Supervised Learning: the Probabilistic Approach

K Nearest Neighbors Method

November 18, 2021

對空間中的群組資料作分組切割，在應用上很常見，因此衍生的方法也很多，譬如採用線性迴歸模型，並運用最小平方法確立模型的參數。這個方式雖簡單，但是切割的效果往往隨著資料來源的佈散愈趨緊密交錯而顯得「心有餘而力不殆」。從二度空間的資料來看，線性迴歸模型試圖在交錯的資料群中畫出一條分界線，將空間一分為二，進而確立群組在空間中的的範圍。這條線切的好不好，影響了後續做群組判別的準確性。本章舉有名的 KNN (K Nearest Neighbors) 方法為例，說明 KNN 依直覺出發的觀點，從機率切入的推導，配合必要的假設，最後看看其分群效果如何。

本章將學到關於程式設計

sklearn 套件使用、特殊散佈圖、Python 矩陣式的計算技巧。

〈本章關於 Python 的指令與語法〉

指令：

Python: magic('reset -sf'), sys.path.append

numpy:loadtxt, tile, linalg.norm, argsort, ravel

seaborn: scatterplot

matplotlib.pyplot:contourf

sklearn.neighbors:KNeighborsClassifier, score, predict

1 背景介紹：K Nearest-Neighbor Method

有時候我們對資料的來源並非一無所知，但是採用最小平方法的迴歸模型，並沒有充分利用資料本身的訊息，譬如資料的變異性。我們希望資料的來源或資料的本身可以提供更多的訊息，做為新資料所屬群組的判別依據。這樣的想法把問題帶進「機率」的範疇來解決。[1]

假設 Y 及 $f(X)$ 分別代表輸出變數與輸出預測值，[2]其中 $X \in R^p$ 表示有 p 個輸入變數。我們期望輸出值與預測值的誤差愈小愈好，如果輸入變數 X 與輸出變數 Y 的聯合機率密度函數 $Pr(X,Y)$ 已知的話，[3]這個問題可以寫成：

$$\min_{f(X)} E_{XY}\left[(Y - f(X))^2\right] \tag{1}$$

也就是找一個輸入與輸出變數間的關係式 $f(\cdot)$ ，使得真正的輸出值 Y 與其預測值 $f(X)$ 間的誤差的平方期望值越小越好。有別於不論機率特性的「最小平方法（Least Squared Errors, LSE）」，這個方法稱為「最小均方誤差（Minimum Mean Squared Error, MMSE）」。在已知樣本值 $X = \mathbf{x}$ 的條件下，經過一番推導之後（習題 1），它的最佳解如（未知函數 $f(X)$ 的最佳選擇）

$$y = \hat{f}(\mathbf{x}) = E_{Y|X}\left(Y|X = \mathbf{x}\right) \tag{2}$$

其中 X, Y 代表輸入輸出變數，$\mathbf{x}$ 與 y 表示輸入值及輸出的預測值（或稱擬合值）。式 (2) 說明當輸入值為 $\mathbf{x}$ 時，最佳的輸出預測值為輸出變數的「條件式均值（Conditional Mean）」。

接下來的問題是如何計算 $y = E_{Y|X}\left(Y|X = \mathbf{x}\right)$？期望值代表的是理論值，至於要如何落實到實際的應用呢？或說若不知道機率密度函數 $Pr(Y|X)$，如何得到這個期望值呢？實務的作法，一般都是利用平均數來估計這個期望值。譬如式 (3) 是個不錯的估計式

[1]一般人不習慣從機率的角度來思考問題，因此解決的方式便侷限在既定的參考模式，譬如套用迴歸模型或其他數學模型，而忽略了問題或資料本身的隱藏資訊。所以學習機率，不只學習它的數學部分，更應該留意機率用來解決現實問題的目的。如此才會在平時思考問題時，自然融入機率的角度。

[2]函數 $f(\cdot)$ 代表輸入與輸出間的關係，而且沒有指定特別型式，譬如迴歸模型 $f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2$。

