第8章　支持向量機

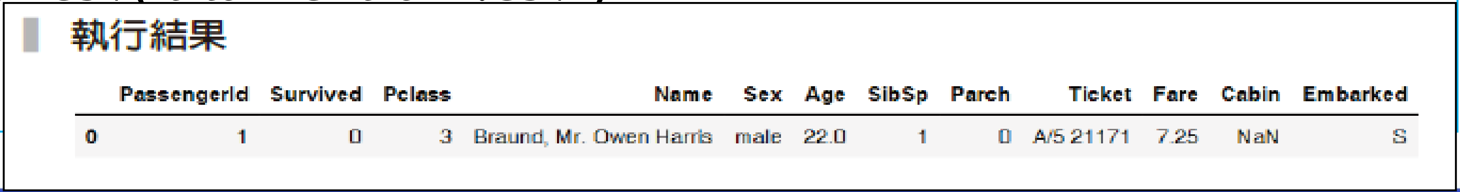
第8章　支持向量機

- 支持向量機（SVM）
- 介紹鐵達尼號的資料集
- 資料探索和切割
- 資料預處理
- 用管道器連結預測器和轉換器
- 管道器的綜合練習，包括將特徵值移除和實驗

範例8-1 載入資料（資料請放在工作目錄）

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')
df = pd.read_csv('titanic_train.csv')
df.head(1)
```



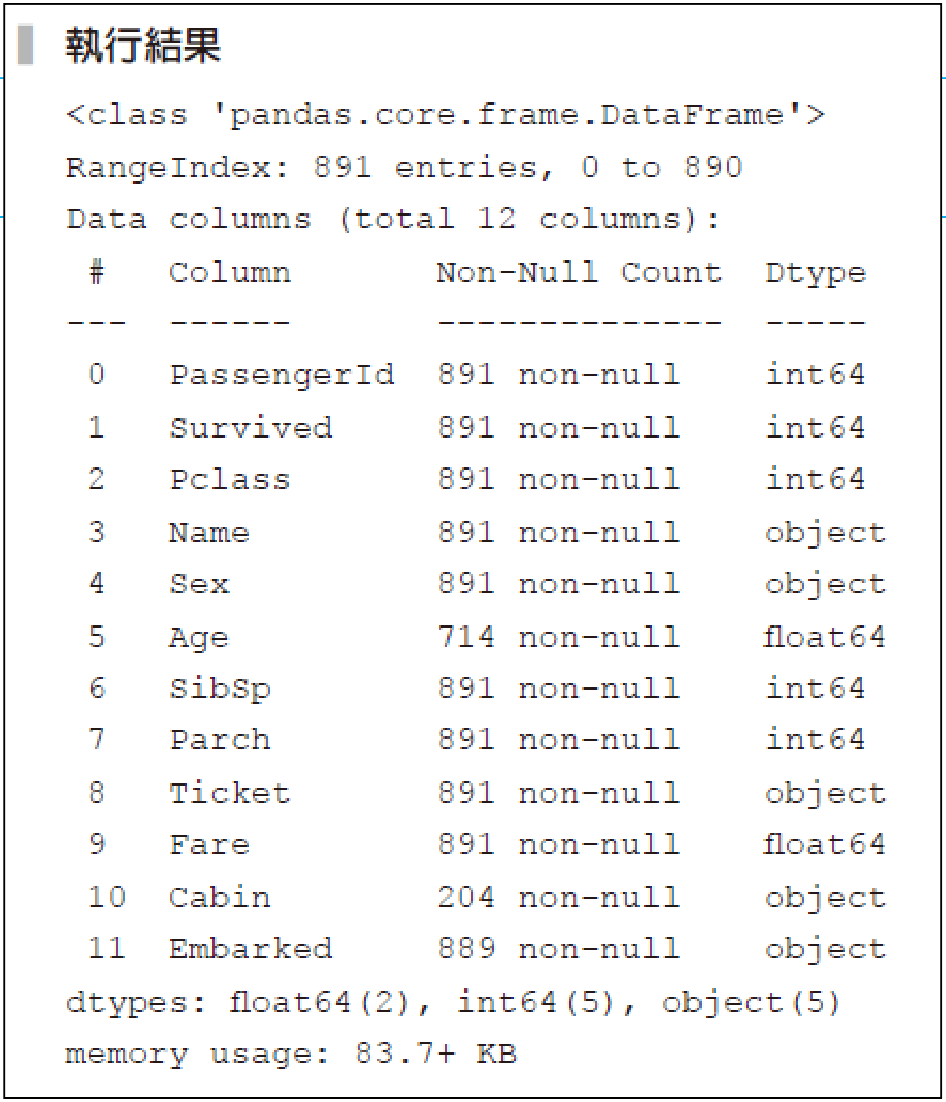
執行結果

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.25	NaN	S

範例8-2 資料檢查和資料欄位說明

程式碼

