機器學習工程應用 期中競賽報告

姓名:盧柏翰　　學號:F14081088　　系所:不分系大四

1. 資料前處理

1. 先導入相關模組，像是 numpy、matplotlib 以及 pandas 後，即使用 pandas 讀取 csv 檔案。
2. 針對 label 進行處理，按照題目要求，依照「推薦類型」進行分類，若有出現錯誤、無法辨識或是不在題目要求的類別中，則先將其的 label 值設為 4，並將其刪除之。

	增加肌肉	增強體魄	舒緩壓力	推薦類型
0	-3.0	0.0	-3.0	肌力
1	0.0	0.0	1.0	error
2	1.0	1.0	0.0	error
3	1.0	1.0	1.0	肌耐力
4	0.0	1.0	1.0	error
5	1.0	1.0	1.0	肌力
6	1.0	1.0	-4.0	心肺
7	0.0	0.0	0.0	肌耐力
8	0.0	0.0	0.0	肌力
9	1.0	1.0	1.0	柔軟度

```
for index in range(len(data_train)):

  target = data_train['推薦類型'][index]

  if (target == '肌力'):
    data_train['推薦類型'][index] = 0
  elif (target == '柔軟度'):
    data_train['推薦類型'][index] = 1
  elif (target == '心肺'):
    data_train['推薦類型'][index] = 2
  elif (target == '肌耐力'):
    data_train['推薦類型'][index] = 3
  else:
    data_train['推薦類型'][index] = 4

drop_list = []

for i in range(len(data_train)):
  if (data_train['推薦類型'][i] == 4):
    drop_list.append(i)

data_train = data_train.drop(data_train.index[drop_list])
```

3. 下一步即針對年紀進行處理，有些年齡及其不合理，像是：負值、極大或是極小，因此，可判定其為不合理之資料，故刪除之。

	性別	年齡	
0	1.0	957.0	0.0
1	1.0	45.0	-28.5
2	1.0	47.0	22.5
3	0.0	1134.0	0.0
4	1.0	3231.0	20.1
5	0.0	1093.0	24.0
6	1.0	46.0	-20.9

```python
drop_list = []

for i in range(len(data_train)):
  if (data_train['年齡'][i] > 100) or (data_train['年齡'][i] < 0):
    drop_list.append(i)

data_train = data_train.drop(data_train.index[drop_list])
```

4. 接下來是處理 BMI 值，因為 BMI 的公式為 體重(公斤) / 身高^2(公尺^2)，故其應為正值，在 training 的資料中，BMI 為負值之資料應為不合理資料，故刪除之。

	性別	年齡	BMI	
0	1.0	45.0	-28.543927	0.(
1	1.0	47.0	22.521372	0.(
2	1.0	46.0	-20.958439	0.(
3	1.0	46.0	23.161083	1.(
4	1.0	43.0	22.053277	0.(
5	1.0	45.0	-17.186579	1.(
6	1.0	47.0	-24.895808	0.(
7	1.0	45.0	-26.472678	0.(
8	0.0	54.0	-20.888581	1.(
9	0.0	42.0	-23.332068	1.(
10	0.0	42.0	-31.714518	0.(
11	1.0	38.0	19.310658	0.(
12	1.0	23.0	-19.280249	0.(
13	1.0	46.0	-27.972961	0.(
14	0.0	58.0	-21.024907	1.(

5. 針對各式興趣，資料的型態應為布林值，或是對應到整數或浮點數應為 1（1.0）及 0（0.0），非上述所描述之數值，一樣為不合理資料，故刪除之。

	性別	年齡	BMI	拳擊	器材	有氧	舞蹈	高強度間歇	瑜伽	肌力訓練	燃脂	減脂瘦身
0	1.0	47.0	22.521372	0.0	0.0	1.0	1.0	1.0	1.0	0.0	1.0	1.0
1	1.0	46.0	23.161083	1.0	1.0	1.0	0.0	1.0	0.0	1.0	1.0	0.0
2	1.0	43.0	22.053277	0.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	1.0
3	1.0	38.0	19.310658	0.0	1.0	1.0	1.0	0.0	0.0	1.0	0.0	1.0
4	1.0	45.0	20.259939	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0
5	0.0	26.0	16.358025	0.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0	1.0
6	1.0	44.0	25.100128	0.0	1.0	1.0	1.0	1.0	0.0	0.0	1.0	0.0
7	1.0	46.0	23.527708	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	0.0
8	1.0	28.0	18.857219	0.0	0.0	1.0	0.0	1.0	1.0	1.0	0.0	0.0
9	1.0	35.0	22.674023	0.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0
10	1.0	12.0	17.301038	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0
11	1.0	29.0	16.160408	0.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	0.0
12	1.0	46.0	23.581898	0.0	1.0	1.0	1.0	1.0	0.0	0.0	1.0	0.0
13	1.0	48.0	17.018127	1.0	1.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0
14	0.0	46.0	22.491349	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	1.0
15	0.0	27.0	16.160693	0.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	0.0

