Supervised Learning: the Deterministic Approach

Linear and Augmented Regression Model

September 4, 2023

機器學習（Machine Learning）依資料的特性概分兩種型態：監督式學習（Supervised Learning）與非監督式學習（Unsupervised Learning）。從資料分析與學習的角度來看也分為兩種：決定性（Determinstic）與機率性（Probabilistic）。[1] 本章舉資料的群組分類（Classification）為例，討論採取決定性模型的監督式學習，並著重在最簡單的兩群組資料的判別。透過幾個簡單、典型的方法，實際去做群組的判別。過程中對 Python 程式設計的技巧（依據理論來寫作程式及使用 **sklearn** 套件的指令）、資料的產生及圖形的繪製都有進一步的延伸，也是本章真正的目的。

本章將學到關於程式設計：
群組資料的繪製技巧、排序資料的索引技巧及最小平方法的矩陣計算方式。特殊線性方程式的繪圖、Python 矩陣式的計算技巧。

〈本章關於 Python 的指令與語法〉
套件與指令：
numpy: `hstack,loadtxt,reshape`
matplotlib.pyplot: `contour,legend,scatter,savefig`
scipy.linalg: `inv,pinv`
sklearn.linear_model: `fit,linearRegression,predict,score`

[1] 決定性與機率性的資料分析模式最大的差別在於對資料的假設。當研究者對資料來源掌握更多的資訊，譬如資料來自某個常態分配的母體，若將此機率型態應用在模型中，稱為機率性的方式（Probabilistic Approach），否則稱為決定性的方式（Determinsitic Approach），譬如本章討論的迴歸模型。

1 背景介紹

監督式學習應用在成對的資料 $(x_i, y_i)_{i=1}^N$，其中 x_i, y_i 分別代表變數 X 與 Y 的樣本資料。監督學習的目的是透過這些已知的資料，確立變數 X 與 Y 之間的相關性，通常以數學式 $Y = f(X)$ 表示，典型的案例常見於迴歸分析與時間序列。所謂「監督式」與「非監督」的差別在因變數 Y 是否已知；譬如，某種體積很小的昆蟲，不易辨別其性別，於是想從其張開的翅膀長度來探知其性別。假設翅膀長度變數為 X，性別變數為 Y，由於性別資料未知，只憑樣本資料 x_i 來推估未知資料 y_i，稱為非監督式學習，意即想從已知資料估計（揭露）未知資料（或稱隱藏資料）。另一方面，若透過某些精密儀器的檢測或長期觀察昆蟲的行為，確認了性別資料 y_i，此時想探索的問題變成翅膀長度是否與性別存在某種關係？如果找不到，也許還必須加入第二個因素，譬如體重。這種從成對的資料中找出其對應關係，稱為監督式學習，如圖 1 的示意圖，未知模型的輸入變數 X_1, X_2 代表翅膀長度與體重（稱為特徵值），輸出變數 Y 代表對應的性別（或稱組別）。

圖 1: 監督式學習示意圖，其中輸出變數 Y 的觀察資料已知

本章討論監督式學習中屬於類別資料的群組分辨（即輸出變數 Y 是類別型的資料），[2]並且著重在最簡單的「兩群組資料判別」。透過典型的迴歸分析方法，實際去做群組的判別。過程中依據理論來寫作程式並繪製相關圖形，讓讀者實際理解機器學習的背景。

為求簡單明瞭起見，本章假設輸入資料具兩個維度（變數），即具 X_1, X_2 的兩個特徵值，且每一筆已知資料的群組別 Y 也是已知。譬如圖 2 顯示 400 筆已知資料，包含輸入（X_1, X_2）與輸出（不同的圖示及顏色代表不同的組別），其關係亦如圖 1 所示。而面臨的問題是，如何從已知的 400 筆成對資料學習 X_1, X_2 與 Y 之間的關係，當給予一組未知群組別的資料時，如何預測其組別？[3]

[2]輸出資料概分兩種：Quantitative 及 Categorical，歸類問題的屬性時常以此為分別。當輸出是 Quantitative 型的資料，屬於迴歸分析（Regression）的範疇，當輸出是 Categorical，叫做分類（Classification）或分群。輸入資料當然也有不同的類型，不過應用的方法上差別比較小。Regression 與 Classification 在方法上也有許多類似之處，因為在 Categorical 資料的表達上，通常會以數字來代表類別，譬如 1 代表「成功」，0 代表「失敗」。這樣一來兩者的差距變模糊了，Regression 的方法也可以直接套用在 Categorical 的資料上。

[3]從已知的資料學習，稱為模型的「訓練」階段，從訓練後的模型輸入資料以計算其所屬組別，一般稱為預測。

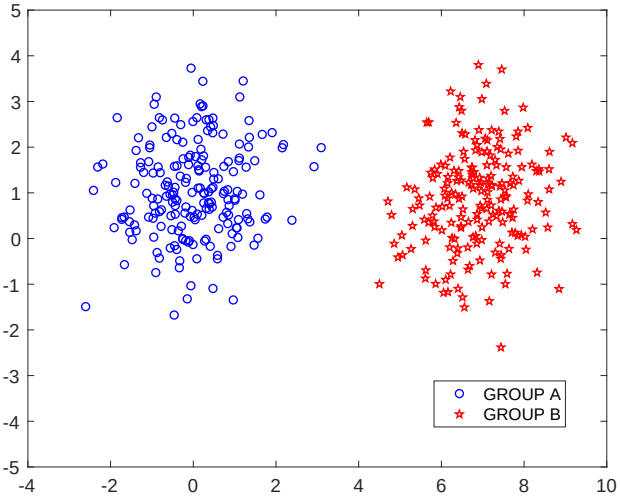


圖 2: 兩群體的輸入（特徵）資料與輸出（組別）資料，共 400 筆

圖 2 的 400 筆資料明顯的將所在的平面空間分成兩半，左半邊屬於群組 A，右半邊屬群組 B。當一筆新的資料需要判別其群組屬性時，只要看它落在平面上的哪一邊，即可判定。但問題是，分割平面空間的分界線如何界定？這條線將做為資料群組預測的根據，但從圖 3 來看，這條分界線似有無限可能，不同的方法形成的分隔線也不同，將如何判斷其優劣呢？[4]