```
df.info()
```



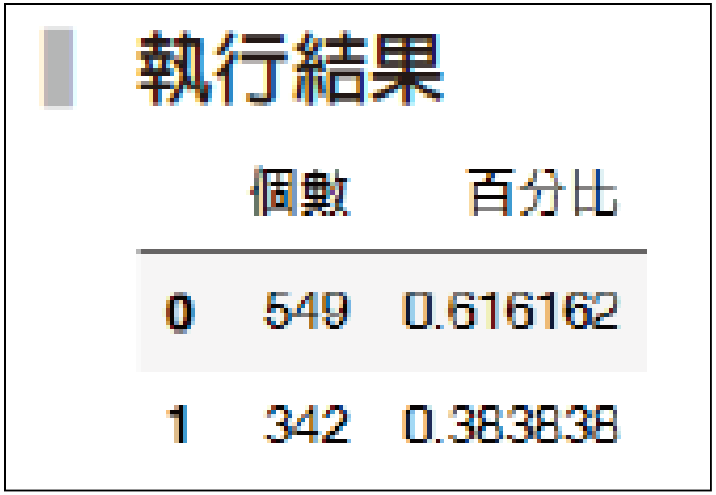
執行結果

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

範例8-3 檢視目標值Survived的分布

程式碼

```
pd.concat([df['Survived'].value_counts(),
        df['Survived'].value_counts(normalize=True)],
        axis=1, keys=['個數','百分比'])
```



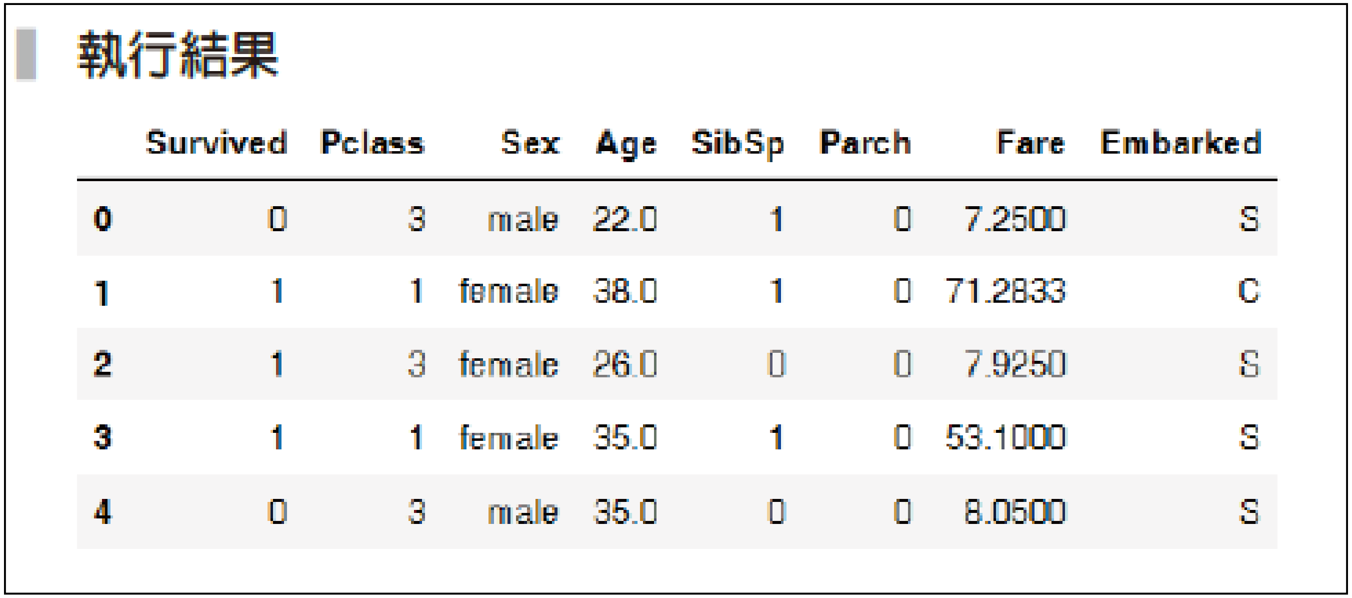
執行結果

	個數	百分比
0	549	0.616162
1	342	0.383838

範例8-4　移除PassengerId、Name、Ticket、Cabin欄位

程式碼

