NYCU Machine Learning HW2

312605001

機器人碩士 歐庭維

1. 分類器實現

```python
class LDA():
    def __init__(self):
        self.mean_class_1 = None
        self.mean_class_2 = None
        self.C1 = 1
        self.C2 = 1

        self.w_T = None
        self.b = None


    def fit(self, x, y):
        class_1 = x[y == 1]
        class_2 = x[y == 0]

        n1 = class_1.shape[0]
        n2 = class_2.shape[0]

        p1 = n1/(n1+n2)
        p2 = n2/(n1+n2)

        self.mean_class_1 = np.mean(class_1, axis = 0)
        self.mean_class_2 = np.mean(class_2, axis = 0)

        covariance_1 = np.cov(class_1, rowvar=False)
        covariance_2 = np.cov(class_2, rowvar=False)
        covariance = covariance_1*p1 + covariance_2*p2

        self.w_T = (self.mean_class_1 - self.mean_class_2).T @ np.linalg.inv(covariance)
        self.b = -0.5 * self.w_T @ (self.mean_class_1 + self.mean_class_2) - np.log((self.C1 * p2) / (self.C2 * p1))

        self.w_T = np.round(self.w_T, 2)
        self.b = round(self.b, 2)

        print('[!] training weight vector : ', self.w_T, ' training bias : ', self.b)


    def LDA_decision_function(self, x, y_true):
        correct_predictions = 0
        x = np.array(x)

        for i in range(len(x)):
            x_col = np.array([[x[i][0], x[i][1]]]).T

            g = self.w_T @ x_col + self.b
            predicted_class = 1 if g > 0 else 0

            if predicted_class == y_true[i]:
                correct_predictions += 1

        accuracy = correct_predictions / len(y_true)

        return accuracy
```

2. 利用 2-fold cross validation 推估 LDA 在 C1/C2 = 1 時之二元分類分類率

```
tingweiou@parallels:~/course_ws/machine_learning_2023_fall/hw2$ python3 312605001_hw2.py
training w =  [ -2.09 -10.46] training b =  28.1 testing accuracy 94.0 %
testing w =  [-3.73 -7.85] testing b =  31.08 training accuracy 94.0 %
average accuracy =  94.0 %
```

- 使用 training_data 計算出之 weight vector = [-2.09, -10.46]
- 使用 training_data 計算出之 bias = 28.1
- 使用 training_data 訓練出之模型套用到 testing_data 得到之分類率為 94%
- 使用 testing_data 計算出之 weight vector =[-3.73, -7.85]
- 使用 testing_data 計算出之 bias = 31.08
- 使用 testing_data 訓練出之模型套用到 training_data 得到之分類率為 94%
- 兩者之平均分類率為 94%

- 問題討論：
  1. 試著調整 C1 & C2 的比值

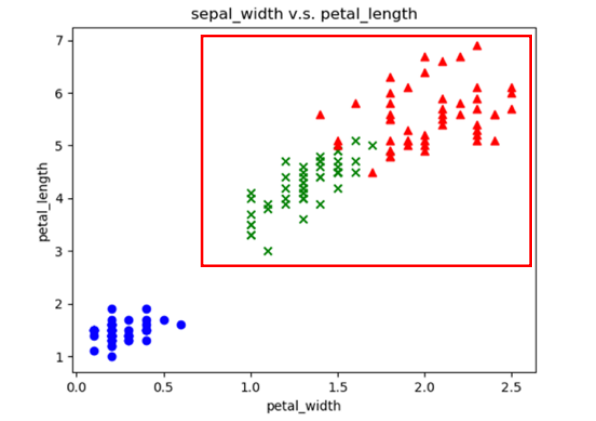
     線性判別分析（LDA）是一種用於降維和分類的統計方法。其主要目標是找到一種線性特徵組合，以區分數據集中的兩個或多個類別。而在二維數據中，可以透過找到一條超平面（LDA Decision Function），透過將測試資料代入這條 function 中，以判斷該資料位於超平面之上方（>0 ）或下方（<0），並對其做分類，這樣看來，如何去找到最適合之 LDA Decision Function，便是首要的問題，我們可以透過調整以下的 c1 & c2 的比值，對模型微調（紅框處）

$$D(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

$$\mathbf{w}^T = (\mathbf{m_1} - \mathbf{m_2})^T\boldsymbol{\Sigma}^{-1}$$

$$b = -\frac{1}{2}(\mathbf{m_1} - \mathbf{m_2})^T\boldsymbol{\Sigma}^{-1}(\mathbf{m_1} + \mathbf{m_2}) - \ln\left(\frac{C_1P_2}{C_2P_1}\right)$$