[3]假設未知的聯合機率密度函數 $Pr(X,Y)$ 為已知，是研究科學問題常用的手段。先不論這個假設的合理性或事後如何「自圓其說」，總是先找到切入點並能解決問題，之後再想辦法突破假設的限制。建議讀者多學學這個解決問題的漸進式手段。

$$\hat{y} = Ave(y_i|X = \mathbf{x}) \tag{3}$$

其中 $Ave(\cdot)$ 代表求平均值。這個估計式解讀為「將輸入資料為 **x** 的所有資料，找出對應的所有輸出值 y_i 取平均」。雖然樣本平均數是期望值的不偏估計，不過這個做法面臨實際的困難，理由是已知的多變量連續型資料中，剛好等於 **x** 的機率等於 0，估計式 (3) 在實務上不可行。[4]

將式 (3) 稍作修改後，下面這個輸出預測值的估計式舒緩了這些困擾。[5]

$$\hat{y} = Ave\left(y_i|\mathbf{x}_i \in N_K(\mathbf{x})\right) = \frac{1}{K}\sum_{\mathbf{x}_i \in N_k(\mathbf{x})} y_i \tag{4}$$

式 (4) 解讀為：從已知的資料中找到 K 個最靠近 **x** 的資料（這是 $N_K(\mathbf{x})$ 的意義），將這些鄰近的 K 筆資料所對應的 y 值平均起來作為「條件式均值」的估計，這個方法叫做 K Nearest-Neighbor method。目前為止所提到的輸出變數並不侷限任何型態，但若輸出變數 Y 的群組屬性屬類別資料時，如式 (1) 的 MMSE 問題可以寫成，

$$\min_{g(X)} E_{XG}\left[L(G, g(X))\right] \tag{5}$$

由於是類別資料的關係，其輸出群組變數改寫為 G，預測群組寫成 $g(X)$，兩者的誤差以「Loss function」$L(G, g(X))$ 取代原先的平方差。當 $L(G, g(X))$ 定義為

$$L(G, g(X)) = \begin{cases} 0 \ \text{ if } \ G = g(X) \\ 1 \ \text{ if } \ G \neq g(X) \end{cases}$$

式 (5) 的最佳解為

$$\hat{g}(X) = g_k \ \text{ if } Pr(g_k|X = \mathbf{x}) = \max_{g \in G} Pr(g|X = \mathbf{x}) \tag{6}$$

其中 g_k 代表第 k 個群組（group），G 是所有群組的集合。這個結果說明：當輸入值為 **x** 時，其所屬群組的 MMSE 預測為

[4]當然也可以嘗試自 $P(Y|X = \mathbf{x})$ 中抽取適量的樣本並計算其均值。

[5]請注意：在解決問題的過程中，我們不斷的退讓，也就是不斷地引入誤差。不過，唯有如此，才能繼續前進。

「在所有的群組中，群組機率密度函數在 **x** 處的值為最大者」

又稱為貝式分類器 Bayes classifier。[6]式 (2)、(6) 的推導請參考 [1]。不管哪一種輸出的型態，這裡都使用到「後驗機率」（Posterior Probability）的觀念，也就是當給定輸入變數 $X = \mathbf{x}$，Y(或 G) 值的可能性（機率）。Bayes 之名也來自於此。

群組判別：從式 (6) 中似乎看不出一個明顯的「分界線」方程式，無法像迴歸模型或判別式分析那樣根據方程式畫出一條分界線，更何況機率密度函數 $Pr(G|X = \mathbf{x})$ 也是未知。不過如迴歸模型應用在類別資料上，當假設兩個群組的輸出為 0（群組 g_1) 與 1（群組 g_2) 時，式 (4) 可以當作式 (6) 的估計式，並配合下列的群組判別式，

$$\mathbf{x} \in \begin{cases} g_1 \ \text{ if } \ \hat{y} \leq 0.5 \\ g_2 \ \text{ if } \ \hat{y} > 0.5 \end{cases} \tag{7}$$