```
df = df.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1)
df.head()
```



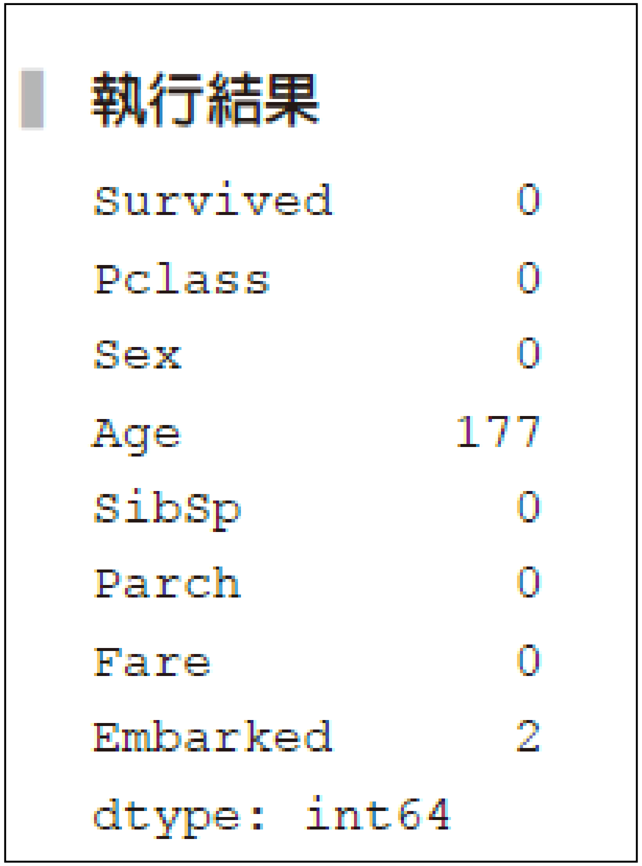
執行結果

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S
1	1	1	female	38.0	1	0	71.2833	C
2	1	3	female	26.0	0	0	7.9250	S
3	1	1	female	35.0	1	0	53.1000	S
4	0	3	male	35.0	0	0	8.0500	S

範例8-5 遺漏值檢查

程式碼

```
df.isnull().sum()
```



執行結果

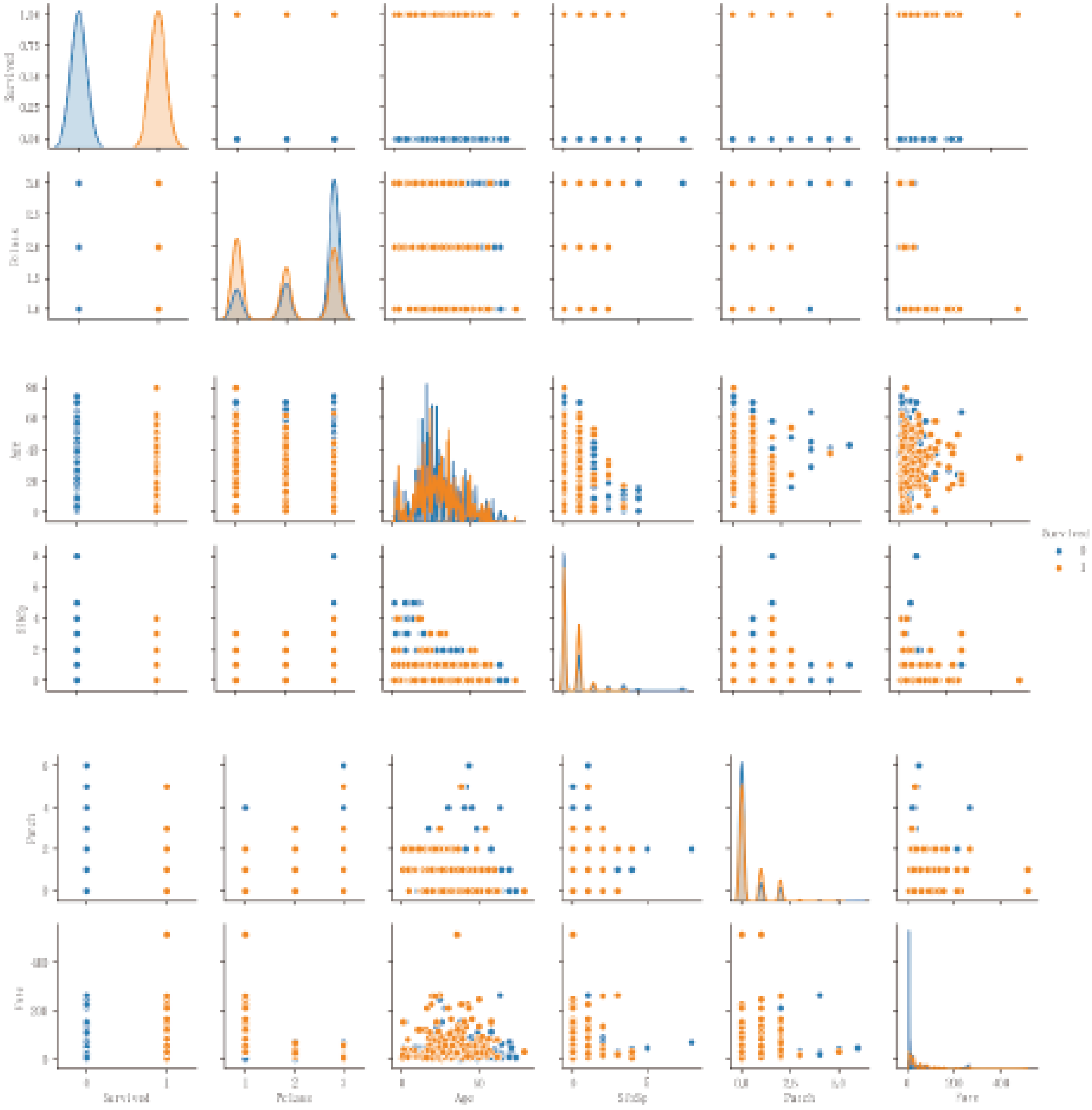
```
Survived      0
Pclass        0
Sex           0
Age         177
SibSp         0
Parch         0
Fare          0
Embarked      2
dtype: int64
```

範例8-6　用seaborn的pairplot來快速檢視變數關係

程式碼

