IIE
Project3

利用 SVM 分類脊椎異常患者

葉紹康 108034552

日期：2020 年 01 月 02 日

一、 主題說明

1. 主題描述

這次我選用的資料庫是 UCI 的公開數據集上面發布的資料，主要針對脊椎異常的患者測量出不同的數據資料能辨別出正常人與病患的差別，總共分成三類分別為正常人、椎間盤突出患者、腰椎滑脫患者，每筆資訊都提供六種資料分別是骨盆入射角、骨盆傾斜度、腰椎前凸角、骶骨傾斜坡、骨盆前凸弧度以及脊椎滑脫等級。這個專案會使用上述六種特徵並使用 SVM 分類器對於三種類別正常人、椎間盤突出患者、腰椎滑脫患者做分類，並在從分析結果中尋找是否有可改善與增加分類準確率的方法。

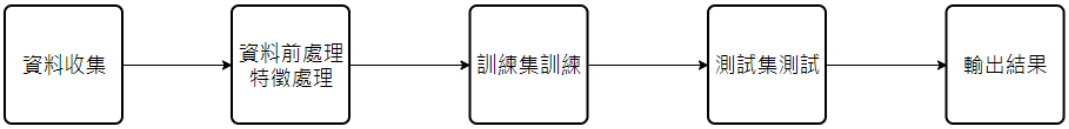
2. 5W1H

- Why
  讓整體醫學研究能更有效率且歸納出相關性。
- What
  建立一個辨識脊椎異常病患的系統，讓往後只需輸入相關資料即可自動辨別出是否有生病。
- Where
  醫院或診所都能使用。
- When
  當民眾希望了解自身身體狀況是否有異常時。
- Who
  任何民眾皆可。
- How
  收集不同脊椎異常患者的資料，利用支援向量機分類這些照片，以及判斷結果是否相符。

3. 執行流程圖

整體執行流程共分以下五個步驟：

(1) 資料收集
(2) 資料前處理
(3) 訓練集訓練
(4) 測試集測試
(5) 輸出結果



二、 資料整理

1. 資料集照片來源

由 UCI 公開數據集中取得脊椎異常與正常人的資料集，共分成 3 種類別，分別包含正常人、椎間盤突出患者、腰椎滑脫患者，且資料集有六種特徵為骨盆入射角、骨盆傾斜度、腰椎前凸角、骶骨傾斜坡、骨盆前凸弧度以及脊椎滑脫等級；資料數分別為正常人 100 筆、椎間盤突出患者 60 筆、腰椎滑脫患者 150 筆資料，總共 310 筆資料。

	A	B	C	D	E	F	G
1	63.03	22.55	39.61	40.48	98.67	-0.25	DH
2	39.06	10.06	25.02	29	114.41	4.56	DH
3	68.83	22.22	50.09	46.61	105.99	-3.53	DH
4	69.3	24.65	44.31	44.64	101.87	11.21	DH
5	49.71	9.65	28.32	40.06	108.17	7.92	DH
6	40.25	13.92	25.12	26.33	130.33	2.23	DH
7	53.43	15.86	37.17	37.57	120.57	5.99	DH
8	45.37	10.76	29.04	34.61	117.27	-10.68	DH
9	43.79	13.53	42.69	30.26	125	13.29	DH
10	36.69	5.01	41.95	31.68	84.24	0.66	DH
11	49.71	13.04	31.33	36.67	108.65	-7.83	DH
12	31.23	17.72	15.5	13.52	120.06	0.5	DH
13	48.92	19.96	40.26	28.95	119.32	8.03	DH
14	53.57	20.46	33.1	33.11	110.97	7.04	DH

2. 資料前處理過程 Data-preprocessing process

(1) 第一步透過 pandas 套件將原始資料匯入程式中，由下圖可看出原始資料共有七列不同的資訊，而這七列資訊最後一列為三種類別的其中之一，其他六列則是上述提到的六種特徵，分別是骨盆入射角、骨盆傾斜度、腰椎前凸角、骶骨傾斜坡、骨盆前凸弧度以及脊椎滑脫等級。

```
import pandas as pd
data = pd.read_csv('C:/Users/Lab905/Desktop/data.csv',header=None)
print(data.head)
print("data shape :", data.shape)
```

```
<bound method NDFrame.head of          0      1      2      3       4      5   6
0    63.03  22.55  39.61  40.48   98.67  -0.25  DH
1    39.06  10.06  25.02  29.00  114.41   4.56  DH
2    68.83  22.22  50.09  46.61  105.99  -3.53  DH
3    69.30  24.65  44.31  44.64  101.87  11.21  DH
4    49.71   9.65  28.32  40.06  108.17   7.92  DH
5    40.25  13.92  25.12  26.33  130.33   2.23  DH
6    53.43  15.86  37.17  37.57  120.57   5.99  DH
7    45.37  10.76  29.04  34.61  117.27 -10.68  DH
```