式 (4) 的 $\hat{y} \leq 0.5$ 相當於式 (6) 的 $Pr(g_1|X = x) > Pr(g_2|X = x)$。這個方法的表現到底如何？比起迴歸模型或同樣從機率觀點出發的判別式分析（LDA、QDA）好或壞？有什麼缺點？有什麼限制？只要把程式寫出來，拿幾組資料實際來測試一番便知分曉。[7]

2 練習

範例 1. 舉模擬資料 la_3.txt 為例（從網頁下載的，資料來源 [1])，圖 1 展示群組資料利用 K Nearest-Neighbor method 切割空間的結果。試著按下列的說明繪製圖 1。

由於 K Nearest-Neighbor method 並沒有定義出一個分界線的方程式，無法在所在的空間明確地畫出群組界線，不過可以如圖 1 的作法，在一定範圍的空間內，將空間等份成格子狀 (grids)，每個格子的座標代表一個資料值 **x**，將每一個座標點當作新資料一樣的拿出來判斷其類別（大部分的格子點都不在已知的資料

[6]這個結果看起來似乎是「廢話」，因為當後驗機率 $Pr(g|X)$ 未知時，如何知道當 $X = \mathbf{x}$ 時的群組機率密度函數值呢？不過這卻是很重要的理論，它說明在期望預測值與實際值的誤差最小的前提下，最好的預測值是來自在 $X = \mathbf{x}$ 處有最大後驗機率的群組。

[7]百聞不如一見，對一個方法的優劣評估，甚至對其理論的理解，往往要透過實驗才能掌握。因此寫程式做實驗非常重要，特別對於初學這個方法的讀者。

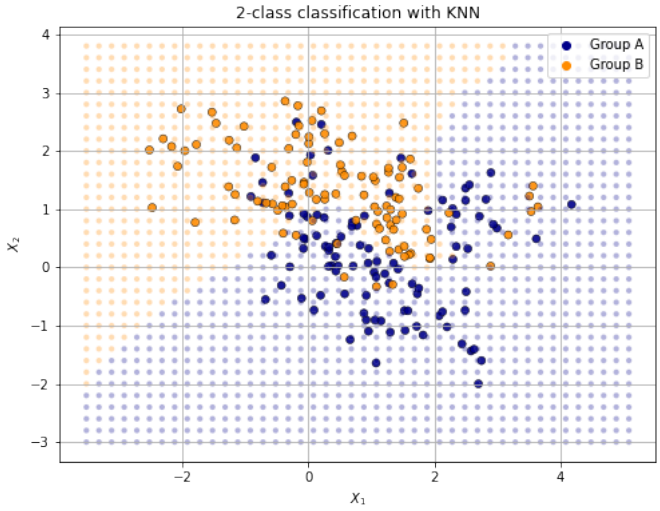


圖 1: 群組資料與 KNN 空間分割（$K = 15$）

裡），依式 (4) 與 (7)，為每個座標點依其群組判斷劃上不同的符號或顏色。格子狀的粗細與組別符號決定了畫出來的感覺，不妨試試看不同的格子密度與群組符號或顏色。

式 (4) 的估計式中需要找出「最靠近 $\mathbf{x}$ 的 K 個已知資料」，這個「靠近」的測量方式可以採用歐式距離（Euclidean Distance）。假設 $\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_N}$ 為 N 個已知資料（含群組別），$\mathbf{x}$ 為空間中某個待判別群組的資料，程式中需要計算 $\mathbf{x}$ 與所有已知資料的距離，再從中選取最靠近的 K 筆資料，最後再將這 K 筆資料的群組值（0 或 1）平均起來，[8]即為式 (7) 中的 $\hat{y}$ 值，程式片段如下