```
sns.pairplot(data=df, hue='Survived',
             size=2, diag_kws={'bw':0.1});
```

執行結果



範例8-10 欄位處理

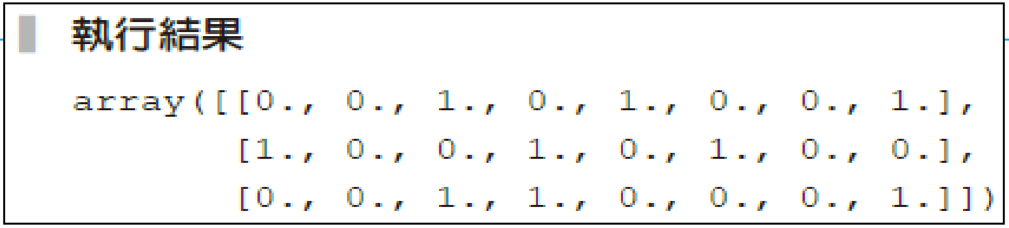
程式碼

```
X_col_num = ['Age', 'SibSp', 'Parch', 'Fare']
X_col_cat = ['Pclass', 'Sex', 'Embarked']
X_cols = X_col_num + X_col_cat
y_col = 'Survived'
```

範例8-11　數值型資料的管道器

程式碼

```
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
num_pl = make_pipeline(
    SimpleImputer(strategy='median'),
    StandardScaler()
)
# 檢查數值管道器的運作
print(f'數值型資料的欄位有：{X_col_num}')
num_pl.fit_transform(df[X_col_num])[:3]
```



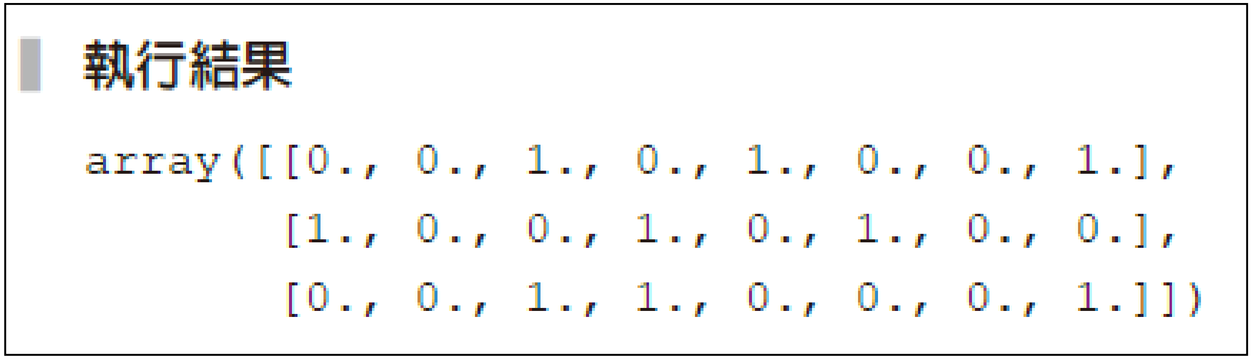
執行結果

```
array([[0., 0., 1., 0., 1., 0., 0., 1.],
       [1., 0., 0., 1., 0., 1., 0., 0.],
       [0., 0., 1., 1., 0., 0., 0., 1.]])
```

範例8-12　類別型資料的管道器

程式碼

