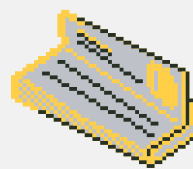
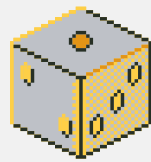
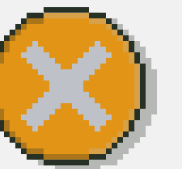
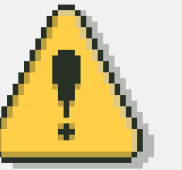
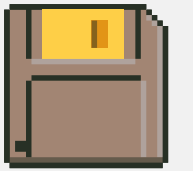


DATA MINING

-Midterm Project

 Group 12

B104020024 蔡尚宸
B103025027 林瀚坪
B103040015 陳承新
B103040061 段向生
B103040063 蕭維亨

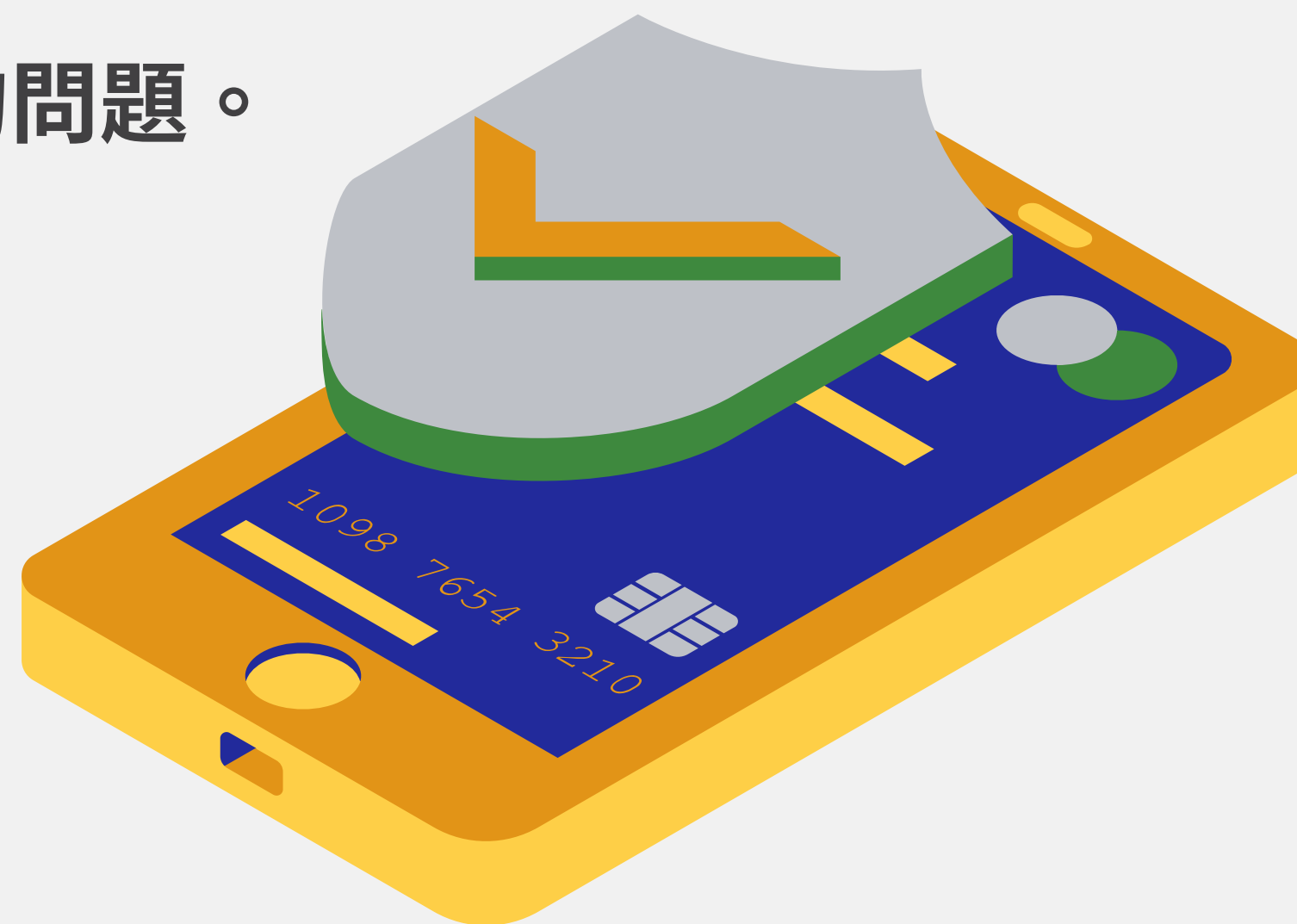


一、摘要

本報告將從KNN開始介紹，介紹幾種資料分類演算法；

再來分享我們設計的程式以及在設計時所遇到的問題。

最後我們會對得出的結果進行分析。



二、分類演算法 簡介

KNN演算法, Random Forest, LVQ

KNN

K-近鄰演算法

一種監督式學習演算法，它可以用於分類和回歸問題。

Random Forest

一種集成學習算法，使用多個決策樹對數據進行建模，然後將它們的預測結果合併來得出最終的預測結果。

LVQ

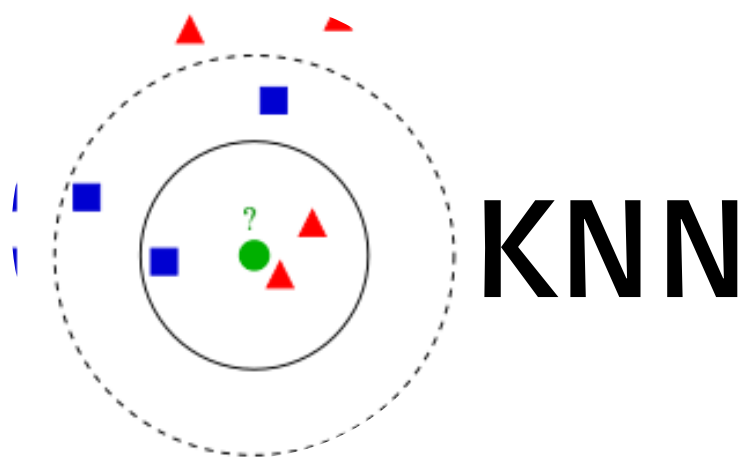
一種監督式學習算法，
在過程中進行即時的調整，讓各類別的代表點趨近最佳值。

不同分類法的比較

步驟

缺點

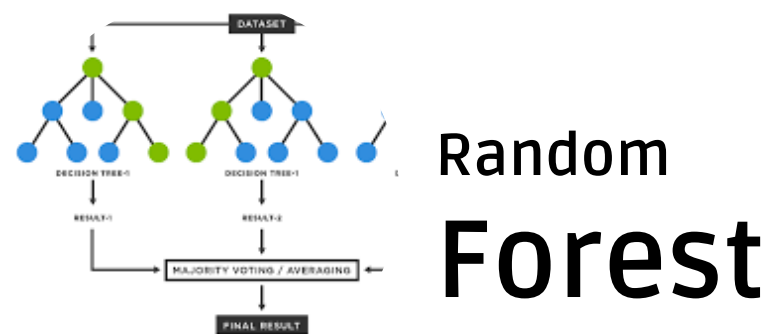
優點



- 確定鄰居數（即K值）。
- 計算測試樣本與訓練樣本之間的距離。
- 找到與測試樣本最相似的K個訓練樣本。
- 決定這K個訓練樣本中最常見的類別，
- 並將其分配給測試樣本的類別。