接下來可以從作業一中的分佈圖了解數據的大致分佈如下圖所示：



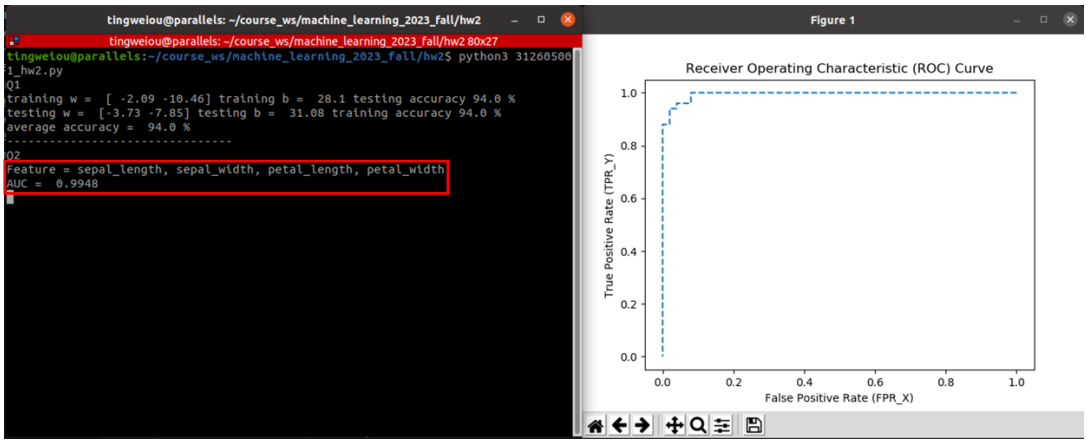
由於兩種特徵有些微樣本交疊在一起，因此可以知道若要以 LDA 方法將其做切分，會無法找到一條超平面將樣本完美的切開，這意味著 LDA 之分類率不可能達到 100%，而接下來我試著調整 c1 & c2 的比例，以找到最佳的分類率，結果如下表所示：

C1 / C2 = 0.8 CR = 95%	Q1 training w = [-2.09 -10.46] training b = 28.32 testing accuracy 96.0 % testing w = [-3.73 -7.85] testing b = 31.3 training accuracy 94.0 % average accuracy = 95.0 %
C1 / C2 = 1.2 CR = 94%	Q1 training w = [-2.09 -10.46] training b = 27.92 testing accuracy 94.0 % testing w = [-3.73 -7.85] testing b = 30.9 training accuracy 94.0 % average accuracy = 94.0 %
C1 / C2 = 0.4 CR = 92%	Q1 training w = [-2.09 -10.46] training b = 29.02 testing accuracy 90.0 % testing w = [-3.73 -7.85] testing b = 32.0 training accuracy 94.0 % average accuracy = 92.0 %
C1 / C2 = 2.6 CR = 95%	training w = [-2.09 -10.46] training b = 27.15 testing accuracy 92.0 % testing w = [-3.73 -7.85] testing b = 30.12 training accuracy 98.0 % average accuracy = 95.0 %