```
from sklearn.preprocessing import OneHotEncoder
cat_pl = make_pipeline(
    SimpleImputer(strategy='most_frequent'),
    OneHotEncoder(sparse=False)
)
# 檢查類別管道器的運作
cat_pl.fit_transform(df[X_col_cat])[:3]
```



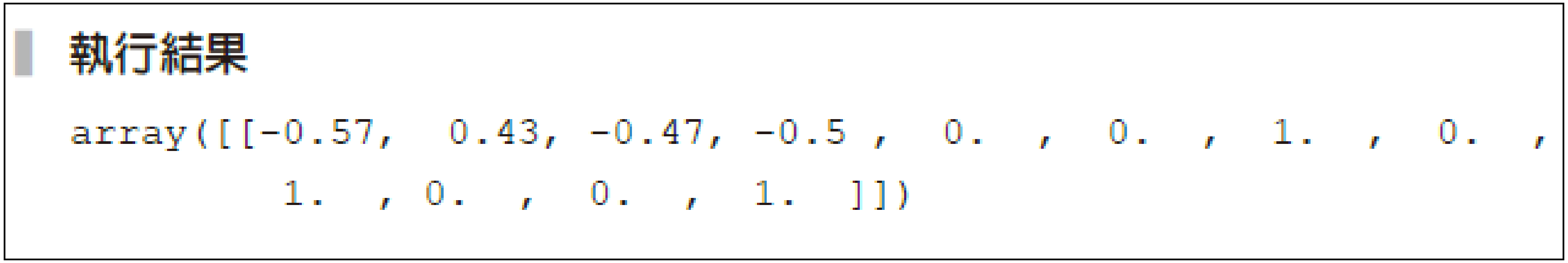
執行結果

```
array([[0., 0., 1., 0., 1., 0., 0., 1.],
       [1., 0., 0., 1., 0., 1., 0., 0.],
       [0., 0., 1., 1., 0., 0., 0., 1.]])
```

範例8-15　用「水平合併器」整合數值和類別資料管道器，並檢視資料

程式碼

```
from sklearn.compose import ColumnTransformer
data_pl = ColumnTransformer([
    ('num_pl', num_pl, X_col_num),
    ('cat_pl', cat_pl, X_col_cat)
])
data_pl.fit_transform(df[X_cols])[:1].round(2)
```



執行結果

```
array([[-0.57,  0.43, -0.47, -0.5 ,  0.  ,  0.  ,  1.  ,  0.  ,
         1.  ,  0.  ,  0.  ,  1.  ]])
```

範例8-16 將資料切成訓練集和測試集

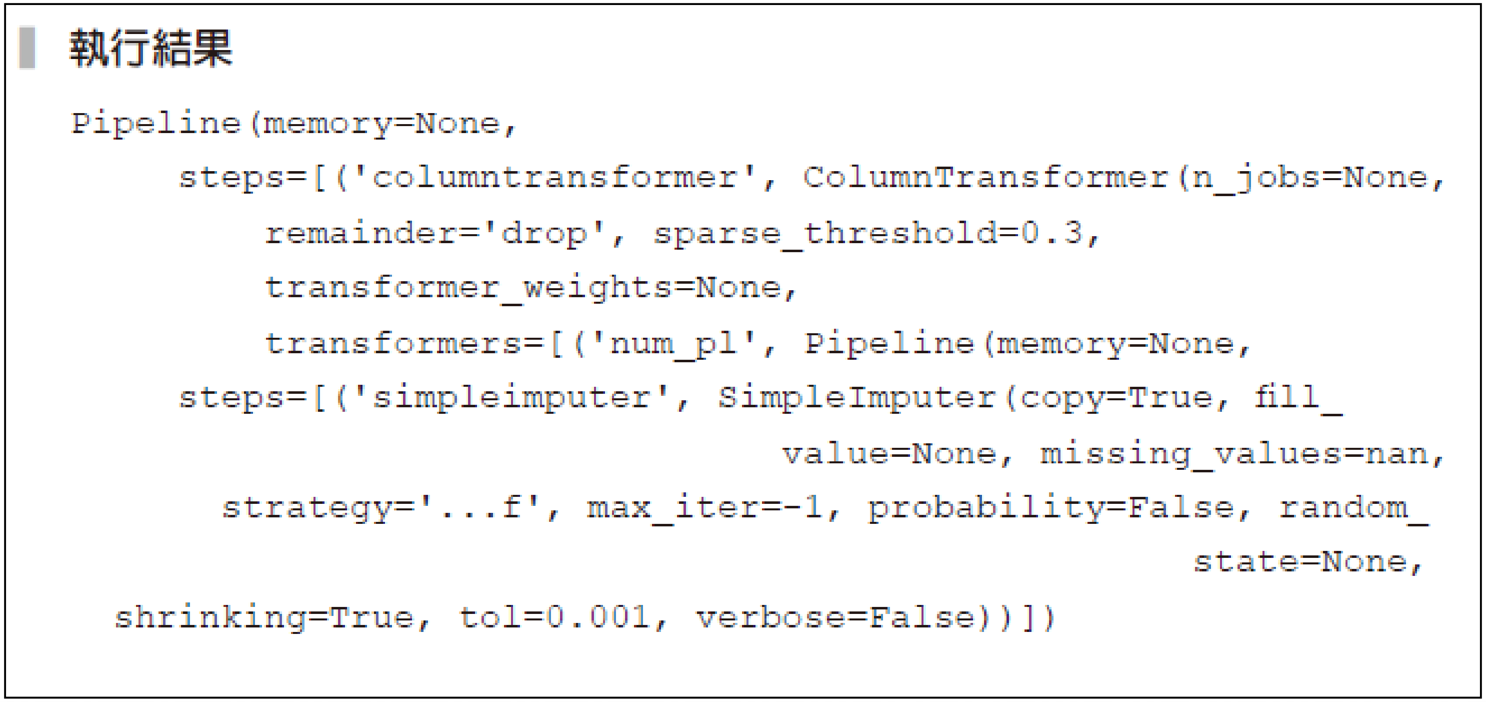
程式碼

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

範例8-17　將支持向量機也加入管道器裡

程式碼