當數據集較大時，
計算成本較高。
對數據集中的噪聲和
異常值比較敏感

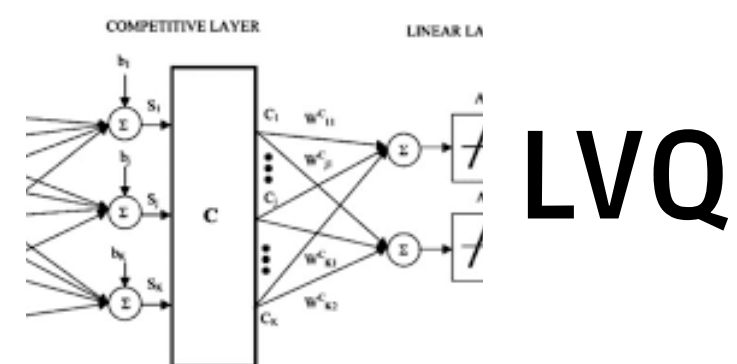
簡單易懂且易於實現



- 從數據集中隨機選取一部分樣本，用於訓練第一棵決策樹。
- 直到建立了所需數量的決策樹。
- 將它們的預測結果進行合併，最終根據合併後的結果進行預測。

難以解釋其內部運作
方式，因為它是由多
個決策樹組成的。且
需要較多空間。

在處理大型數據集時，
速度快
能夠處理高維數據集，並
且可以檢測出重要的特徵

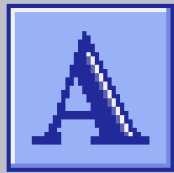


- 計算樣本與每個權重向量之間的距離。
- 確定最近鄰分類器。
- 如果最近鄰分類器與樣本的真實標籤相同則將其向樣本的方向移動。
- 反之則相反

需要進行參數調
整，並且不易擴展
到多類別分類問
題。

具有很好的可解釋性，
可以更好地理解
模型的內部運作方式

[Back to previous Page](#)



三、程式部分

本次使用的資料集有8個attribute
用來train的資料分別有567及668筆



KNN-處理步驟

1. 讀檔後，將數據轉換成list

2. 歸一化

3. 切割數據

```
#將數據轉換成list
list_train_a = [list(train_data_a[x]) for x in feature]
list_test_a = [list(test_data_a[x]) for x in feature]
list_train_b = [list(train_data_b[x]) for x in feature]
list_test_b = [list(test_data_b[x]) for x in feature]

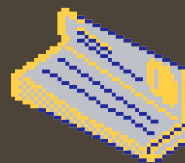
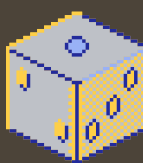
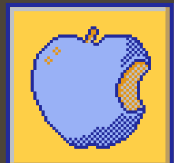
#歸一化
for i in range(8):
    max_train_a = max(list_train_a[i])
    min_train_a = min(list_train_a[i])

    for j in range(len(list_train_a[i])):
        list_train_a[i][j] = auto_norm_data( list_train_a[i][j], max_train_a, min_train_a)

