.edu/prism/lectures/pr/pr_l10.pdf