```
from sklearn.svm import SVC
model_pl_svc = make_pipeline(data_pl, SVC())
model_pl_svc
```



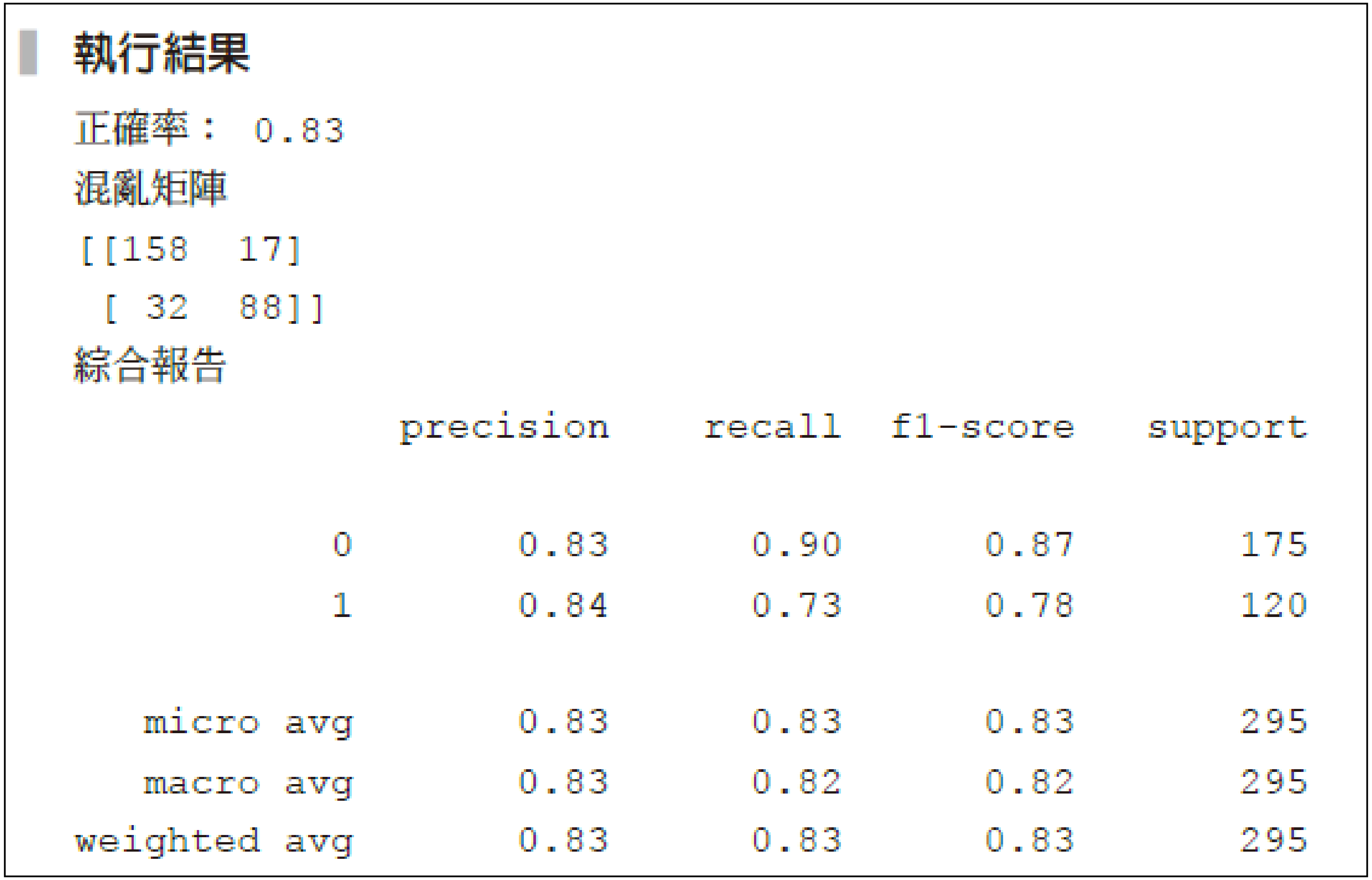
執行結果