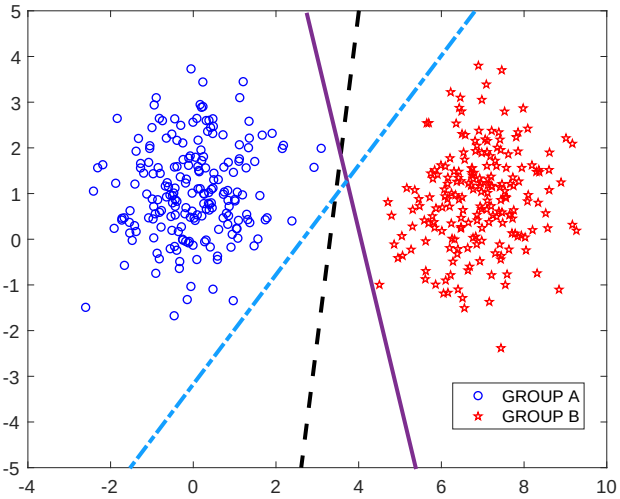


圖 3: 兩個群組的可能分界線

本章介紹線性與非線性迴歸模型與最小平方法在群組分析上的應用，試著在兩群組的資料空間畫上一條適當的分界線，並採理論與實作並進的方式逐步完成程式的設計，其中介紹 Python 在這方面提供的指令與做法。

[4]本章內容僅介紹分隔線的建立，至於不同分隔線的優劣比較留到習題再做分析，請讀者自行寫程式做比較。

1.1 線性迴歸模型

假設圖 1 的輸入輸出關係為「線性迴歸模式」。如果輸出資料屬於類別資料時，譬如，Group A 及 Group B，我們仍可以假設當輸入資料屬於群組 A 時，輸出變數以數字表示，譬如：$Y = 0$，另一個群組則為 $Y = 1$。將類別資料量化之後的問題，便可以直接套入以下的線性迴歸模式（Linear Regression Model）來分析，[5]

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \tag{1}$$

假設共有 N 筆已知的輸入與輸出資料 ($[x_1(i) \;\; x_2(i)], y(i)$)，則迴歸係數 $\beta_0, \beta_1, \beta_2$ 以最小平方法求得的最佳解為

$$\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T \mathbf{y} \tag{2}$$

其中

$$\hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix}, \quad X = \begin{bmatrix} 1 & x_1(1) & x_2(1) \\ 1 & x_1(2) & x_2(2) \\ \vdots & \vdots & \vdots \\ 1 & x_1(N) & x_2(N) \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} \tag{3}$$

分別代表迴歸模型模型的參數估計、輸入及輸出資料矩陣。式 (2) 假設反矩陣 $(X^T X)^{-1}$ 存在，而每個輸出值 $y(i)$ 根據其類別，非 0 即 1。一旦迴歸模型的參數 $\hat{\beta}$ 估計完成，我們說機器完成了學習。接下來便可以拿來對新資料做群組屬性的判別（預測）。規則如：

[5]用某個數學關係式，譬如迴歸模型，去配適變數間的相關性，便是「決定性（Determinstic）」的模式。

群組判別：當給予一個新的輸入資料 $x = (x_1, x_2)$，根據迴歸模型 (1)，其輸出擬合值為：

$$\hat{y} = \mathbf{x}^T \hat{\boldsymbol{\beta}} \tag{4}$$

其中 $\mathbf{x}^T = [1\ x_1\ x_2]$。在迴歸模型下的擬合值 $\hat{y}$ 不一定剛好是 0 或 1，它可以是任何數值，但作為類別判斷時，可以依下列規則判別：假設 G 代表判定的類別：

$$G = \begin{cases} \text{Group } A \ \text{ if } \ \hat{y} \leq 0.5 \\ \text{Group } B \ \text{ if } \ \hat{y} > 0.5 \end{cases}$$

換句話說，上述規則以 $\hat{y} = \mathbf{x}^T \hat{\boldsymbol{\beta}} = 0.5$ 做為平面空間中兩個群組的分界線，將 $\Re^2$ 平面一分為二，線的一邊以集合 $\{\mathbf{x} | \mathbf{x}^T \hat{\boldsymbol{\beta}} \leq 0.5\}$ 代表 Group A，另一邊則為 Group B。很明顯的，這條分界線的形成受到下列因素的影響：

- 已知資料 X 與 $\mathbf{y}$
- 迴歸模式 (1)
- $\hat{\boldsymbol{\beta}}$ 的估計方法，即式 (2) 的最小平方法。

以下練習舉兩組資料為例（從網頁下載 `la_1.txt, la_3.txt` 兩組資料），[6]如圖 4，協助初學者如何畫出群組分佈圖、計算 $\hat{\boldsymbol{\beta}}$ 值與及分界線。