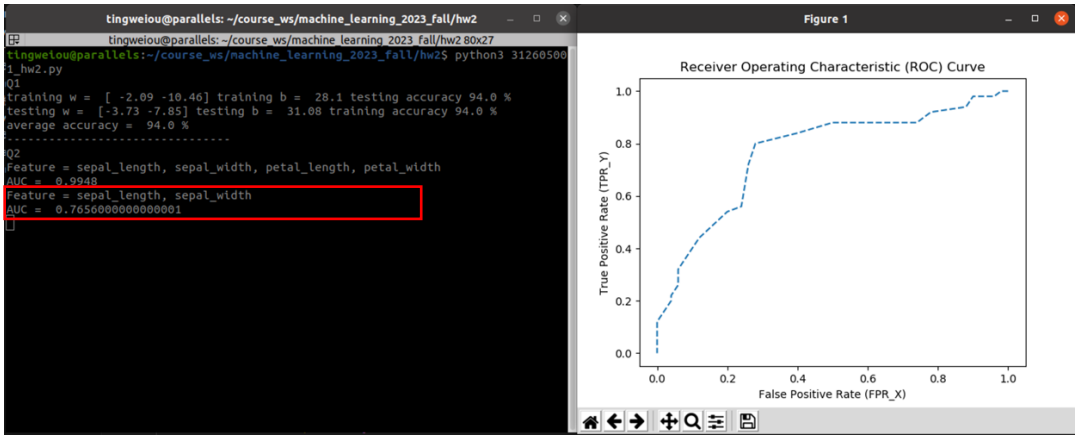
經過不斷調適，得到的平均分類率最好為 95%。

3. 繪製 ROC 與計算 AUC

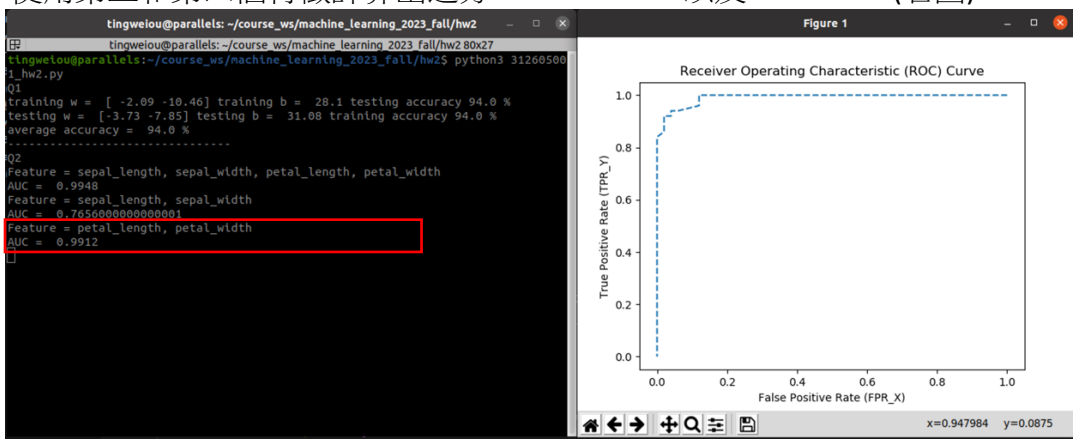
- 使用四個特徵計算出之 AUC= 0.9948 以及 ROC Curve (右圖)



- 使用第一和第二個特徵計算出之 AUC = 0.7656 以及 ROC Curve (右圖)



- 使用第三和第四個特徵計算出之分 AUC = 0.9912 以及 ROC Curve(右圖)

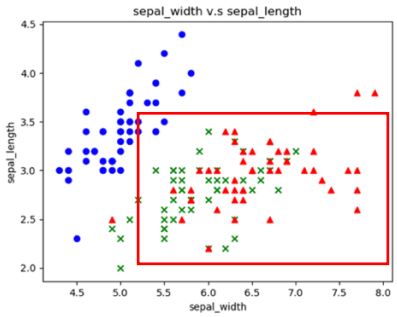
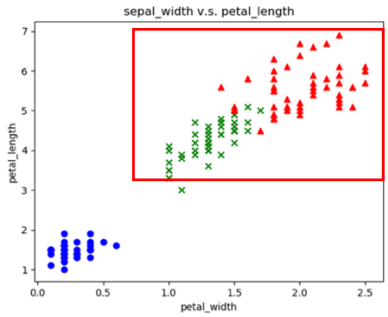


- 問題討論：
  - 討論上述三個不同情形之 AUC，並解釋特徵組合和 AUC 之關聯性？

四個特徵	第一和第二個特徵	第三和第四個特徵
AUC = 0.9948	AUC = 0.7656	AUC = 0.9912

AUC 是基於 ROC（Receiver Operating Characteristic）曲線的下面積，ROC 曲線顯示了不同分類閾值下模型的真正例率（True Positive Rate）和假正例率（False Positive Rate）之間的關係。而一個完美模型的 AUC = 1，一個隨機猜測的模型的 AUC = 0.5。而 ROC 的最左上方代表了最完美的結果（沒有偽陽性及偽陰性），由實驗結果的 ROC 可以觀察到，第一組合第三組的 ROC 非常接近 1，這也代表了其分類效果不錯，而第二組效果則不好。

先看兩兩特徵組合的圖表如下圖：

Sepal length v.s. sepal width	Petal width v.s. petal length
	

可以由樣本的分佈很好的觀察到左圖的兩個樣本幾乎交疊在一起，若要強制使用 LDA 進行分類，則可以預測到其 AUC 會不高，而右圖相對來說兩群較為分散，AUC 會較接近 1。