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

data_dir = '../Data/'
D = np.loadtxt(data_dir + 'la_3.txt', comments='%')
X = D[:, 0:2]
y = D[:,2].astype('int') # convert to integers
n = len(y)
cmap_bold = ['darkblue', 'darkorange']
Group_name = np.array(["Group A", "Group B"])

plt.figure(figsize=(8, 6))
sns.scatterplot(x = X[:, 0], y = X[:, 1], \
    hue = Group_name[y], palette = cmap_bold, \
        alpha = 0.9, edgecolor = "black")
```

[8]群組值（0 或 1）的平均與 0.5 作比較，相當於比較 0 與 1 的群組個數多寡。

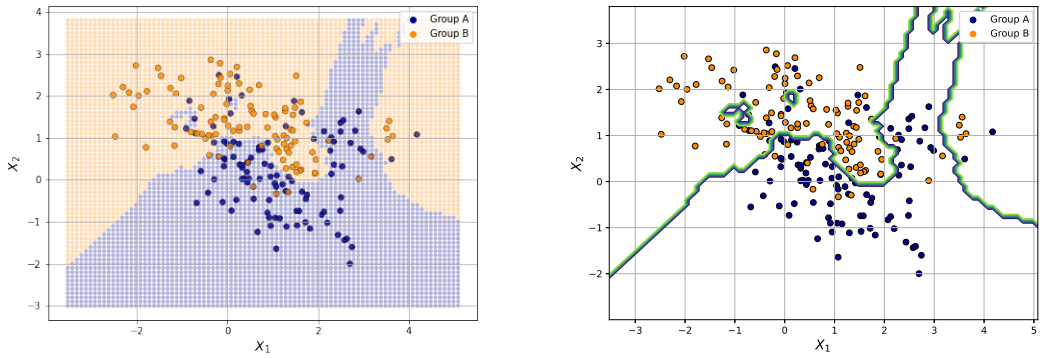
```
# KNN learning
K = 15
intrvl = 0.2  # grid interval
x_min, x_max = X[:,0].min() - 1, X[:,0].max() + 1
y_min, y_max = X[:,1].min() - 1, X[:,1].max() + 1

xx, yy = np.meshgrid(np.arange(x_min, x_max, intrvl), \
    np.arange(y_min, y_max, 0.1))  # grid points: matrices
z = np.zeros(xx.size) # a vector for KNN predictions

for i in range(xx.size) :
    tmp = np.tile([xx.ravel()[i], yy.ravel()[i]], (n, 1))
    d = np.linalg.norm(tmp - X, axis = 1) # n distances
    idx = np.argsort(d) # sorting K distances
    z[i] = np.mean(y[idx[:K]]) # average K sorted y-values

z = [0 if i < 0.5 else 1 for i in z]
sns.scatterplot(x = xx.ravel(), y = yy.ravel(), size = 2,\
        markers = '.', palette = cmap_bold, hue = z, \
                alpha = 0.3, legend = False)
```

程式中的矩陣 X 與向量 y 變數分別代表原始資料與其群組別。為了擷取適當的空間範圍來呈現分群現象，使用了原始資料的最大與最小值。範圍界定後，利用 np.meshgrid 產生 X-Y 平面的座標矩陣，xx 與 yy，為了計算每一個座標點與所有原始資料的距離，將 xx 與 yy 拉成向量，再透過一個迴圈，逐一計算並取最近的 K 個點的 y 值的平均。最後根據平均值的大小分群並繪製散佈圖。

當程式完成後，讀者可以自行調整 K 值，看看這個控制鄰近範圍的值對空間分群的效果。譬如，圖 2 (a) 展現 K=5 的空間分群結果，同時也調整繪圖的密度，呈現不同的風貌。圖 2 (b) 則是透過繪圖技巧呈現出中間那條不規則的 KNN 分界線。圖 2 (b) 繪製分界線的程式碼如下：