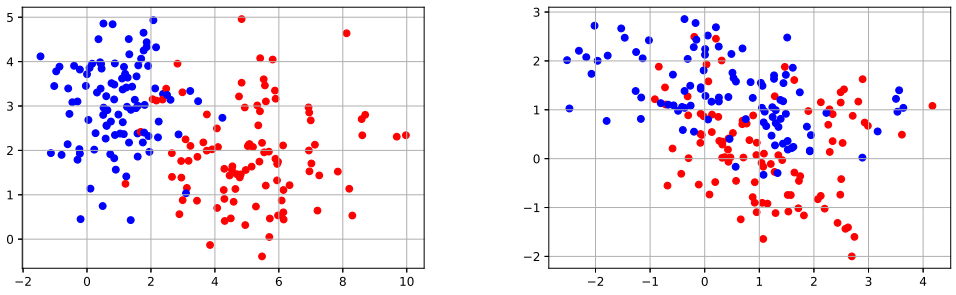


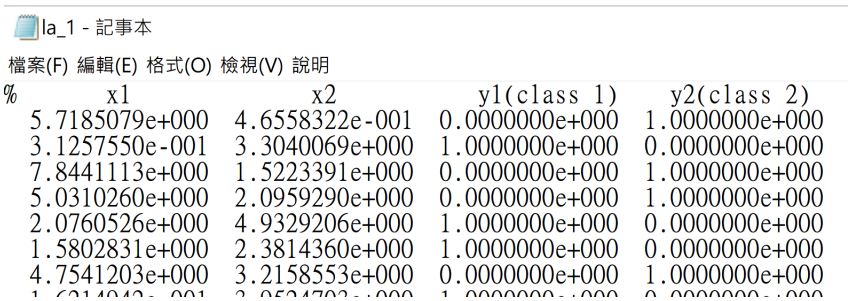
圖 4: 群組資料 `la_1.txt`（左）與 `la_3.txt`（右）

在計算出分界線之前，通常會先將資料畫出來觀察其群組關係。當然這僅限於兩個輸入變數以下的情況。左邊的資料 (`la_1.txt`) 是模擬出來的，右邊 (`la_3.txt`) 則來自參考文獻 [1] 提供的資料。以下練習協助畫出圖 4。

[6]https://ntpuccw.files.wordpress.com/2021/11/ml_data.zip

範例 1. 根據輸出資料 Y 的類別，在 X1-X2 平面上以不同顏色或符號描繪出群組的散佈圖。

在此設定資料檔所在的目錄（Data）與程式檔平行，而資料檔 `la_1.txt` 的內容如圖 5 所示


la_1 - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
%          x1              x2             y1(class 1)     y2(class 2)
  5.7185079e+000  4.6558322e-001  0.0000000e+000  1.0000000e+000
  3.1257550e-001  3.3040069e+000  1.0000000e+000  0.0000000e+000
  7.8441113e+000  1.5223391e+000  0.0000000e+000  1.0000000e+000
  5.0310260e+000  2.0959290e+000  0.0000000e+000  1.0000000e+000
  2.0760526e+000  4.9329206e+000  1.0000000e+000  0.0000000e+000
  1.5802831e+000  2.3814360e+000  1.0000000e+000  0.0000000e+000
  4.7541203e+000  3.2158553e+000  0.0000000e+000  1.0000000e+000
```

圖 5: 群組資料 `la_1.txt` 的內容

Python 的 `matplotlib` 套件可以採用 `plot` 與 `scatter` 指令畫散佈圖。其中 `scatter` 利用顏色來分別群組，使用方式如下：

```
import numpy as np
import scipy.linalg as LA
import matplotlib.pyplot as plt

# 1. load data
data_dir = '../Data/'
D = np.loadtxt(data_dir + 'la_2.txt', comments='%')

# 2. Scatter plot
# --- define attributes for the scatter plot
s = 30 # define the size of markers
# define the colors of markers according to group value 0, 1
# colors = D[:,2]
# colors = ['red' if i == 0 else 'blue' for i in D[:,2]]
colors = [[1,0,0] if i == 0 else [0,0,1] for i in D[:,2]]

plt.scatter(D[:, 0], D[:, 1], c = colors, s = s, \
        marker = 'o', alpha = 0.5)

plt.grid(True)
plt.savefig("../ImgOut/la3.eps", format='eps')
plt.show()
```

以 `scatter` 畫散佈圖，可以定義每個點的大小（`s`）與顏色（`colors`）。在此將大小設為一致（`s=30`），而顏色以資料矩陣中代表群組別的第三欄定義（非 0 即 1）之。若不在意顏色，可以直接以 0,1 值作為顏色值，否則必須另外定義，例如上述程式將群組值為 0 者，訂為紅色（`[1, 0, 0]`）；群組值為 1 者，訂為藍色（`[0, 0, 1]`）。而如上述程式的定義方式，非常簡潔明快，值得初學者模仿使用。[7]

另，資料檔為 txt 的文字檔型態，其中第一行（見圖 5）以% 作為註解行，因此在輸入資料的指令中必須明示註解行的標示（`np.loadtxt(..., comments='%')`），以便在讀入資料時排除。

畫散佈圖的第二個方式是先將資料依群組分開，分別以 `plot` 指令繪製，示範如下。其中，在 `plot` 指令加入標籤（`label`）文字，方便在圖形帶出 legend。[8]

```
Idx = (D[:,2]==0)
plt.plot(D[Idx, 0], D[Idx, 1], 'ro',\
    alpha = 0.5, label = 'Group A')

Idx = (D[:,2]==1)
plt.plot(D[Idx,0], D[Idx,1],'bo', \
    alpha = 0.5, label = 'Group B')

plt.legend(), plt.grid(True)
plt.xlabel('$X_1$'), plt.ylabel('$X_2$')
```

上述程式運用了矩陣的索引方式，以擷取矩陣中所需的部分資料。值得初學者模仿。

範例 2. 利用前述範例的資料，估計迴歸模型的參數 (2) 並畫出式 (4) 中 $\hat{y} = 0.5$ 的分界線，也就是劃開兩群組空間的分界線。

計算式 (2) 的 $\hat{\boldsymbol{\beta}}$ 比較簡單，先從原始資料建構資料矩陣 X 與 $\mathbf{y}$，再套入 (2) 的公式即可。接續之前的程式碼，$\hat{\boldsymbol{\beta}}$ 的估計與分界線的呈現如下列程式碼與圖 6。