    for j in range(len(list_test_a[i])):
        list_test_a[i][j] = auto_norm_data( list_test_a[i][j], max_train_a, min_train_a)
for i in range(8):
    max_train_b = max(list_train_b[i])
    min_train_b = min(list_train_b[i])
    for j in range(len(list_train_b[i])):
        list_train_b[i][j] = auto_norm_data( list_train_b[i][j], max_train_b, min_train_b)
    for j in range(len(list_test_b[i])):
        list_test_b[i][j] = auto_norm_data( list_test_b[i][j], max_train_b, min_train_b)

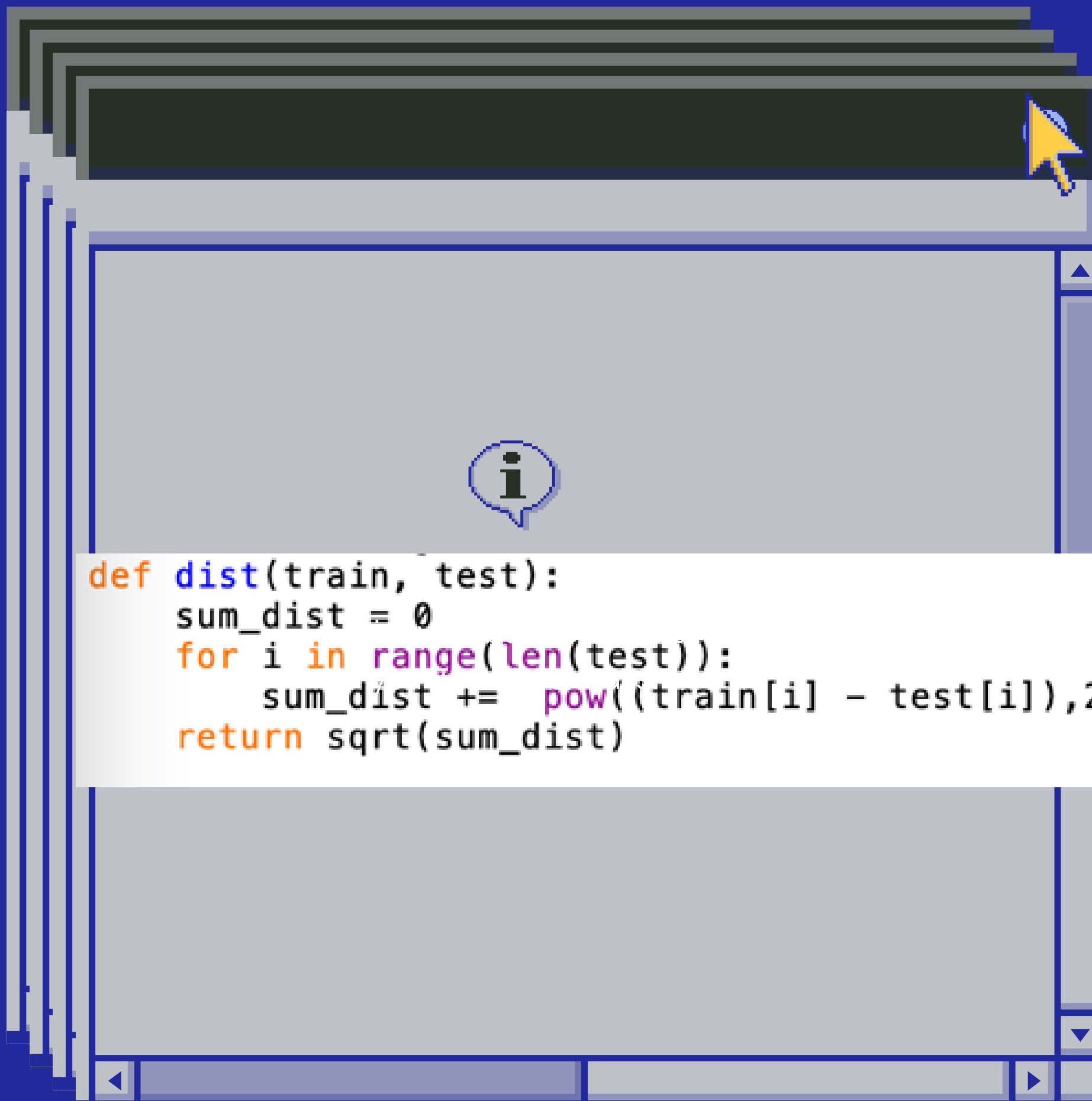
#方便查看數據
"""
new_train = pd.DataFrame({feature[x]:list_train_a[x] for x in range(8)})
new_test = pd.DataFrame({feature[x]:list_test_a[x] for x in range(8)})
print(new_train.head())
print(new_test.head())
"""

#切割數據
single_train_a = [[list_train_a[x][y] for x in range(8)] for y in range(len(list_train_a[0]))]
single_test_a = [[list_test_a[x][y] for x in range(8)] for y in range(len(list_test_a[0]))]
single_train_b = [[list_train_b[x][y] for x in range(8)] for y in range(len(list_train_b[0]))]
single_test_b = [[list_test_b[x][y] for x in range(8)] for y in range(len(list_test_b[0]))]
```

