資料探勘期中報告—第九組

鄭璟翰
國立中山大學資工系 113 級
b093040003@student.nsysu.edu.tw

郭晏涵
國立中山大學資工系 113 級
b093040024@student.nsysu.edu.tw

1. 簡介

透過python實作K-Nearest Neighbor以及Linear classifiers、Decision Tree、Random Forest、Multilayer Perceptron(透過scikit-learn完成)分析一個資料集，內容為病人之各項測量值(總共包含八種數據，如 BMI、年齡等)，以及病人是否罹患糖尿病(位於資料最末欄)。各別觀察這五種分類器所產生的結果，計算各個分類器的準確率等表現。藉此分析其結果與不同分類器之間的關係。

流程圖如下**圖1.**。

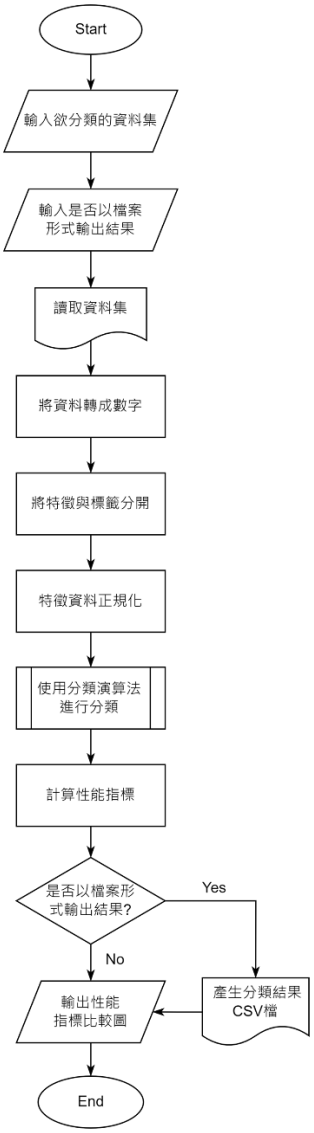

圖1. 分類流程圖

2. 資料整理及輸出

由於不論是 KNN 或是其他分類演算法，在進行分類前，都需要事先進行資料整理、輸入等步驟，因此我們將這個部分整理於 tool.py 這個檔案之中。當需要進行與分類無關的處理時，就會透過呼叫 tool.py 來獲得需要的資料以及輸出分類結果、統計畫面。

2.1 資料整理

包含讀檔案(readData)、將資料轉成數字(toNumber)、分開標籤(splitLable)、正規化(normalize)五個部分。

需要將資料轉成數字的原因是因為將 csv 檔案讀入時，其形態會是字串，因此在計算之前需要將結果轉成數字。

正規化的作法是將資料等比例壓縮到 0~1，OD 表示原本資料，ND 表示正規化後的資料，OMax、OMin 代表原本資料及的最大與最小值。

$$\mathrm{ND} = \frac{\mathrm{OD} - \mathrm{OMin}}{\mathrm{OMax} - \mathrm{OMin}}$$

需要進行正規化的理由是因為透過觀察資料集可以發現，各個欄位的資料分佈並不相同，如懷孕次數的分布範圍在 0~17，而家族病史糖尿病函數的分布範圍 0.078~2.42，以此為單位計算資料之間的距離，那家族糖尿病函數的因素就會被稀釋掉，因此需要進行正規化。只有在實作 KNN 時有進行這個步驟。

另一種作法則是透過標準化(Standardization)將資料是採用 z 分數，μ是原資料集的平均、σ是原資料集的標準差。

$$\mathrm{z} = \frac{\mathrm{OD} - \mu}{\sigma}$$

2.2 結果統計及輸出結果

輸出結果分成三個部分，包括輸出檔案、計算統計結果、以圖表呈現統計結果。

輸出檔案是指將每一筆資料判斷的結果重新以 csv 檔輸出，在原本的檔案最後一欄位加上預測結果，即可藉由比對最後兩欄來觀察每一筆的預測情形。

輸出統計結果包括計算 accuracy、precision、recall、random 四種比例。只有 KNN 會出現 random 這種數據，詳細將於KNN解釋為什麼會有 random percentage 的出現。