```
# 3. Estimate the coefficients
```

[7] 這個做法稱為 list comprehension (loop in bracket)。
[8] 指令 `scatter` 也可以加入 legend。

```
n = len(D[:, 0])
X = np.hstack((np.ones((n, 1)), D[:, 0:2]))
y = D[:, 2]
b = LA.inv(X.T @ X) @ X.T @ y.T

# Draw a linear regression line
x = np.array([1.5, 4.5])
y = -(b[0] - 0.5 + b[1] * x) / b[2]

plt.plot(x, y, lw=3)
plt.show()
```

其中估計式 (2) 也可以直接以 pseudo invesre 計算。如

```
b = LA.pinv(X) @ y
```

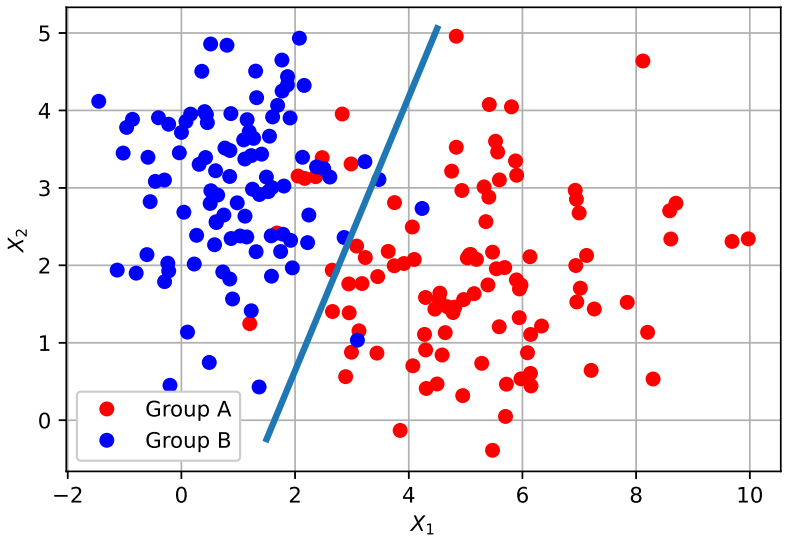


圖 6: 線性迴歸模型的分界線，資料來源 `la_1.txt`

要畫出兩群組間的分界線 $\hat{Y} = 0.5$，需要琢磨一下。這條分界線的方程式可以表示為

$$x_2 = f(x_1) = (0.5 - \hat{\beta}_0 - \hat{\beta}_1 x_1)/\hat{\beta}_2$$

如上述程式所採用。或表示為雙變量函數

$$f(x_1, x_2) = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2$$

再繪製等高線圖，高度設為單一值 0.5。程式示範請參考下一節繪製加廣型迴歸模型的曲線。

範例 3. Python 的 **sklearn** 套件提供了 **linear_model** 模組，其中指令 LinearRegression 建立了如式 (1) 的線性迴歸模型。本範例展示其使用方式，並利用資料檔 `la_3.txt` 示範繪製分界線，如圖 7。

承接上述程式碼，將參數估計部分以下列程式碼取代。

```
from sklearn.linear_model import LinearRegression

# Linear Regression by sklearn
Mdl = LinearRegression()# 建立新的 linear regression model
Mdl.fit(X, y) # 進行估計（配適）
# R2 = Mdl.score(X, y) # R-square
intrcp = Mdl.intercept_ # 估計結果：beta0
coeffs = Mdl.coef_ # # 估計結果：beta1, beta2, ...

# Draw a regression line
x = np.array([-3, 5])
f = -(intrcp - 0.5 + coeffs[0] * x) / coeffs[1]

plt.plot(x, f, lw=3)