```
Z = np.reshape(z, xx.shape) # reshape z to a matrix
plt.contour(xx, yy, Z)
```

指令 plt.contour 是畫立體圖的等高線，因其中函數值 Z 非 0 即 1，於是產生如圖 2 (b) 的效果（那條邊界線其實是懸崖的邊緣）。另，也刻意只標示出高度為 0.5 的等高線，而這條等高線剛好就是 0 與 1 之間的界線。讀者可以試著用 plt.contour(xx, yy, Z, levels = [0.5]) 看看畫出甚麼效



(a) 空間顏色分群　　(b) 分界線

圖 2: 群組資料與 KNN 空間分割（$K = 5$）

果。此外也可以直接繪製立體圖，看看這個非 0 即 1 的 Z 函數的立體長相。

範例 2. 利用 `sklearn.neighbors` 套件的 KNN 模組指令 `KNeighborsClassifier` 做擬合（`fit`）與預測 (`predict`)，並繪製圖 1 與圖 2。

`KNeighborsClassifier` 建立一個 KNN 模組，其下的方法 `fit` 其實沒做什麼計算，只是記錄資料。實際做 KNN 距離計算分群的方法是 `predict`。[9]以下程式碼執行如前一個範例的繪圖，如圖 3。此外，也直接使用指令 `score` 計算分群的準確率。

```
K = 15
weights = 'uniform'
Knn = neighbors.KNeighborsClassifier(K, weights = weights)
Knn.fit(X, y)
trainingErr = 1 - Knn.score(X, y)
x_min, x_max = X[:,0].min() - 1, X[:,0].max() + 1
y_min, y_max = X[:,1].min() - 1, X[:,1].max() + 1

xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),\
    np.arange(y_min, y_max, 0.1))

z = Knn.predict(np.c_[xx.ravel(), yy.ravel()])
Z = z.reshape(xx.shape)
cmap_light = ListedColormap(['cornflowerblue', 'orange'])
plt.contourf(xx, yy, Z, cmap = cmap_light, alpha = 0.3)
plt.title('Training␣error␣=␣%.4f␣for␣K␣=␣%i' % (trainingErr,K))
```

[9] `sklearn` 安排指令 `predict` 為 Machine Learning 套件內所有的分群演算法做分群預測。

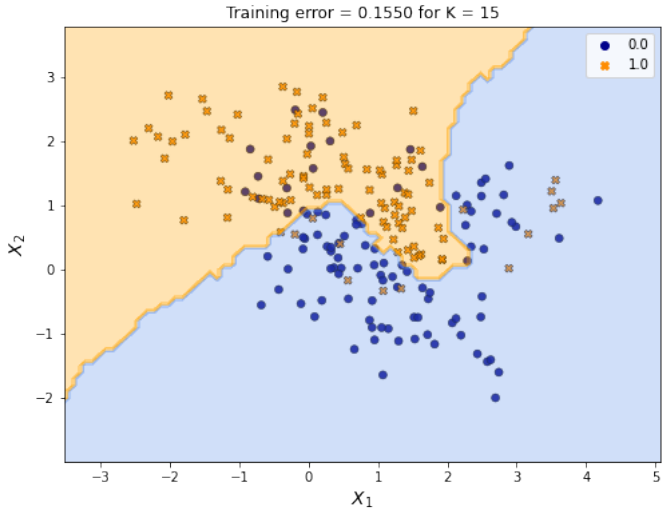


圖 3: KNN 空間分割（$K = 15$）與等高線圖的特殊呈現

範例 3. 想知道 KNN 對於三個群組資料的分群能力，請自行生成資料並進行測試。譬如，假設三個群組的中心點與共變異矩陣分別為：

$$\mu_1 = \begin{bmatrix} 0.5 \\ -0.2 \end{bmatrix}, \mu_2 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \mu_3 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

$$\Sigma_1 = \begin{bmatrix} 2 & 0.3 \\ 0.3 & 0.5 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

群組大小分別為 $n_1 = 200, n_2 = 200, n_3 = 200$。

圖 4 呈現 KNN 在 `K = 5` 的表現。至於分群誤判率及與其他學習器的比較，需要讀者進一步模擬各式情境的資料，如群組間較分散或集中的、群組大小不一的、群組內共變異不同的...。

另，生成模擬資料的函數經常被其他程式使用，因此可以當成共用的函式庫，被所有程式呼叫。譬如生成服從多變量常態的群組資料的函數名為 `mvn_multiclass_data`，而這個函數程式碼所在的程式名為 `Lib_GenData.py`，相對路徑為`../Lib/`。本範例程式開頭必須先取得路徑（path），如下列程式碼前兩行，如此才能宣告第三行取得該函數。