表 1. 預測結果與真實情況的組合

		Predicted Class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Accuracy 代表所有情況下判斷正確的比例。

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision 代表判斷得病的情況下，真正得病的正確率，可用於判斷偽陽性的比例。

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Recall 代表為得病的情況下，有多少被正確判斷出來，可用於判斷偽陰性的比例。

3. 分類結果分析

透過 2.1 資料整理，得到各筆資料的內容(data)，以及是否罹患糖尿病(label)，接著分別放入各個分類器後，即可得到每筆資料的結果。

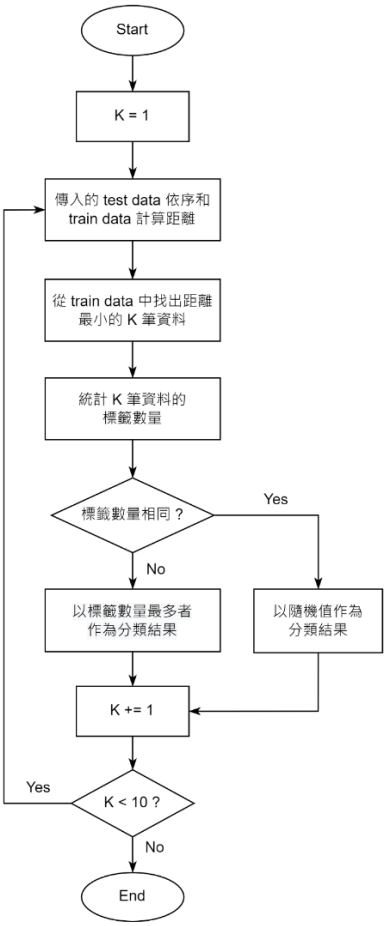

圖2. KNN流程圖

3.1 K-Nearest Neighbor

KNN 是一種基於計算測試資料與訓練資料中距離的分類演算法，藉由找出訓練資料中和測試資料最近的 K 筆來判斷測試資料屬於哪一種分類，判斷流程圖如上**圖 2.**。

找出距離最小 K 筆資料的方法是依照距離分類：找出第 K 筆資料的距離，其中所有小於等於該距離的資料都列入考量。

其中 K 是 hyperparameters，是需要自行調整的參數，而我們選擇將 K 從 1~9 的結果都執行，以比較不同的超參數對於結果的影響。

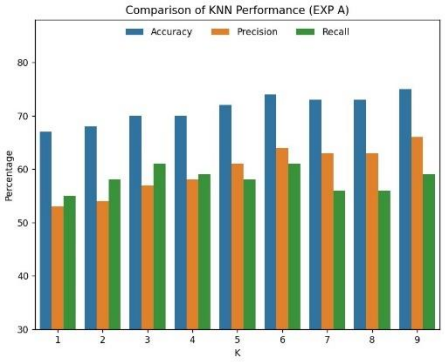

圖3. KNN分類實驗A結果

透過圖3. 可以得知隨著K增加，Accuracy和Precision也會上升；而Recall則是在K在3~6之間達到最高值。(因此可以判斷K=6時是最適合的hyperparameters)

```
k = 1
accuracy: 67%
precision: 53%
recall: 55%
random percentage: 0%
--------------------
k = 2
accuracy: 68%
precision: 54%
recall: 58%
random percentage: 39%
--------------------
k = 3
accuracy: 70%
precision: 57%
recall: 61%
random percentage: 0%
--------------------
k = 4
accuracy: 70%
precision: 58%
recall: 59%
random percentage: 26%
--------------------
k = 5
accuracy: 72%
precision: 61%
recall: 58%
random percentage: 0%
```

```
k = 6
accuracy: 74%
precision: 64%
recall: 61%
random percentage: 14%
--------------------
k = 7
accuracy: 73%
precision: 63%
recall: 56%
random percentage: 0%
--------------------
k = 8
accuracy: 73%
precision: 63%
recall: 56%
random percentage: 17%
--------------------
k = 9
accuracy: 75%
precision: 66%
recall: 59%
random percentage: 0%
```