# Calculate testing error
y_hat = Mdl.predict(X) # 預測或計算擬合值
y_pre = [1 if i > 0.5 else 0 for i in y_hat]  # 群組判讀

plt.title("Accuracy in training is {:.2f}".format( \
        100 * np.mean(y_pre == y)))
plt.xlabel("$X_1$"), plt.ylabel("$X_2$")
plt.show()
```

上述程式碼展示了 Python 在 Machine Learning 的基本運作模式：第一、建立模型（如 `MDL = LinearRegression()`），第二、參數估計（資料配適 `MDL.fit()`），第三、處理資料配適過程中所產生的結果，包括：所估計的所有參數、配適度（R^2）... 等視模型而定的其他「副產品」。Machine Learning 另一個重要的目的是預測，因此相關的模型除了具備做估計（配適）的 `fit()` 功能外，還有 `predict()` 可供未來資料做預測之用。如果使用配適時的資料做預測，則預測值一般也稱為「擬合值」（Fitted values）。從擬合值（`y_hat`）與原始值（`y`）的誤差，便可得知模型配適的誤差。由於線性迴歸模型的輸出結果並非類別型態，必須先做轉換再與群組資料（`y`）做比較。

讀者可以比較前述兩個範例，若針對同一筆資料，是否得到相同結果？答案是

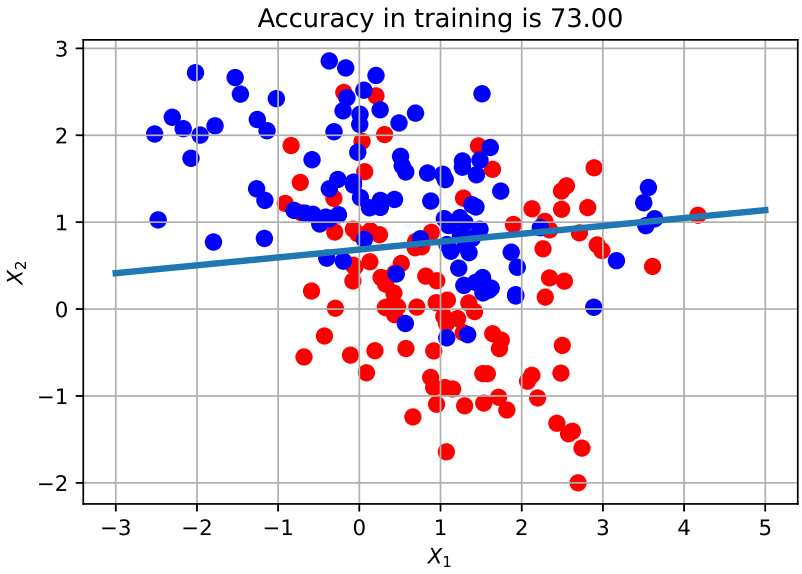


圖 7: 線性迴歸模型的分界線：資料檔 `la_3.txt`

肯定的。這也是從事研究工作者檢驗軟體提供的指令適用與否的方式，進而能更準確地掌握該指令的所有功能，而非盲目，甚至無知地使用。

1.2 加廣型迴歸模型（Augmented Regression Model）

從圖 7 中資料分布的情況，很清楚的發現這組資料的兩群組較密合，於是直線的分界線產生較大的判別誤差，這個誤差在機器學習的領域被稱為「訓練誤差（Training Error）」。想降低訓練誤差的方式很多，其一是變更模型，譬如改為加廣型迴歸模型，這是一條非線性的分界線，也許能提供更適切分隔效果。

假設輸入變數為 X_1, X_2，則 (X_1, X_2) 所有可能的值涵蓋二度空間。此時如果將兩個變數擴展為五個變數 $X_1, X_2, X_1X_2, X_1^2, X_2^2$，同樣利用迴歸模式與最小平方法建立一條分界線，當將此分界線投映回原來的空間時，它將呈現出一條曲線。這五個變數因其彼此相關的本質，並非將空間拓展為五度空間，實際仍在二度空間裡，這個所謂的加廣型迴歸模型寫成

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \beta_4 X_1^2 + \beta_5 X_2^2 \tag{5}$$

其參數的估計與群組判別的方式與線性模式相同，在程式的編寫上只需在資料矩陣 X 再加入三欄分別來自 X_1X_2, X_1^2, X_2^2 的資料，即

$$X = \begin{bmatrix} 1 & x_1(1) & x_2(1) & x_1(1)x_2(1) & x_1^2(1) & x_2^2(1) \\ 1 & x_1(2) & x_2(2) & x_1(2)x_2(2) & x_1^2(2) & x_2^2(2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_1(N) & x_2(N) & x_1(N)x_2(N) & x_1^2(N) & x_2^2(N) \end{bmatrix} \tag{6}$$

接著計算參數 $\hat{\beta}$ 的最小平方估計：$\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T \mathbf{y}$，如同線性迴歸模式，則式 (5) 的加廣型迴歸模型的分界線表示為集合

$$\left\{(X_1, X_2) | \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3 X_1 X_2 + \hat{\beta}_4 X_1^2 + \hat{\beta}_5 X_2^2 = 0.5\right\}$$

在平面上如圖 8 那條彎彎的線，函數寫成

$$f(X_1, X_2) = \hat{\beta}_0 - 0.5 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3 X_1 X_2 + \hat{\beta}_4 X_1^2 + \hat{\beta}_5 X_2^2 \tag{7}$$

讀者可以自己寫程式計算上述分界線的係數 $\hat{\boldsymbol{\beta}}$，並試著用等高線圖畫出分界線函數 (7)，參考程式碼如：

```
# Augmented regression line by a contour line at 0.5
x1 = D[:, 0:1]  # n x 1 vector
x2 = D[:, 1:2]
X = np.hstack((np.ones((n, 1)), \
        x1, x2, x1 * x2, x1 ** 2, x2 ** 2))
y = D[:, 2:3]
b = LA.pinv(X) @ y # pseudo inverse
f = (
    lambda x: b[0]
    + b[1] * x[0]
    + b[2] * x[1]
    + b[3] * x[0] * x[1]
    + b[4] * x[0] ** 2
    + b[5] * x[1] ** 2)

xx = np.linspace(x_min, x_max, nx)
yy = np.linspace(y_min, y_max, ny)
X, Y = np.meshgrid(xx, yy)
Z = f([X, Y])