6. 將不合理的資料刪除後，將「推薦類型」（也就是結果）設為訓練之結果（train_y），其餘資料去除推薦類型欄位後，設為訓練資料集（train_x）。

7. 上述有對資料進行刪除，都續對刪除後的資料進行重新編號，可透過 reset_index(inplace=True, drop=True) 完成 index 之 reset。
8. 為使用 sklearn 模組進行訓練，將上述 train_y 及 train_x 從 dataframe 轉為 numpy 之資料型態（使用 .values）。

```
train_y = data_train['推薦類型']

# Make label's type as int
train_y = train_y.astype('int')

print(train_y)
```

```
0        0
1        0
2        1
3        0
4        2
       ..
3423     0
3424     0
3425     1
3426     2
3427     2
Name: 推薦類型, Length: 3428, dtype: int64
```

```
train_x = data_train.drop(['推薦類型'], axis=1)
print(train_x)
```

```
        性別    年齡         BMI   拳擊   器材   有氧   舞蹈  高強度間歇   瑜伽  肌力訓練   燃脂  減脂瘦身  \
0     1.0  47.0  22.521372  0.0  0.0  1.0  1.0   1.0  1.0   0.0  1.0   1.0
1     1.0  46.0  23.161083  1.0  1.0  1.0  0.0   1.0  0.0   1.0  1.0   0.0
2     1.0  43.0  22.053277  0.0  1.0  1.0  1.0   1.0  0.0   0.0  0.0   1.0
3     1.0  38.0  19.310658  0.0  1.0  1.0  1.0   0.0  0.0   1.0  0.0   1.0
4     1.0  45.0  20.259939  1.0  1.0  1.0  1.0   1.0  1.0   1.0  1.0   0.0
...   ...   ...        ...  ...  ...  ...  ...   ...  ...   ...  ...   ...
3423  0.0  46.0  21.589910  1.0  1.0  1.0  1.0   0.0  1.0   1.0  1.0   1.0
3424  1.0  46.0  22.491349  0.0  0.0  1.0  0.0   1.0  1.0   1.0  1.0   0.0
3425  0.0  45.0  22.157158  0.0  1.0  0.0  1.0   1.0  0.0   0.0  1.0   0.0
3426  1.0  26.0  14.634752  1.0  0.0  0.0  0.0   0.0  0.0   0.0  0.0   0.0
3427  0.0  30.0  18.818924  0.0  1.0  1.0  0.0   1.0  1.0   0.0  1.0   0.0

      增加肌肉  增強體魄  舒緩壓力
0      1.0   1.0   1.0
1      1.0   0.0   1.0
2      1.0   1.0   1.0
3      0.0   1.0   1.0
4      1.0   0.0   0.0
...    ...   ...   ...
3423   1.0   0.0   1.0
3424   1.0   1.0   0.0
3425   1.0   1.0   0.0
3426   0.0   0.0   1.0
3427   0.0   1.0   1.0

[3428 rows x 15 columns]
```

9. 另外，為了測試模型對於未看過資料之準確度，亦有測試將 train_y 及 train_x 透過 train_test_split 分成 X_train, X_test, y_train, y_test 對模型進行驗證，而非只單純將結果上傳至 Kaggle 去測試模型對於未看過資料的準確度。

2. 訓練模型建立

我使用了四種分類器進行測試，分別為 Decision tree、Random forest、SVM 及 GradientBoost。上面也有提到，為了確認模型對於「未看過」之資料的準確度，我也有將訓練集分成 train 和 test，並使用 cross_val_score 去追蹤模型的

成效。目前而言，最佳的分類器為 Random forest，針對 Kaggle 平台的結果已達到 80% 以上的準確率。

```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.svm import SVC

rfc = RandomForestClassifier()

param_grid = {
    'n_estimators': [459,460,461],
    'max_depth' : [14,15,16],
    'criterion' :['gini', 'entropy']
}

CV_rfc = GridSearchCV(estimator = rfc, param_grid = param_grid, cv= 3)
CV_rfc.fit(train_x, train_y)

# Use Cross Validation To Evaluate Model
CV_Result = cross_val_score(CV_rfc, train_x, train_y, cv = 3, n_jobs = -1)
print(); print(CV_rfc.best_params_)
print(); print(CV_Result)
print(); print(CV_Result.mean())
print(); print(CV_Result.std())

test_y_predicted = CV_rfc.predict(data_test)
```

```
{'criterion': 'gini', 'max_depth': 14, 'n_estimators': 459}

[0.81889764 0.80927384 0.81611208]

0.8147611875427422
```

3. 參數調整

除了上述所說，我使用四種分類器的模型進行比較外，我也同時使用 GridSearchCV，去測試多組參數，並印出其平均準確率，以評斷模型之效力。透過 GridSearch 即可透過類似暴力解的形式，將所給予的參數排列組合，找出成效最佳的參數組合和模型，CV 則可對資料集處理，將其拆分為 train 及 test（這邊應稱為 validation 較佳，因為此題之 test 為未公開之數據），測試模型對於未看過資料的準確度。

4. 預測結果

我是將結果存入 list，因為先前除了測試分類器外，我亦有使用回歸分析做測試，方式為模型預測結果做四捨五入，以達成分類效果，然而，成效不盡理想。最後，為達成規定的輸出格式，我導入 csv 和

itertools 的 zip_longest，將輸出結果寫入 csv 中，並加入 index（即為測試資料之資料數量），後再上傳至 Kaggle，目前準確率最高分之模型為透過 GridSearchCV 找出的 Random forest 分類器，效果達八成。

```
test_y = []

for num in test_y_predicted:
  test_y.append(round(num))

list_index = list()

for x in range(len(data_test)):
    list_index.append(x)

import csv
from itertools import zip_longest

# data = [list_index, test_y_predicted]
data = [list_index, test_y]
columns_data = zip_longest(*data)

with open("F14081088_1019.csv","w") as f:
    writer = csv.writer(f)
    writer.writerow(['index', 'Label'])
    writer.writerows(columns_data)
```