第10章　交叉驗證

第10章　交叉驗證

- 介紹交叉驗證
- 介紹威斯康辛大學的乳癌腫瘤資料
- 資料探索和切割
- 機器學習模型大亂鬥——不完美版
- 交叉驗證說明
- 機器學習模型大亂鬥——正確版
- 用決策樹找出重要變數

- 先請教各位一個問題：如果想要證明A同學的成績比B同學好，需要用一次考試的成績，還是觀察三次考試的成績會比較客觀？
- 答案當然是──次數越多愈客觀。這就是交叉驗證（cross validation）的主要精神：透過讓機器進行多次的學習和預測取其平均，來判斷哪一個比較好。

範例10-1 用DESCR來了解資料的來龍去脈

程式碼

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
plt.rcParams['font.sans-serif'] = ['DFKai-sb']
plt.rcParams['axes.unicode_minus'] = False
%config InlineBackend.figure_format = 'retina'
import warnings
warnings.filterwarnings('ignore')
from sklearn.datasets import load_breast_cancer
breast_cancer = load_breast_cancer()
print('\n'.join(breast_cancer['DESCR'].split('\n')[:15]))
```

執行結果

```
.. _breast_cancer_dataset:

Breast cancer wisconsin (diagnostic) dataset
--------------------------------------------

**Data Set Characteristics:**

    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and the class

    :Attribute Information:
        - radius (mean of distances from center to points on the perimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
```

範例10-2 檢視資料的特徵值

程式碼

```
print(breast_cancer['feature_names'])
```

執行結果

```
['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
```

範例10-3　將資料和預測的目標值都整合到DataFrame裡

程式碼

```
df = pd.DataFrame(data = breast_cancer['data'], columns = breast_
                cancer['feature_names'])
df['target'] = breast_cancer['target']
df.head()
```

執行結果

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...

5 rows x 31 columns

worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890	0
23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902	0
25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758	0
26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300	0
16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678	0

範例10-5 觀察目標值

程式碼

```
print(f'標籤0為{breast_cancer["target_names"][0]}，是惡性腫瘤的意思')
print(df['target'].value_counts(normalize=True))
```

執行結果

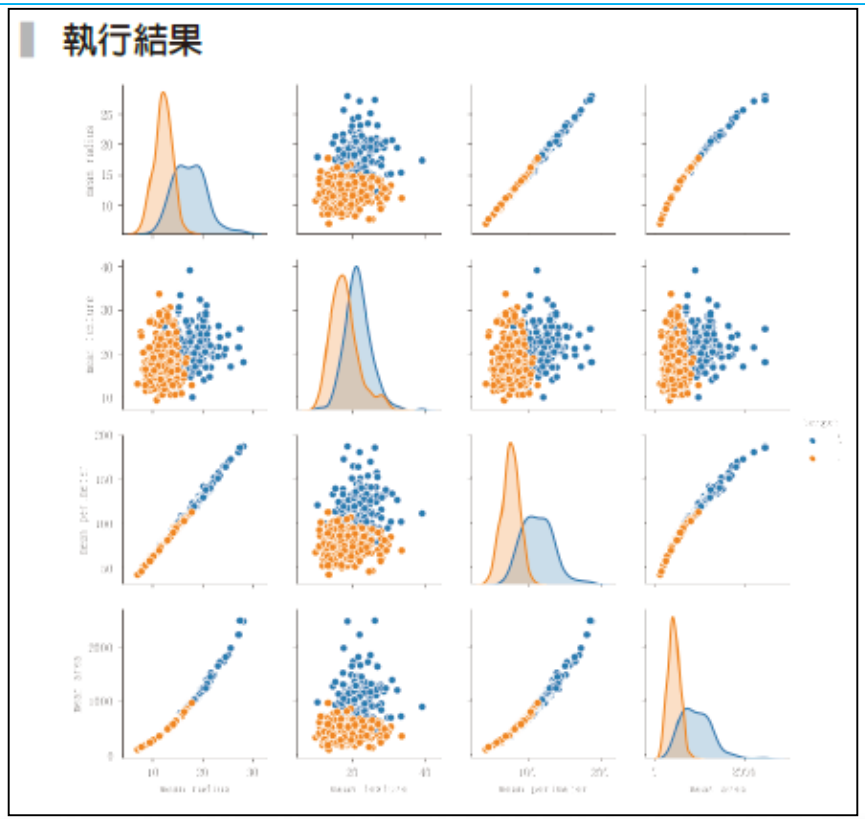
```
標籤 0 爲 malignant，是惡性腫瘤的意思
1    0.627417
0    0.372583
Name: target, dtype: float64
```

範例10-6 pairplot 的資料探索

程式碼

```
sns.pairplot(df, vars=['mean radius','mean texture','mean perimeter',
'mean area'], hue='target', size=2);
```

執行結果



範例10-10　欄位處理和將資料切割成訓練集和測試集

程式碼

```
X_cols = df.columns.drop('target')
X = df[X_cols]
y = df['target']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                   test_size=0.33, random_state=42)
```

範例10-11　載入所有模組，並進行模型學習和預測

程式碼

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

models = [LogisticRegression(), SVC(),
          KNeighborsClassifier(),
          DecisionTreeClassifier(max_depth=5)]
```