圖4. KNN分類實驗A詳細情況

圖4. 中的random percentage所代表的是以亂數分配結果的機率，會有隨機分配的情況是因為最近的K筆訓練資料當中，可能有兩種以上最多的情形發生，造成無法分類的情況。以實驗結果而言，因為實驗結果有陽性和陰性兩種結果，因此

當K是偶數時，可能會出現陽性和陰性兩種結果相同的狀況，此時就已亂數分配結果；但若K屬於奇數時，不可能出現陽性和陰性兩種結果相同的情況，所以random percentage都是0%。

若K為2的話，有將近2/5的測試資料是隨機分配。透過圖4.可以觀察隨著K的增加，random percentage有降低的趨勢，原因是參考越多訓練資料，能夠成功分類的可能就越高。

圖5. 圖6. 為實驗B的實驗結果。可以觀察到K=6有Accuracy和Precision有最大值，然而Recall和實驗A比較起來都比較低，推測認為是因為實驗B的資料集當中，陽性比陰性的比例還要少。

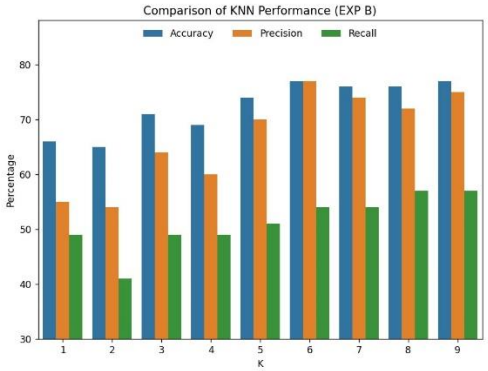

圖5. KNN分類實驗B結果

```
k = 1
accuracy: 66%
precision: 55%
recall: 49%
random percentage: 0%
--------------------
k = 2
accuracy: 65%
precision: 54%
recall: 41%
random percentage: 40%
--------------------
k = 3
accuracy: 71%
precision: 64%
recall: 49%
random percentage: 0%
--------------------
k = 4
accuracy: 69%
precision: 60%
recall: 49%
random percentage: 28%
--------------------
k = 5
accuracy: 74%
precision: 70%
recall: 51%
random percentage: 0%
```

```
k = 6
accuracy: 77%
precision: 77%
recall: 54%
random percentage: 15%
--------------------
k = 7
accuracy: 76%
precision: 74%
recall: 54%
random percentage: 0%
--------------------
k = 8
accuracy: 76%
precision: 72%
recall: 57%
random percentage: 17%
--------------------
k = 9
accuracy: 77%
precision: 75%
recall: 57%
random percentage: 0%
```

圖6. KNN分類實驗B詳細情況

3.2 其他分類器

第二部分我們使用 scikit-learn 來完成以下四種分類器 Linear classifiers (SGD)、Decision Tree(DT)、Random Forest(RF)、Multilayer Perceptron(MLP)。

A. Linear Classification

第一個是透過 Stochastic Gradient Descent 方法來訓練的線性分類器，所使用的 loss function 為 Logistic。SGD 是隨機梯度下降法，讓 loss function 達到最小值以減少誤差。和 GD 比較起來，SGD 是隨機抽取樣本進行計算，計算時間較快。

$$\text{Log} = \sum_{(x,y)\in D}^{n} -\text{xlogy} - (1-\text{x})\log(1-\text{y})$$

因為訓練出來的是一個線性分類器，然而資料集是一個八個維度的資料，因此透過**圖 7.**可以很明顯地觀察到 Linear classifiers 的測試結果並不穩定；而且經過多次測試會發現，透過 SGD 所訓練出來的結果每次都不一樣，偶爾會出現 recall 或 precision 特別高的情況，推測的原因是因為剛好訓練出來的分類器對應到結果上。

B. Decision Tree

DT 是根據訓練資料產生一個將資料分類的 tree，方法是透過評估這個分類規則的好壞來決定是否採用這個分類規則，分類規則的好好壞以 Gini Impurity 高低決定，接著對分類完的子類別在分類，直到最大深度為止。

$$\textbf{Gini Impurity} = 1 - \sum_{j} p^2$$

C. Random Forest

RF 則是將原本的樣本抽樣產生較小的樣本，每個樣本各自產生 DT，最後將各個 DT 依照決定分類結果。因為 RF 是透過 DT 為基礎，因此可以觀察到 RF 表現筆 DT 還要好一些。