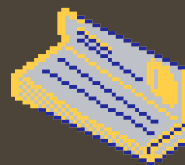
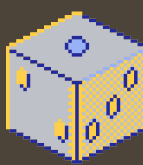
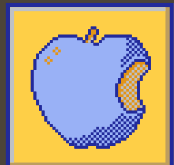


KNN-處理步驟

4. 計算測試資料和每一筆
要訓練資料的距離，使用
的是歐基里德距離
(Euclidean distance)。



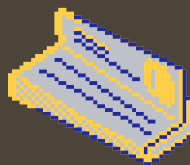
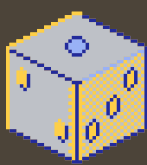
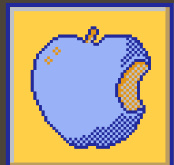
```
def dist(train, test):  
    sum_dist = 0  
    for i in range(len(test)):  
        sum_dist += pow((train[i] - test[i]),2)  
    return sqrt(sum_dist)
```



KNN-處理步驟

5. 尋找K個最近距離及紀錄下來，並按標籤數量投票分類。

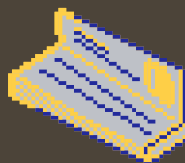
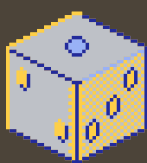
```
#尋找K個最近距離
idx_list = min_idx(K_VALUE, arr_dist)
#記錄最近的K個標籤
outcome_list = [train_data_a["Outcome"][idx_list[x]] for x in range(K_VALUE)]
#按標籤數量投票分類
final_list_a = final_list_a + [classify(outcome_list)]
```