(2) 第二步將資料每一種特徵加入標籤，並且將三種類別分別以數字表示，原始資料中 DH（椎間盤突出患者）以數字 1 表示、SL（腰椎滑脫患者）以數字 2 表示、NO（正常人）以數字 3 表示。

```
data.columns = [
    'pelvic_incidence', 'pelvic_tilt',
    'lumbar_lordosis_angle',
    'sacral_slope', 'pelvic_radius', 'grade_of_spondylolisthesis', 'labels'
]
for i in range(0,len(data)):
    if(data['labels'][i] == 'DH'):
        data['labels'][i] = 1
    elif(data['labels'][i] == 'SL'):
        data['labels'][i] = 2
    elif(data['labels'][i] == 'NO'):
        data['labels'][i] = 3
print(data.head)
```

(3) 第三步將資料集前六列設定為特徵分別是骨盆入射角、骨盆傾斜度、腰椎前凸角、骶骨傾斜坡、骨盆前凸弧度以及脊椎滑脫等級。而第七列為三種患者與一般人的標籤，並將資料集做分割，資料的前 60 筆為椎間盤突出患者；第 61 筆至第 210 筆為腰椎滑脫患者；最後 100 筆為一般正常人。

```
all_attribute = data.iloc[:, 0:6]
all_label = data.iloc[:, 6]
all_label = all_label.values.reshape((310))
print('all_attribute : ', all_attribute.shape)
print('all_label : ', all_label.shape)
```

```
all_attribute :  (310, 6)
all_label :  (310,)
```

```
DH = all_attribute.iloc[0:60, :]
DH_label = all_label[0:60]

SL = all_attribute.iloc[60:210, :]
SL_label = all_label[60:210]

normal = all_attribute.iloc[210:310, :]
normal_label = all_label[210:310]

print('DH = ', DH.shape)
print('SL = ', SL.shape)
print('normal = ', normal.shape)
```

```
DH =  (60, 6)
SL =  (150, 6)
normal =  (100, 6)
```

(4) 第四步將數據集分割成 80%為訓練集，20%為測試集，以利後續分類時使用。

```
DH_train = DH.iloc[0:48, :]
DH_train_label = DH_label[0:48]
DH_test = DH.iloc[48:60, :]
DH_test_label = DH_label[48:60]

SL_train = SL.iloc[0:120, :]
SL_train_label = SL_label[0:120]
SL_test = SL.iloc[120:150, :]
SL_test_label = SL_label[120:150]

normal_train = normal.iloc[0:80, :]
normal_train_label = normal_label[0:80]
normal_test = normal.iloc[80:100, :]
normal_test_label = normal_label[80:100]

print('DH_train = ', DH_train.shape)
print('DH_test = ', DH_test.shape)
print('SL_train = ', SL_train.shape)
print('SL_test = ', SL_test.shape)
print('normal_train = ', normal_train.shape)
print('normal_test = ', normal_test.shape)
```

```
DH_train =  (48, 6)
DH_test =  (12, 6)
SL_train =  (120, 6)
SL_test =  (30, 6)
normal_train =  (80, 6)
normal_test =  (20, 6)
```

(5) 將訓練集與測試集的資料做整合，並資料型態 float 轉為 int，此處轉為 int 的型態是因為原始資料有小數點無法正常使用 SVM 做分類，最後發現轉為整數型態以後就能順利進行。

```
train = np.vstack((normal_train, DH_train, SL_train))
label = np.hstack((normal_train_label, DH_train_label, SL_train_label))
test = np.vstack((normal_test, DH_test, SL_test))
test_label = np.hstack((normal_test_label, DH_test_label, SL_test_label))
```

```
train = train.astype(int)
label = label.astype(int)
test = test.astype(int)
test_label = test_label.astype(int)
```