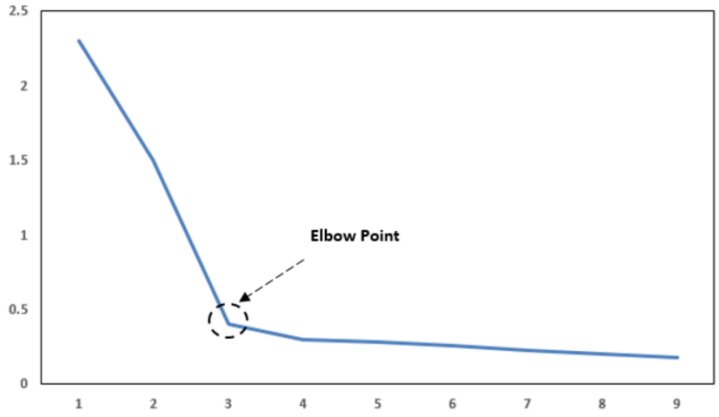
再來討論關於四個特徵與第三和第四個特徵之間的 AUC 比較，可以發現同時考慮四個特徵時，表現是三者中最接近 1 的(較好)，而我推測這是因為同時使用多個特徵可以增加模型的 Robustness，當今天只考慮兩兩特徵的比較，模型會對數據中的離群值（noise）更敏感，而同時考慮四個可以減少這種影響；另外，每個特徵可能包含不同方面的訊息，當使用所有特徵同時考慮時，比較能夠同時利用這些訊息進行分類。

但若是對於一些特徵很多的數據集來說，如果同時考慮全部的特徵，我認為並不是一項好的策略，因為用 LDA 進行數據分類的主要目標在於捕捉低維度的特徵表示方法。若是基於以上觀點來看這個問題，在特徵全

取和取後兩個特徵等兩個 AUC 差不多的前提下，我認為優先選擇取後兩個特徵的策略較佳。

- AUC 是否可以當作特徵比較之量化工具？

  AUC 是一項重要的指標，用來評估二元分類器的性能，通常 AUC 越接近一代表性能越好。在特徵數量少（四個）的前提下，我們可以很輕易的去嘗試每個特徵組合，並繪製出 AUC，以此來大略判斷這些特徵組合的效果如何。但如果面臨到特徵很多的問題，且需要將特徵間的比較作為量化的工具時，單純看 AUC 可能就不是一個好的方法。首先我會將各種不同的特徵的相互組合丟進分類器中取平均，繪製出類似以下的表格：



  其中水平軸代表不同數量的 feature，垂直軸代表性能評估指標，我們可以找到圖中的 elbow point（平衡點），這代表了對應的特徵數量會有較好的表現，以圖中的例子為例，可以知道考慮三個特徵的時候表現會比較好，接下來就可以將所有的三個特徵組合丟入計算 AUC，並取最佳表現得即可。

4. 多類別分類問題
   - 1st fold CR = 96% , 2nd fold CR = 96% , Average CR = 96%

```
tingweiou@parallels:~/course_ws/machine_learning_2023_fall/hw2$ python3 312605001_hw2.py
Q1
training w =  [ -2.09 -10.46] training b =  28.1 testing accuracy 94.0 %
testing w =  [-3.73 -7.85] testing b =  31.08 training accuracy 94.0 %
average accuracy =  94.0 %
--------------------------------
Q2
Feature = sepal_length, sepal_width, petal_length, petal_width
AUC =  0.9948
Feature = sepal_length, sepal_width
AUC =  0.7656000000000001
Feature = petal_length, petal_width
AUC =  0.9912
---------
Q3
fold-1 =  96.0 %
fold-2 =  96.0 %
average =  96.0 %
```

- 問題討論

  One-against-one strategy 是一種用於多類別分類的方法，透過將二元分類器擴展處理多類別問題，而這裡我所使用的二元分類器是 LDA，而在 iris 數據集中，label 分為三種，分別是：Setosa/ Versicolor/ Virginica，對於這種 label 較少的狀況來說，使用 One-against-one strategy 是較好的策略，因為需要訓練的二元分類模型較少（1v1 / 1v3 / 2v3），計算成本相對不高，透過將資料進行兩兩比較，並投票，以決定最終的分類率，而經過 fold1 & fold2 的計算後，得到最終的 average CR = 96%，表現較單一使用 LDA 還佳。