KNN-處理步驟

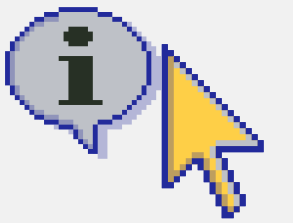
6.最後將結果進行比對。

```
#計算準確率
for i in range(len(single_test_a)):
    if final_list_a[i] == test_data_a["Outcome"][i]:
        current_a = current_a + 1;
for i in range(len(single_test_b)):
    if final_list_b[i] == test_data_b["Outcome"][i]:
        current_b = current_b + 1;
print("Accuracy of A : %.2f%%" % (current_a/len(single_test_a)*100))
print("Accuracy of B : %.2f%%" % (current_b/len(single_test_b)*100))
```



Random Forest

-處理步驟



1. 首先，我們載入了需要使用的套件

```
import csv
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
```

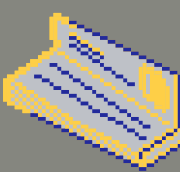
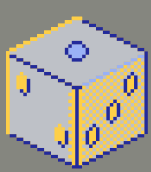
2. 使用csv套件讀取我們需要的資料

```
csvfile = open(r"C:\Users\User\Desktop\school\2nd\DataMining\A\test_data.csv")
testfile = list(csv.reader(csvfile))
csvfile.close()

csvfile = open(r"C:\Users\User\Desktop\school\2nd\DataMining\A\train_data.csv")
trainfile = list(csv.reader(csvfile))
csvfile.close()

csvfile = open(r"C:\Users\User\Desktop\school\2nd\DataMining\B\test_data.csv")
test2 = list(csv.reader(csvfile))
csvfile.close()

csvfile = open(r"C:\Users\User\Desktop\school\2nd\DataMining\B\train_data.csv")
train2 = list(csv.reader(csvfile))
csvfile.close()
```



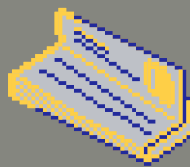
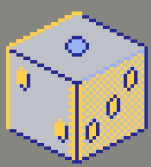
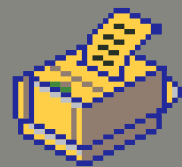
Random Forest

-處理步驟



3. 接下來，我們從trainfile和testfile中特徵和標籤將其分別存儲在x_train、x_test、y_train和y_test中。

```
x_train = [i[:-1] for i in trainfile[1:]]  
x_test = [i[:-1] for i in testfile[1:]]  
y_train = [i[-1] for i in trainfile[1:]]  
y_test = [[i[-1]] for i in testfile[1:]]
```



Random Forest

-處理步驟

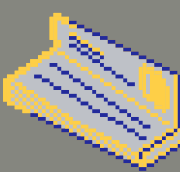
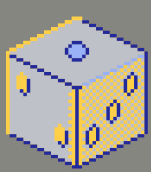


4.使用

randomForestModel.fit()方法
來訓練模型，傳入的參數為
x_train和y_train。

```
randomForestModel = RandomForestClassifier(n_estimators=100)  
randomForestModel.fit(x_train2, y_train2)
```

*randomForestModel.fit()
是用來訓練隨機森林模型的方法。
它的作用是根據給定的訓練資料（通常
包括特徵和標籤）來訓練一個隨機森林
模型。



Random Forest

-處理步驟

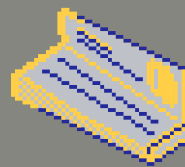
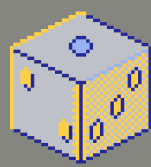


5. 使用

`randomForesModel.predict()` 方法可以對新的數據進行預測，以預測它屬於哪個類別，並將預測結果存儲在 `predict` 中。

```
predict = randomForestModel.predict(x_train2)
```