```
import sys
sys.path.append('../Lib/')
```

```
from Lib_GenData import mvn_multiclass_data

n = [200, 200, 200] # sample size for each group
mean = np.array([[0.5, -0.2], [2, 2], [-1, 2]])
cov = np.array([[2.0, 0.3], [0.3, 0.5], \
        [1.0, 0.], [0., 1.], [1.0, 0.], [0., 1.]])
X, y = mvn_multiclass_data(mean, cov, n)
```

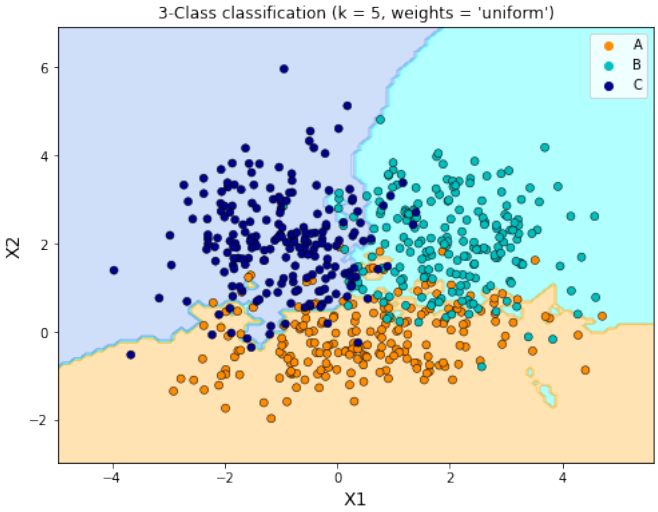


圖 4: KNN 對三個群組資料的分群能力（K=5）

3 觀察與延伸

1. KNN 做出來的空間切割並非直線，幾乎都是不規則形狀。換句話說，分界線是很複雜的非連續函數。
2. K 值的選擇對於空間的切割有重大的影響，對新資料作分類預測時當然也會不同。試著嘗試不同的 K 值，看看切割出來的空間有何不同？
3. 採用 KNN 並不是在找一條空間的分界線，相反的它是比較區域型（範圍較小）的，在一個小區域的空間中找界線，這個區域的大小由 K 及資料的分佈情形來決定。
4. 如果將期望值的估計式 (4) 改為 $\hat{y} = Median\left(y_i | \mathbf{x}_i \in N_k(\mathbf{x})\right)$，結果會變好還是變壞？做做看。
5. 式 (4) 的估計值將鄰近點視為等值影響，對某些區域來說並不「公平」，如果依距離之遠近改變其權重，是否能提供更平滑的分界線？請試試看依常態分配的機率值來權衡其重要性。

4 習題

1. 推導出式 (2)，其中

$$\begin{aligned} E_{XY}\left(Y-f(X)\right)^2 &= \int\int (Y-f(X))^2 Pr(X,Y)dXdY \\ &= \int\int (Y-f(X))^2 Pr(Y|X)Pr(X)dXdY \\ &= E_X E_{Y|X}((Y-f(X))^2|X) \end{aligned}$$

2. 推導出式 (6)，其中

$$E_{XG}\left[L(G,g(X))\right] = E_X \sum_{k=1}^{K} L(g_k, g(X))Pr(g_k|X)$$

3. 以 la_3.txt 資料為例，利用本章介紹的 KNN 分群方法，依 $K = 15, 10, 5, 1$ 各畫出分界線（共四張圖）。

4. 任選一個 K 值，將式 (4) 的 Average 以 Median 取代，畫一張以顏色及符號分組的圖，另一張只畫分界線。