範例10-11　載入所有模組，並進行模型學習和預測

承接上一頁

```
scores = {}
for model in models:
    model_pl = make_pipeline(StandardScaler(), model)
    model_pl.fit(X_train, y_train)
    score = model_pl.score(X_test, y_test)
    scores[model._ _class_ _._ _name_ _] = score
scores
```

執行結果

```
{'LogisticRegression': 0.978723404255319,
 'SVC': 0.9680851063829787,
 'KNeighborsClassifier': 0.9574468085106383,
 'DecisionTreeClassifier': 0.9468085106382979}
```

範例10-12　將範例10-11的執行結果整理到DataFrame，並加以排序

程式碼

```
pd.Series(scores).sort_values(ascending=False)
```

執行結果

```
LogisticRegression          0.978723
SVC                         0.968085
KNeighborsClassifier        0.957447
DecisionTreeClassifier      0.946809
dtype: float64
```

10-4 交叉驗證

範例10-13 資料切割

程式碼

```
from sklearn.model_selection import KFold
data = np.arange(10,18)
kfold = KFold(n_splits=4)
for train_idx, test_idx in kfold.split(data):
    print(f'訓練集資料: {data[train_idx]}， 測試集資料
:{data[test_idx]}')
```

執行結果

```
訓練集資料： [12 13 14 15 16 17]， 測試集資料：[10 11]
訓練集資料： [10 11 14 15 16 17]， 測試集資料：[12 13]
訓練集資料： [10 11 12 13 16 17]， 測試集資料：[14 15]
訓練集資料： [10 11 12 13 14 15]， 測試集資料：[16 17]
```

範例10-15 5折交叉驗證的簡單方式

程式碼

```
from sklearn.model_selection import cross_val_score
model_pl_lr = make_pipeline(StandardScaler(),
LogisticRegression())
scores = cross_val_score(model_pl_lr, X_train, y_train,
                         scoring='accuracy', cv=5)
print(f'5折交叉驗證的每次結果 {scores}')
print(f'5折交叉驗證的平均結果{np.mean(scores)}')
```

執行結果

```
5 折交叉驗證的每次結果 [0.98701299 0.96052632 1.         0.96052632 0.97368421]
5 折交叉驗證的平均結果 0.9763499658236501
```

範例10-17 機器學習模型大亂鬥——正確版

程式碼

```
models = [LogisticRegression(), SVC(), KNeighborsClassifier(),
          DecisionTreeClassifier(max_depth=10)]
scores = {}
for model in models:
    model_pl = make_pipeline(StandardScaler(), model)
    score = cross_val_score(model_pl, X_train, y_train,
                            scoring='accuracy', cv=10)
    scores[model._ _class_ _._ _name_ _] = score.mean()
pd.Series(scores).sort_values(ascending=False)
```

執行結果

```
LogisticRegression          0.978794
SVC                         0.970963
KNeighborsClassifier        0.965696
DecisionTreeClassifier      0.926343
dtype: float64
```

範例10-18 檢視羅吉斯迴歸的各項數據報表

程式碼

```
from sklearn.metrics import accuracy_score, confusion_matrix,
    classification_report

model_pl_lr = make_pipeline(StandardScaler(),
LogisticRegression())
model_pl_lr.fit(X_train, y_train)
y_pred = model_pl_lr.predict(X_test)
print('正確率：', accuracy_score(y_test, y_pred).round(3))
print('混亂矩陣')
print(confusion_matrix(y_test, y_pred))
print('綜合報告')
print(classification_report(y_test, y_pred))
```

執行結果

正確率： 0.979

混亂矩陣

```
[[ 66   1]
 [  3 118]]
```

綜合報告

	precision	recall	f1-score	support
0	0.96	0.99	0.97	67
1	0.99	0.98	0.98	121
micro avg	0.98	0.98	0.98	188
macro avg	0.97	0.98	0.98	188
weighted avg	0.98	0.98	0.98	188

範例10-19　用決策樹檢視前五重要的變數和係數

程式碼

```
model_tree = DecisionTreeClassifier(max_depth=10)
model_tree.fit(X_train, y_train)
pd.Series(model_tree.feature_importances_,
        index=X.columns).sort_values(ascending=False).head()
```

執行結果

```
mean concave points     0.723194
worst perimeter         0.077242
worst concavity         0.039281
worst radius            0.033358
worst texture           0.022524
dtype: float64
```

範例10-20 將決策樹結果繪圖

程式碼

```
from sklearn.tree import export_graphviz
import pydot
from IPython.display import Image

features = X.columns
class_names = ['惡性', '良性']
dot_data = export_graphviz(model_tree, out_file=None,
                           feature_names=features,
                           class_names = class_names,
                           proportion = False,
                           max_depth=3,
                           filled=True,
                           rounded=True)
```

範例10-20 將決策樹結果繪圖

承接上一頁

```
graph = pydot.graph_from_dot_data(dot_data)
graph[0].write_png('tumor.png')
Image(graph[0].create_png(), width=800)
```

執行結果