6. 在模型訓練完成後，我們使用 `score()` 方法來計算模型的精度，並將結果輸出到屏幕上。





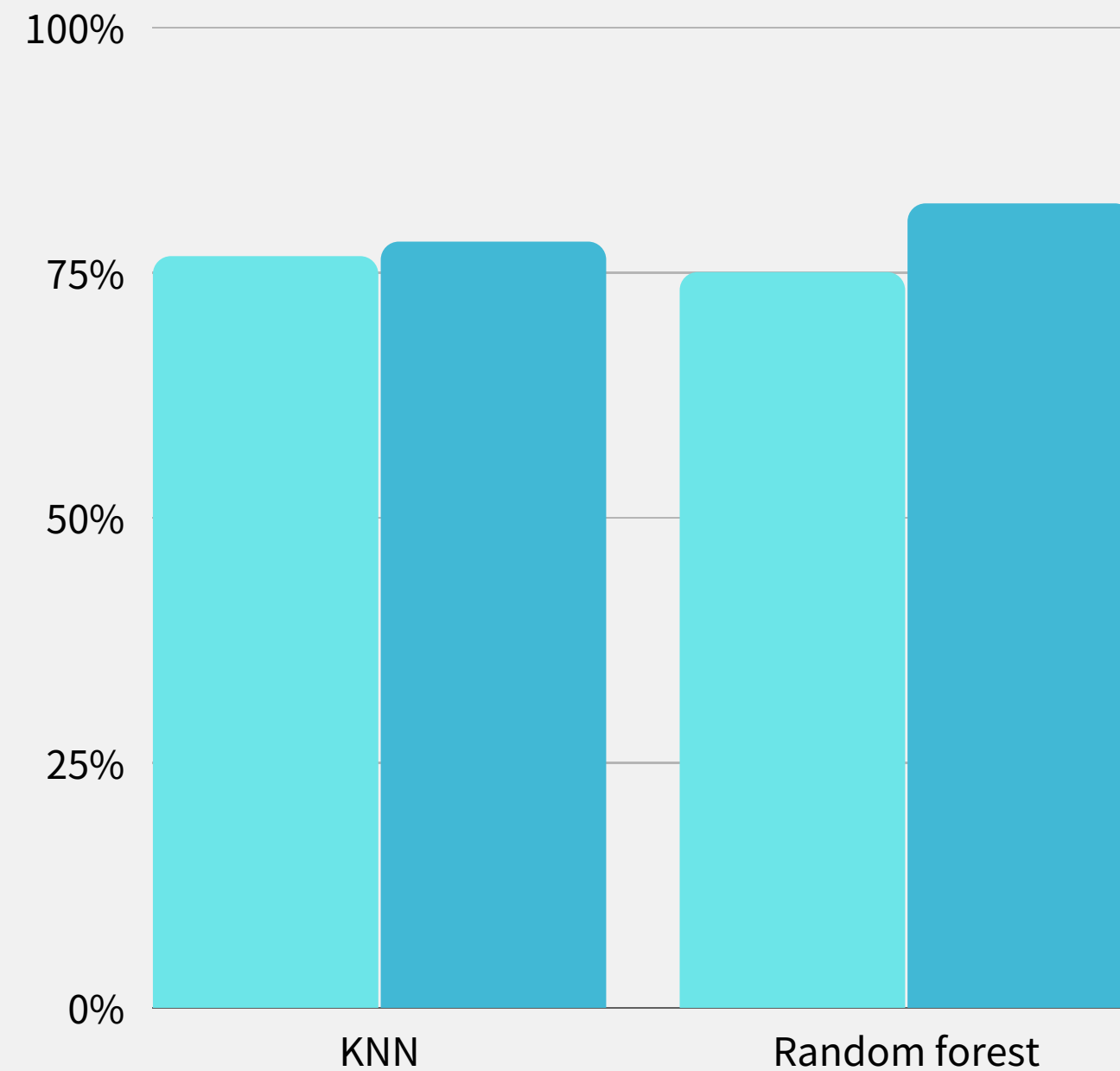
四、結果與討論

[Back to Agenda Page](#)