contours = plt.contour(
    X, Y, Z, levels = [0.5], colors="g", linestyles="--")
plt.title("Linear␣and␣Augmented␣Regression␣lines")
plt.show()
```

上述程式碼以繪製雙變量函數 (7) 在單一高度（0.5）的等高線圖來呈現 X_1 與 X_2 之間的非線性分界線。在建構式 (6) 的資料矩陣 X 時，利用 **numpy** 套件的向量組合指令 `hstack()` 來組合不同的行向量（Column vectors）成為矩陣。此時參與組合的向量必須有相同的維度，譬如，$n \times 1$，因此採用 `x1 = D[:, 0:1]`。這有別於同樣讀取矩陣 `D` 的第一行資料，`x1 = D[:, 0]` 將得到一個維度不明確的向量，當置入指令 `hstack()` 內時，會得到錯誤訊息。理由是 Python 語言對向量不分行或列，一概視為 `(n,)`，而非 `(n,1)` 或 `(1,n)`。[9]

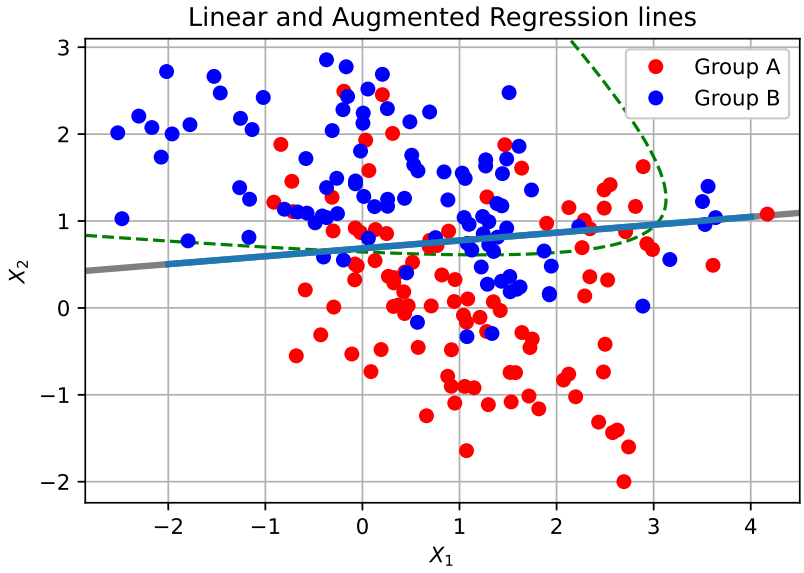


圖 8: 線性與加廣型迴歸的非線性群組分界線

1.3 學習器評比

在機器學習的領域，將不同的學習方法（模型）通稱為學習器，例如線性迴歸模型與加廣型迴歸模型都是學習器。而學習器的選擇、訓練與評比是機器學習的重要步驟。前兩節選擇兩種學習器並經過模擬或實務資料的訓練，接著我們想知道（一）學習器從訓練資料學習得有多好？（二）經過訓練後的學習器面對新的資料時（不同於訓練用的資料），表現如何？

圖 8 的直線與彎曲分界線，分別代表線性迴歸模型與加廣型迴歸模型兩種學習器的學習（訓練）成果。哪一條線的分群效果較好呢？或說，何者對於訓練資料的分群誤差比較小呢？其實直接將迴歸模型用於分群，有點突兀，理由是如式 (1) 與式 (5) 的迴歸模型，主要用在反應變數為連續型的資料，而非群組型的

[9]試試另一個組合向量的指令 `np.c_[...]`，可以直接使用不分行列的向量。其實，不論可以或不可以，記得規則或忘記也罷，程式設計者都能很快寫幾行指令來確認心中的疑惑。

類別資料。[10]當考慮迴歸模型對群組資料的配適性時，並不會參考迴歸分析常見的 R-Squared 或 Adjusted R-Squared 等值。反而是計算學習器對於訓練資料的群組錯判率，[11]才是觀察重點。

範例 4. **sklearn** 套件提供 **linear_model** 模組及指令 LinearRegression 建立了式 (1) 的線性迴歸模型，同樣也適用於加廣型迴歸模型 (5)。請試著製作如圖 9 中兩個學習器對於資料 `la_3.txt` 分群的錯判率（misclassification rate）。

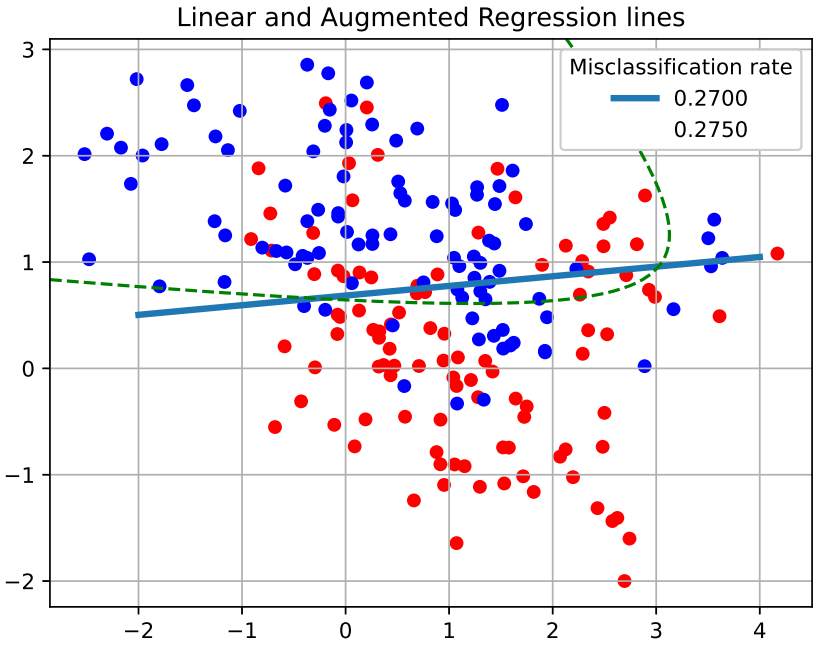


圖 9: 線性與加廣型迴歸的錯判率

指令 LinearRegression 適用於線性與加廣型迴歸模型，做法也完全一樣，差別僅在所給的資料結構。下列程式碼的資料矩陣 X 為加廣型，所以參數估計便會是 $\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_5$，讀者不妨列印出來與之前的程式碼比較。

```
Mda = LinearRegression()
x1, x2 = D[:, 0:1], D[:, 1:2]
X = np.hstack((np.ones((n, 1)), \
        x1, x2, x1 * x2, x1 ** 2, x2 ** 2))
Mda.fit(X, y)
```

[10]群組型的類別資料常使用羅吉斯迴歸模型（Logistic Regression）。詳見第 2 節「觀察與延伸」的介紹。

[11]學習器學得好不好，一般以訓練誤差來評比。以圖 7 的直線分界線為例，這條線是訓練後作為分群的依據。群組錯判率便是計算錯誤分群的資料比例。