```
Pipeline(memory=None,
     steps=[('columntransformer', ColumnTransformer(n_jobs=None,
         remainder='drop', sparse_threshold=0.3,
         transformer_weights=None,
         transformers=[('num_pl', Pipeline(memory=None,
     steps=[('simpleimputer', SimpleImputer(copy=True, fill_
                                 value=None, missing_values=nan,
       strategy='...f', max_iter=-1, probability=False, random_
                                                    state=None,
  shrinking=True, tol=0.001, verbose=False))])
```

範例8-18 用mode_pl_svc來做機器學習和預測

程式碼

```
from sklearn.metrics import confusion_matrix, accuracy_score,
classification_report
model_pl_svc.fit(X_train, y_train)
y_pred = model_pl_svc.predict(X_test)
print('正確率：', accuracy_score(y_test, y_pred).round(2))
print('混亂矩陣')
print(confusion_matrix(y_test, y_pred))
print('綜合報告')
print(classification_report(y_test, y_pred))
```



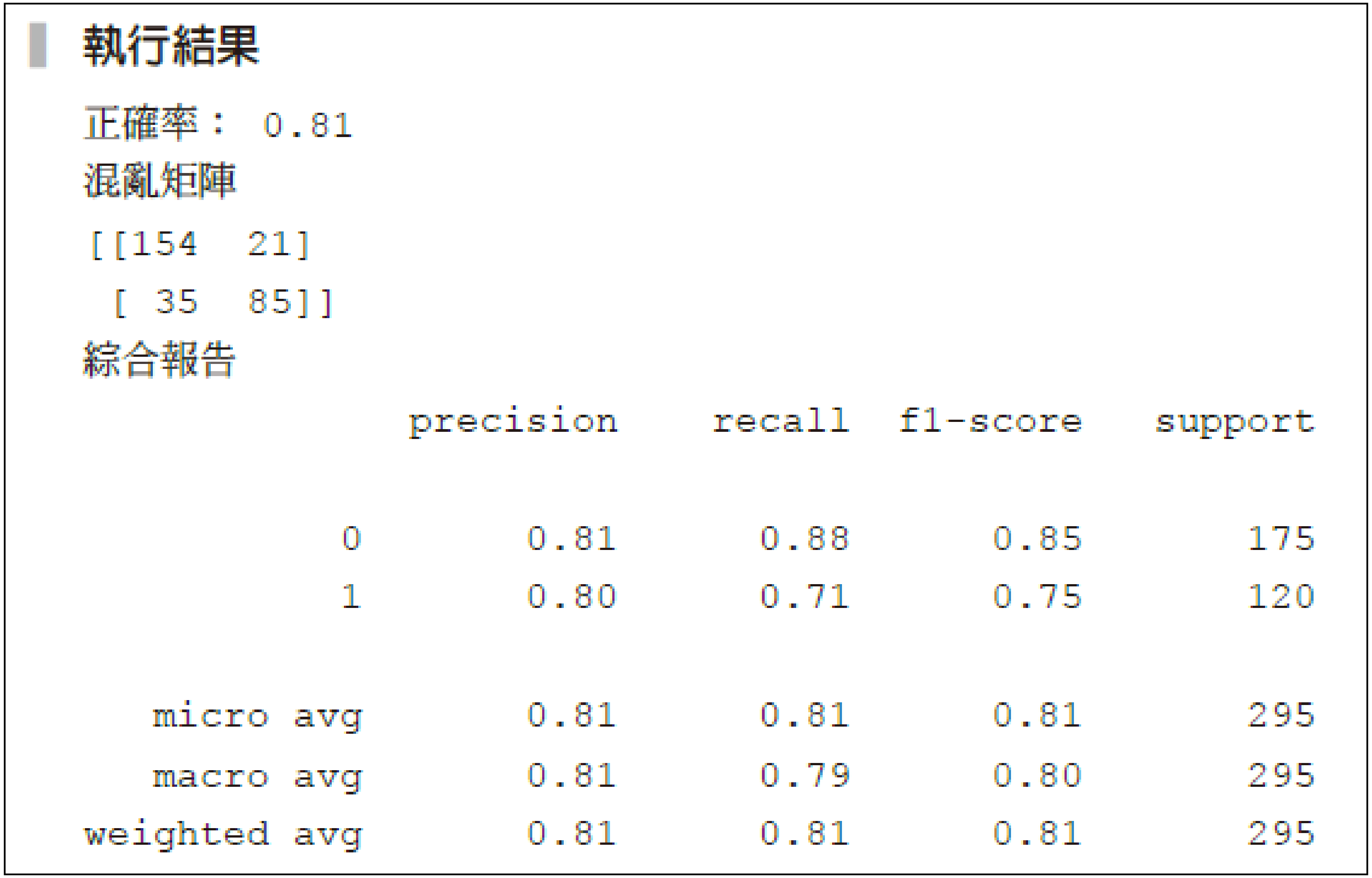
執行結果