KNN之結果

- Accuracy of A : 76.62%
- Accuracy of B : 75.00%



▲程式結果

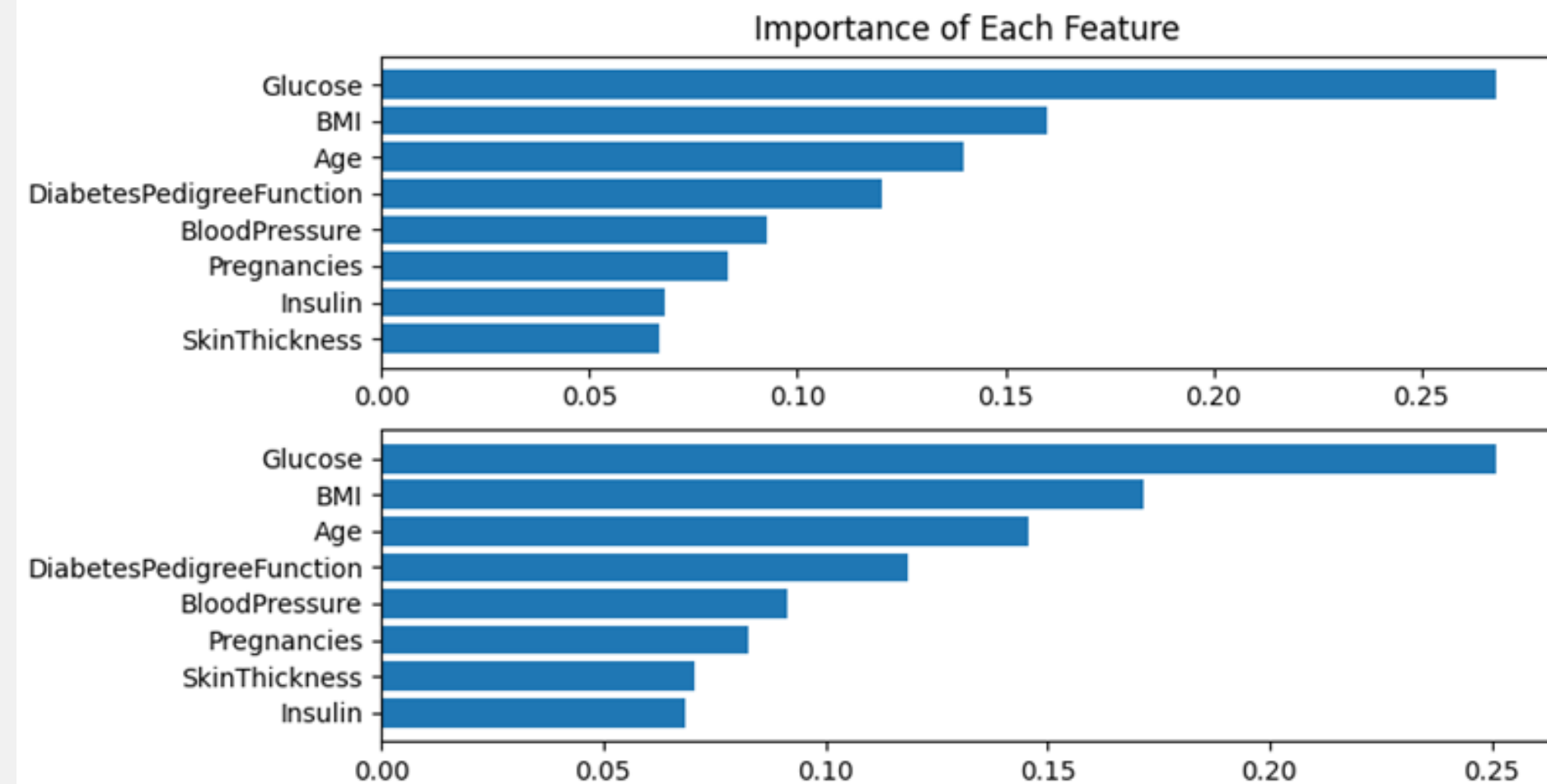
Random forest之結果

A:

- Test: 100%
- Train: 78.1%

B

- Test: 100%
- Train 82%



▲各features的重要性

討論

從程式得出的結論是
血液中葡萄糖濃度
與BMI為前二重要的特徵

關於Features 的重要性，程式得出的數據合理嗎？



現實中

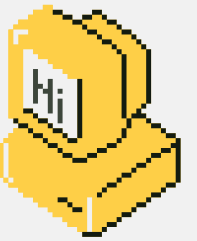
與現實情況成功連結

糖化血色素及血糖	BMI
是確認是否有糖尿病的重要指標， 已確認有糖尿病的患者更是要定期檢查。 *最好每 3 個月追蹤檢查一次	體重控制有助於糖尿病的血糖、血脂及血壓控制。 資料來源:糖尿病友「腰」約控糖！ - 衛生福利部 https://www.mohw.gov.tw/fp-2649-20037-1.html