```
y_hat = Mda.predict(X)
# convert y_hat to group number 0, 1
y_pre = [1 if i > 0.5 else 0 for i in y_hat]
misclassification_rate = 1 - np.mean(y_pre == y))
# 列印出估計的參數
intrcp = Mda.intercept_
coeffs = Mda.coef_
print(intrcp)
print(coeffs)
```

從圖 9 來看，加廣型迴歸模式的分界線的表現不一定比線性迴歸模式好。從機器學習的角度來看，這不代表面對新的資料時，也會表現比較差。本章後面的習題請讀者將圖 9 的資料檔 `la_3.txt` 分成兩部分（比例 8:2)，佔八成的資料分別用來訓練兩個迴歸模型，並計算錯判率。之後，利用另外兩成資料當作新資料，做為測試之用。測試資料的錯判率往往才是好與壞的參考指標。[12]

另外，圖 9 將兩個模型的錯判率寫在 legend 裡面。這裡有個小麻煩，一般的 `plot` 圖形可以指令裡面加 label 做為 `legend` 的輸出字串。但是對 `contour` 圖而言，並沒有 label 這個擴充參數。彌補的方式可以參考下列程式碼：

```
% 繪製 linear regression line
plt.plot(x, f, lw=3, \
        label = '{:.4f}'.format(1 - np.mean(y_pre == y)))

% 繪製 augmented regression line
contours = plt.contour(
    XX, YY, Z, levels = [0.5], colors="g", linestyles="--"
)
contours.collections[0].set_label(\
        '{:.4f}'.format(1 - np.mean(y_pre == y)))

plt.legend(title = "Misclassification rate")
```

2 觀察與延伸

1. 當分界線劃上去之後，有多少資料被錯置組別呢？錯置的資料愈多，代表什麼意義？當兩個群組部分交錯時，資料的錯置是否不可避免？有更好的分界線可以讓錯置的情況降低嗎？這些都是我們考量使用何種學習器，必須思考的問題。

[12]當資料必須按比例被分為訓練與測試兩分資料時，若資料量不夠大，將造成訓練資料不足，導致學習成效不彰。

2. 使用已知的資料做出一條分界線，企圖將原母體在空間中的範圍切割出來。這個切割的好壞當然取決於已知資料的品質及分界線的決定方式。試試看給予一些新的資料（從原母體去產生），測試一下這條分割線能否對新的資料做出正確的組別判斷？譬如 100 個新資料有多少比率被正確辨別？

3. 由於資料的取得誤差或樣本數不夠，群組的區隔有時候不是很明顯，當然也可能是群組本身就非常靠近。圖 4 左圖的資料看起來分離的很好，直覺上比較容易作區域的切割，如中間的那一條分界線。而右圖的兩個群組相對緊密，即使能劃上一條分隔線，也可能必須選擇曲線比較能滿足現有資料能提供的訊息。而根據有限的資料做出最好的判斷，就是這門學問的精神所在。

4. 當群組數量大於 2 時，分界線將如何切割？想一想。手癢的話就動手做看看吧！

5. 本單元的資料模擬自雙變量常態的母體（Bivariate Normal Distribution），而且兩個變數是獨立的。如果變數間有相依性，本單元的方法還是可行嗎？如何去模擬具相依性的資料呢？模擬資料的產生可參考下列程式碼。假設資料來自兩個雙變量常態的母體，其平均數、共變異矩陣與樣本數分別為：

$$\boldsymbol{\mu}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \boldsymbol{\mu}_2 = \begin{bmatrix} 4 \\ 1 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1 & 0.2 \\ 0.2 & 1 \end{bmatrix}$$

$$n_1 = 200, n_2 = 200$$

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import multivariate_normal

n1, n2 = 200, 200
m1, m2 = np.array([0, 0]), np.array([4, 1])
Cov1 = np.array([[1, 0], [0, 1]])
Cov2 = np.array([[1, 0.2], [0.2, 1]])

mvn1 = multivariate_normal(mean = m1, cov = Cov1)
mvn2 = multivariate_normal(mean = m2, cov = Cov2)

A, B = mvn1.rvs(n1), mvn2.rvs(n2)
X = np.vstack((A, B))
```

```
y = np.hstack((np.zeros(n1), np.ones(n2)))

np.savetxt('demo_data.txt', np.c_[X, y],\
     fmt = "%.4f␣%.4f␣%d", header = "X1␣X2␣y")

colors = ['red' if i == 0 else 'blue' for i in y]

plt.scatter(X[:, 0], X[:, 1], \
        c=colors, s=30, marker="o")
plt.grid(True)
plt.show()
```

上述程式碼生成的資料以 `txt` 的格式被儲存在 `demo_data.txt`，包含兩個變量 X_1, X_2 與群組值變量 y 共 400 筆資料，並在第一行賦予註解行文字（header）。生成資料的散佈圖如圖 10 所示。

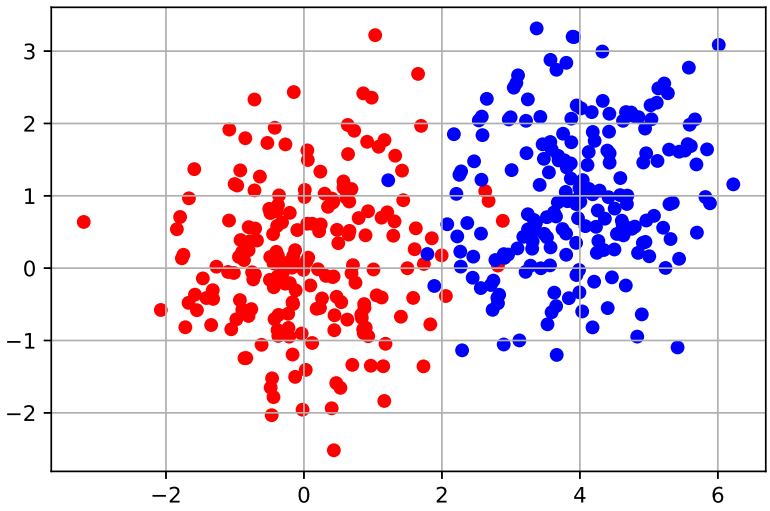


圖 10: 服從兩個雙變量常態的模擬資料散佈圖

6. 羅吉斯迴歸（logistic regression）：本文以線性與加廣型迴歸做為分群的模型其實並不恰當，雖然可以將擬合值轉換為 0 與 1 的群組類別，但是當群組數量大於兩組時，便會遇到困難。因此當輸出為類別型態時，以羅吉斯迴歸模型較為合適，如式 (8) 以兩個群組別為例。