5. 不同的 K 值其分割組別的能力不同。要判斷分割的精準度，可以將所有的擬合值與原始值做比較，看看整體的誤差（譬如所有誤差值的平方合）就可以看出精準度。請根據 K 值的不同，從 K =15,10,5,1 分別計算其誤差平方合，並畫一張圖來表示其趨勢。

6. 從上題的精準度問題來看，實際上精準度愈高並不代表其分辨新資料的能力愈好，針對不同的資料，會有一個最佳的 K 值，其對新資料的分辨能力最好。請自行產生一組資料（俗稱 Training Data）來決定分界線，再產生另一組資料作為測驗這個分界線的測試資料（稱為 Testing Data），一樣採誤差平方合來作為優劣的依據。畫一張圖含兩條線，一條是根據 Training Data 所產生的誤差，另一條是根據 Testing Data 產生的。橫軸都是 K 值（譬如 $K = 15, 14, \cdots, 1$）。請注意，這裡的資料可以依下一題的方式產生。

7. KNN classifier（式 (4)、(7)）其實是 Bayes classifier（式 (6)）的近似版，當條件式機率分配 $P(X|G)$ 已知時，Bayes optimal boundary 可以被準確的計算出來，如圖 5 所示。

   計算 Bayes optimal boundary 的前提是隨機資料的產生機率密度函數（Generating Density）必須已知。圖 1 的資料（mix）來自兩組混合常態（Normal Mixtures）的母體，每組各由 10 個雙變量（Bivariate）常態母體組成，如

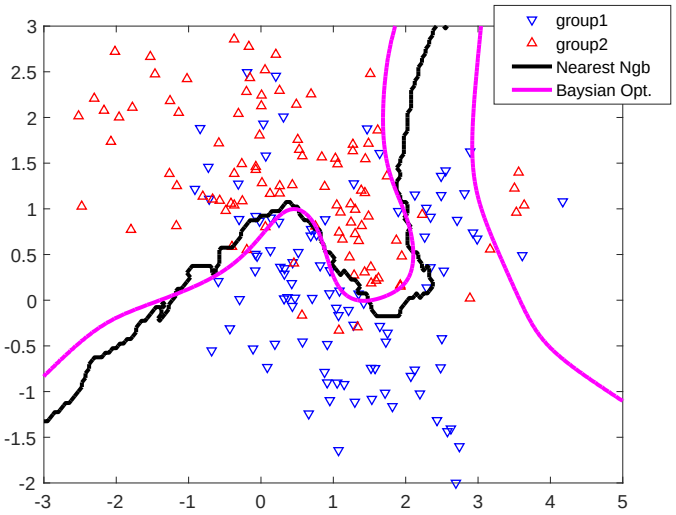


圖 5: Bayes optimal boundary

$$\text{第一組} \qquad P(X|G=1) = \sum_{k=1}^{10} \alpha_k \phi(X, \mu_k^1, \Sigma_k)$$

$$\text{第二組} \qquad P(X|G=2) = \sum_{k=1}^{10} \alpha_k \phi(X, \mu_k^2, \Sigma_k)$$

其中，混合比例 α_k 皆為 $\frac{1}{10}$，共變異矩陣 Σ_k 皆為 $I/5$，第一組常態分配的均值 μ_k^1 由一個雙變量常態 $N((1,0), I)$ 產生 10 個，另一組由 $N((0,1), I)$ 產生 10 個。參考資料 mix 亦包含這些 mean.mat 的樣本（變數名稱 x_mean）。Bayes optimal boundary 為

$$Pr(G = g_1|X) = Pr(G = g_2|X) \tag{8}$$

利用貝氏定理並將上述的假設代入 (8)，可以得到這條線的數學式子，用 MATLAB 畫等高線圖，即為圖 5 所示。從式 (8) 的角度來看，KNN 對兩個群組的分割可說是接近完美的演出。試著將這條理論上最好的分界線畫出來。

References

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