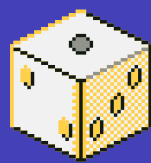
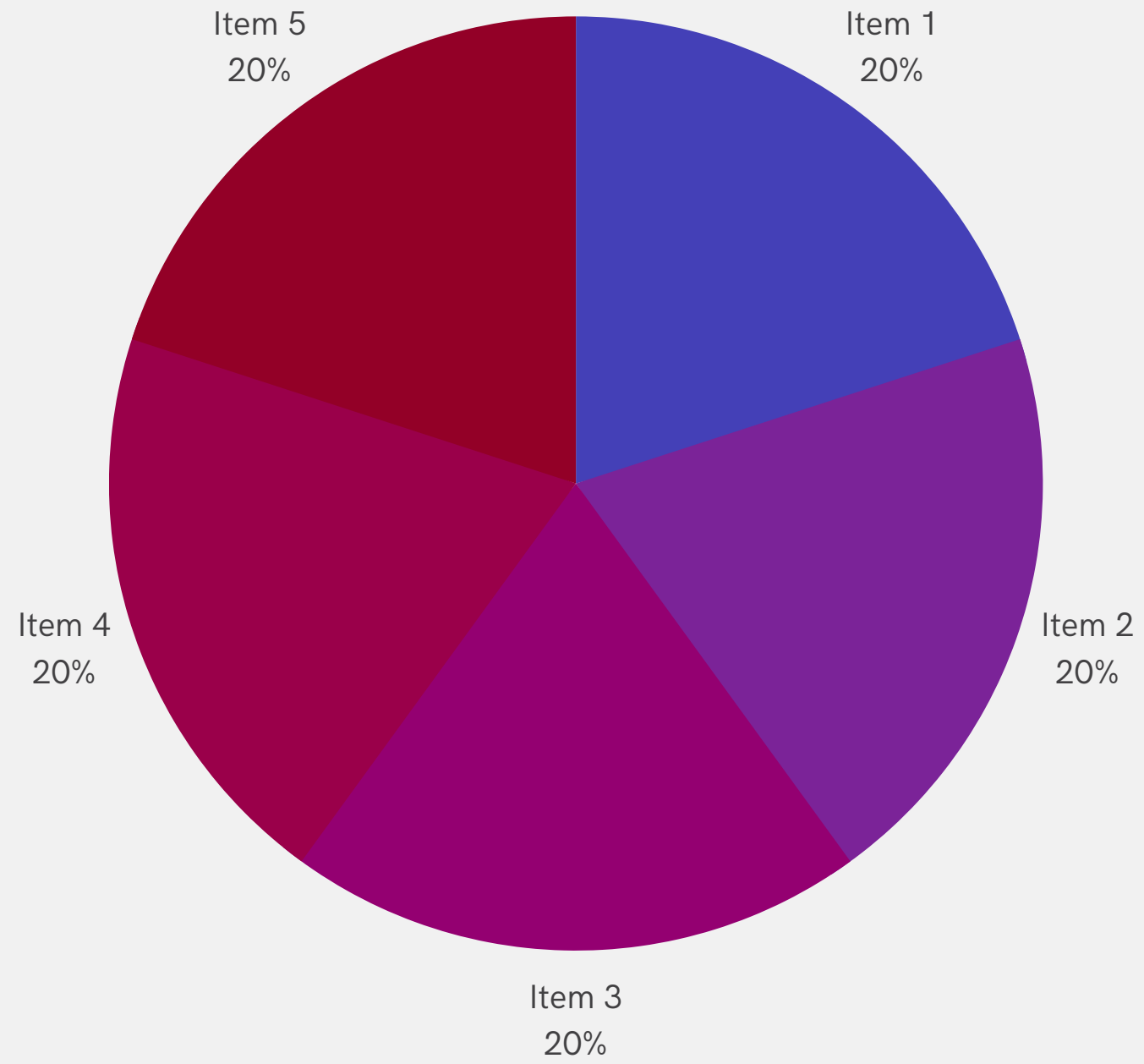
資料來源:中國醫藥大學-衛教單張-糖化血色素及血糖控制標準

<https://www.cmuh.cmu.edu.tw/HealthEdus/Detail?no=4865>





QA~



[Back to Agenda Page](#)

參考資料

林昕潔, 柯皓仁, & 楊維邦. (2005). 以 SVM 與詮釋資料設計書籍分類系統 (Doctoral dissertation)

Han, J., Pei, J., & Tong, H. (2022). Data mining: concepts and techniques. Morgan kaufmann.

糖尿病友「腰」約控糖！ - 衛生福利部

<https://www.mohw.gov.tw/fp-2649-20037-1.html>

中國醫藥大學-衛教單張-糖化血色素及血糖控制標準

<https://www.cmuh.cmu.edu.tw/HealthEdus/Detail?no=4865>



Thank you!

以上是12組的期中報告，謝謝聆聽!