三、 模型設定

- 模型架構 model architecture

  初始模型為 kernel 核函式型別設為 rbf；C 懲罰係數，即對誤差的寬容度，C 越大對分錯樣本的懲罰程度越大，因此在訓練樣本中準確率越高，但是泛化能力降低，也就是對測試數據的分類準確率降低。相反，減小 C 的話，容許訓練樣本中有一些誤分類錯誤樣本，泛化能力強。初始的懲罰係數設為 100。

```
clf = SVC(kernel='rbf',C=100,decision_function_shape=None)
clf.fit(train,label)
```

```
C:\Users\Lab905\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: Fut
'auto' to 'scale' in version 0.22 to account better for unscaled feature
his warning.
  "avoid this warning.", FutureWarning)
```

```
SVC(C=100, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape=None, degree=3, gamma='auto_deprecated',
    kernel='rbf', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

四、 訓練過程

1. 如何提升模型性能

使用原先之最初始模型測試以後，發展出第二個模型使用 kernel 為 poly 發現準確率有所上升，第三個模型將 kernel 更改為 linear，最後再將 C 調整為 0.01 發現準確率達到最高。

2. 訓練過程

（1）測試 1 為原先之最初始模型，測試集準確率為 48.38%。

```
prediction = clf.predict(test)
```

```
from sklearn import metrics
print('Accuracy: ', metrics.accuracy_score(test_label, prediction))
```

```
Accuracy:  0.4838709677419355
```

（2）測試 2 將 kernel 更改為 poly，測試集準確率為 74.19%。

```
clf = SVC(kernel='poly',C=100,decision_function_shape=None)
clf.fit(train,label)
```

```
C:\Users\Lab905\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: Futu
'auto' to 'scale' in version 0.22 to account better for unscaled features
his warning.
  "avoid this warning.", FutureWarning)
```

```
SVC(C=100, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape=None, degree=3, gamma='auto_deprecated',
    kernel='poly', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

```
prediction = clf.predict(test)
```

```
from sklearn import metrics
print('Accuracy: ', metrics.accuracy_score(test_label, prediction))
```

```
Accuracy:  0.7419354838709677
```

(3) 測試 3 將 kernel 更改為 linear，準確率為 80.64%。

```
clf = SVC(kernel='linear',C=100,decision_function_shape=None)
clf.fit(train,label)
```

```
SVC(C=100, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape=None, degree=3, gamma='auto_deprecated',
    kernel='linear', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

```
prediction = clf.predict(test)
```

```
from sklearn import metrics
print('Accuracy: ', metrics.accuracy_score(test_label, prediction))
```

```
Accuracy:  0.8064516129032258
```

(4) 測試 4 將 C 更改為 0.01，準確率為 87.09%。

```
clf = SVC(kernel='linear',C=0.01,decision_function_shape=None)
clf.fit(train,label)
```

```
SVC(C=0.01, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape=None, degree=3, gamma='auto_deprecated',
    kernel='linear', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

```
prediction = clf.predict(test)
```

```
from sklearn import metrics
print('Accuracy: ', metrics.accuracy_score(test_label, prediction))
```

```
Accuracy:  0.8709677419354839
```

五、 結論

1. 結果

透過測試的結果可看出，最初的訓練模型準確度約為 48.38%，思考是否有辦法再提升整體的準確度，後來更改一些參數使用不同的模型訓練與測試，對於準確度都能夠有顯著的提升，最終結果可達到

87.09%。

這些結果可以應用在醫學診斷上判斷，當有患者測量出這些特徵以後，透過 SVM 模型分類可以讓整體更有效率，且準確度可以達到 87.09%。

改善步驟	準確度
kernel：rbf → poly	48.38% → 74.19%
kernel：poly → linear	74.19% → 80.64%
C：100 → 0.01	80.64% → **87.09%**

2. 未來改進方向

由於這次數據只有 310 筆，未來可增加更多數據集讓結果更具有可靠性，且也能透過 SVM 分析每個特徵的權重，考慮這些權重的重要性以後，可以嘗試利用更少的特徵值去檢測是否脊椎有問題，如此一來對於日後的醫療診斷就能更快速的判別出患者的情況，提升醫學的進步。

六、 參考來源

- python 機器學習庫 sklearn——支援向量機 svm
https://www.itread01.com/content/1549429381.html
- 支援向量機
https://zh.wikipedia.org/wiki/%E6%94%AF%E6%8C%81%E5%90%91%E9%87%8F%E6%9C%BA
- R 筆記 – (14)Support Vector Machine/Regression(支持向量機 SVM)
https://rpubs.com/skydome20/R-Note14-SVM-SVR
- SVM 的實現多分類的幾種方法以及優缺點詳解
https://www.itread01.com/content/1546997242.html
- SVM 的核函式選擇和調參
https://www.itread01.com/content/1550635205.html
- UCI 資料集
http://archive.ics.uci.edu/ml/datasets/vertebral+column