```
正確率： 0.83
混亂矩陣
[[158  17]
 [ 32  88]]
綜合報告
              precision    recall  f1-score   support

           0       0.83      0.90      0.87       175
           1       0.84      0.73      0.78       120

   micro avg       0.83      0.83      0.83       295
   macro avg       0.83      0.82      0.82       295
weighted avg       0.83      0.83      0.83       295
```

範例8-19 用羅吉斯迴歸來做預測

程式碼

```
from sklearn.linear_model import LogisticRegression
model_pl_lr = make_pipeline(data_pl, 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(2))
print('混亂矩陣')
print(confusion_matrix(y_test, y_pred))
print('綜合報告')
print(classification_report(y_test, y_pred))
```



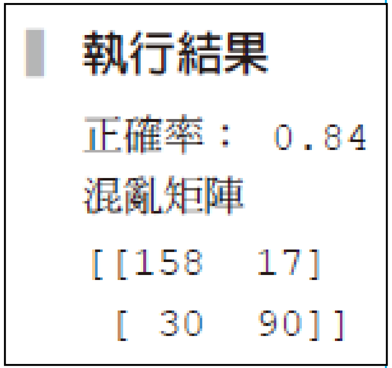
執行結果

```
正確率： 0.81
混亂矩陣
[[154  21]
 [ 35  85]]
綜合報告
              precision    recall  f1-score   support

           0       0.81      0.88      0.85       175
           1       0.80      0.71      0.75       120

   micro avg       0.81      0.81      0.81       295
   macro avg       0.81      0.79      0.80       295
weighted avg       0.81      0.81      0.81       295
```

範例8-20　將'p_class'從類別型管道，移到數值型管道

程式碼

```
data_pl = ColumnTransformer([
    ('num_pl', num_pl, ['Age', 'SibSp', 'Parch', 'Fare', 'Pclass']),
    ('cat_pl', cat_pl, ['Sex', 'Embarked'])
])
model_pl_svc = make_pipeline(data_pl, SVC())
model_pl_svc.fit(X_train, y_train)
y_pred = model_pl_svc.predict(X_test)
print('正確率：', accuracy_score(y_test, y_pred).round(2))
print('混亂矩陣')
print(confusion_matrix(y_test, y_pred))
```



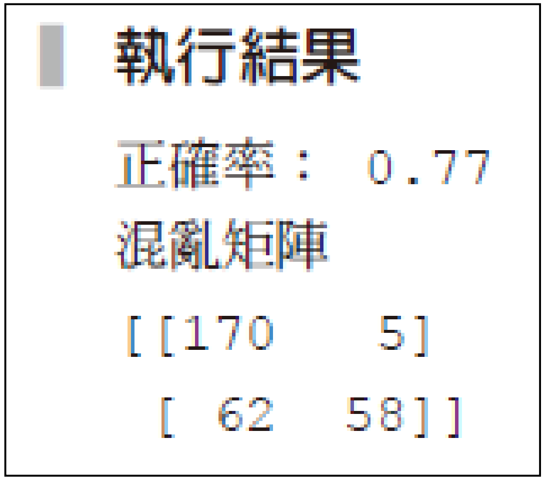
執行結果

```
正確率： 0.84
混亂矩陣
[[158  17]
 [ 30  90]]
```

範例8-21　將SelectKBest加到資料管道器的後端，並選最重要的3個特徵值

程式碼

```
from sklearn.feature_selection import SelectKBest, f_classif
data_pl = ColumnTransformer([
    ('num_pl', num_pl, X_col_num),
    ('cat_pl', cat_pl, X_col_cat)])
model_pl_svc = make_pipeline(data_pl,
                             SelectKBest(f_classif, 3),
                              SVC())
model_pl_svc.fit(X_train, y_train)
y_pred = model_pl_svc.predict(X_test)
print('正確率：', accuracy_score(y_test, y_pred).round(2))
print('混亂矩陣')
print(confusion_matrix(y_test, y_pred))
```



執行結果

```
正確率： 0.77
混亂矩陣
[[170   5]
 [ 62  58]]
```