D. Multilayer Perceptron

MLP 是依照類神經網路所建構的分類器，可以設定隱藏層的數目、迭代的世代以及 activation function。經過測試，iter = 1000，activation = sigmoid function 準確率最高。

$$\textbf{Sigmoid}(\text{x}) = \frac{1}{1+e^{-x}}$$

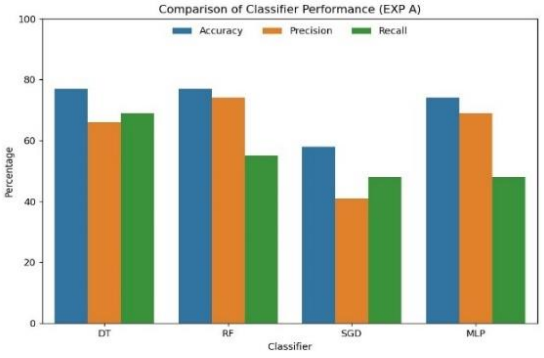

圖7. Classifier分類實驗A結果

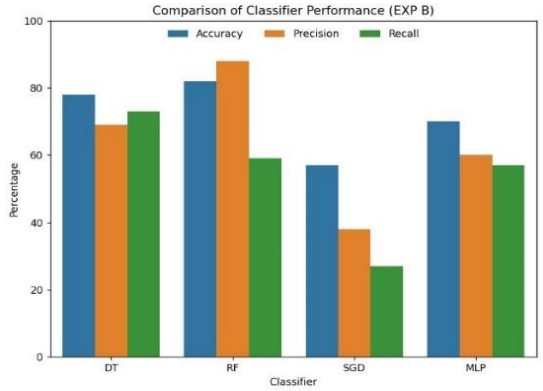

圖8. Classifier分類實驗B結果

```
DT  :  decision_tree    DT  :  decision_tree
accuracy: 77%           accuracy: 78%
precision: 66%          precision: 69%
recall: 69%             recall: 73%
--------------------    --------------------
RF  :  random_forest    RF  :  random_forest
accuracy: 77%           accuracy: 82%
precision: 74%          precision: 88%
recall: 55%             recall: 59%
--------------------    --------------------
SGD  :  stochastic_gra  SGD  :  stochastic_gra
accuracy: 58%           accuracy: 57%
precision: 41%          precision: 38%
recall: 48%             recall: 27%
--------------------    --------------------
MLP  :  multilayer_per  MLP  :  multilayer_per
accuracy: 74%           accuracy: 70%
precision: 69%          precision: 60%
recall: 48%             recall: 57%
```

圖9. Classifier分類詳細結果

4. 結論

透過以上的實驗結果可以發現：

(1)、KNN分類演算法中的超參數K若太小將會影響準確性很低，且依資料特性，會出現random case。

(2)、其他分類演算法中，以 RF 的分類結果最為準確，其次分別是 DT、MLP，LC(SGD)最為不穩定。

(3)、比較各分類的 precision 和 recall 可以發現，大部分的分類器 precision 都大於 recall，顯示相較於正確判斷全部的得病病例，分類器在得病情況下的正確率較高。推測的原因是因為資料集中陰陽的各數比例並不平均。

參考文獻

[1]. Scilit-learn Classifier comparison
https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

[2]. seaborn.barplot
https://seaborn.pydata.org/generated/seaborn.barplot.html

[3]. os — Miscellaneous operating system interfaces
https://docs.python.org/3/library/os.htmlhttps://seaborn.pydata.org/generated/seaborn.barplot.html

[4]. Stackoverflow- Transpose list of lists
https://stackoverflow.com/questions/6473679/transpose-list-of-lists

[5]. pandas.DataFrame
https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

[6]. matplotlib.pyplot
https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.html

[7]. Scilit-learn - Random Forest Classifier
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[8]. Scilit-learn - Decision Tree
https://scikit-learn.org/stable/modules/tree.html

[9]. Scilit-learn - SGDClassifier
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html