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \tag{8}$$

其中 p 代表其中一個類別發生的機率，可以寫為

$$p(X_1, X_2) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2}}{1 + e^{\beta_0+\beta_1 X_1+\beta_2 X_2}}$$

讀者可以試著使用 **sklearn** 套件提供 **linear_model** 模組中的 LogisticRegression，先套用在本文的資料，如 la_3.txt，並與其他兩個方法做比較。再來嘗試三個甚至多群組的模擬資料。圖 11 參考 **sklearn** 的 LogisticRegression 使用說明的範例製作的（繪圖部分參考 Logistic Regression 3-class Classifier 的鳶尾花範例）。

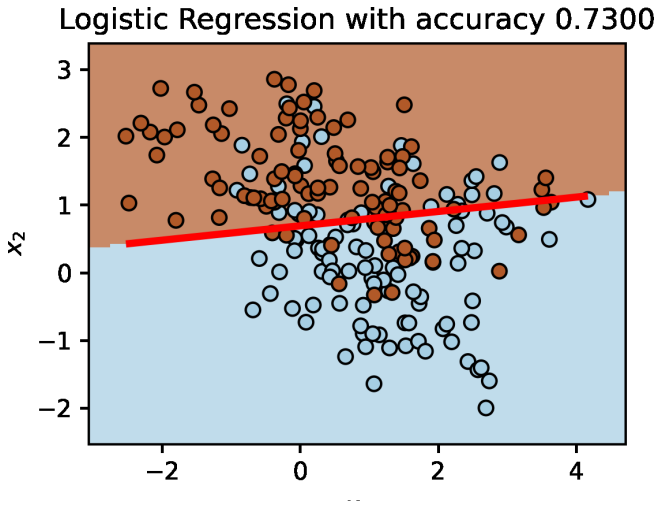


圖 11: 羅吉斯迴歸的準確率：資料來源 la_3.txt

3 習題

1. 證明式 (2) 是迴歸模型 (1) 的最小平方法解，即

$$\hat{\boldsymbol{\beta}} \doteq arg \min_{\boldsymbol{\beta}} ||X\boldsymbol{\beta} - \mathbf{y}||^2$$

2. 當資料來源為 `la_2.txt`，請畫出圖 8 的資料散佈圖與兩條分界線並計算分群錯判率。這裡採用「分群錯判率（Misclassification rate）」，也稱為訓練誤差（Training error）。

3. 圖 12 展示線性迴歸與加廣型迴歸模型在訓練階段的誤差。但依訓練資料所做的錯判率並不能當作判別式好壞的依據，尚須對訓練資料以外的資料做群組判別測試，才能論定。請將現有的資料檔 `la_3.txt` 分成兩部分（比例 8:2），佔八成的資料分別用來訓練兩個迴歸模型，並計算錯判率。再將訓練好的判別式測試另外 2 成的資料，再計算出兩個模型針對測試資料的錯判率。

4. 同前題，資料來源換成檔案 `la_1.txt`、`la_2.txt`。

5. 模擬資料能提供大量的訓練資料與測試資料，不受限於真實資料有限的樣本。於是在驗證模型分群效果時，常常需要大量製造模擬資料。請安排兩

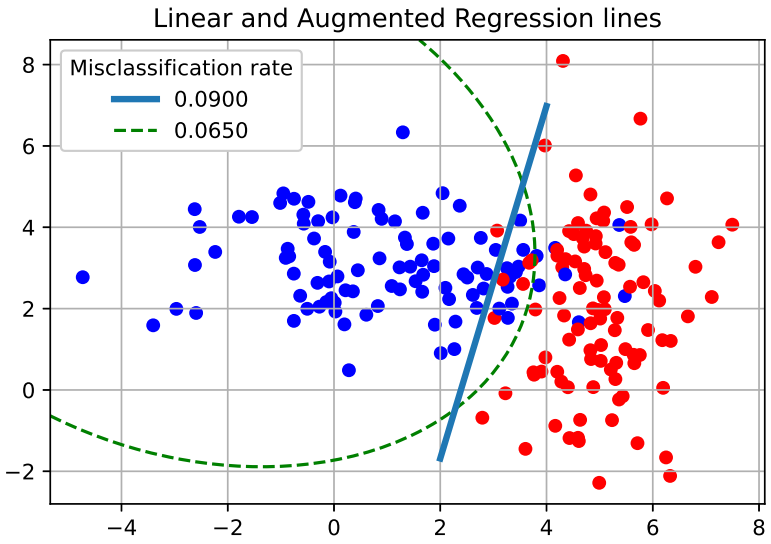


圖 12: 兩個迴歸模型的分群錯誤率比較：資料來源為 `la_2.txt`

種群組分隔的模擬情境，其一使兩群組分隔較遠，另一種則是較近。資料皆來自雙變量常態的母體，平均數與標準差自訂（可參考上一節的模擬資料的程式碼）。

6. 本章的示範以兩群組資料為主，請讀者自行模擬三個群組的資料，能否畫出其間的兩兩的分隔線或是將空間區分為分屬三個群組的區塊。

References

[1] T. Hastie, R. Tibshirani, J. Friedman, ”The Elements of Statistical Learning:Data Mining, Inference, and Prediction,” Springer.

[2] A.G. Rencher, ”Multivariate Statistical Inference and Applications,” John Wiely & Sons, INC.