Chapter 9 - Exercise 2: Titanic

Yêu cầu: Áp dụng Grid Search và Random Search cho bài Titanic đã làm trước đó.

In [1]:
```
# from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
```

In [2]:
```
# %cd '/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter9_KyThuatBo
```

In [3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import math
```

In [4]:
```
data = pd.read_csv("titanic_csv.csv", index_col=0)
```

In [5]:
```
type(data)
```

Out[5]:
```
pandas.core.frame.DataFrame
```

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1309 entries, 1 to 1309
Data columns (total 12 columns):
pclass       1309 non-null int64
survived     1309 non-null int64
name         1309 non-null object
sex          1309 non-null object
age          1046 non-null float64
sibsp        1309 non-null int64
parch        1309 non-null int64
ticket       1309 non-null object
fare         1308 non-null float64
cabin        295 non-null object
embarked     1307 non-null object
home.dest    745 non-null object
dtypes: float64(2), int64(4), object(6)
memory usage: 132.9+ KB
```

In [7]:
```
data.head()
```

Out[7]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked
1	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S
2	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S
3	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S
4	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	S
5	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	S

In [8]:
```
data = data.interpolate()
```

In [9]:
```
X=data[['pclass', 'sex', 'age', 'sibsp',
        'parch', 'fare', 'embarked']]  # Features
y=data['survived']  # Labels
```

In [10]:
```
X = pd.get_dummies(X)
X.head()
```

Out[10]:

	pclass	age	sibsp	parch	fare	sex_female	sex_male	embarked_C	embarked_Q	emb
1	1	29.0000	0	0	211.3375	1	0	0	0	
2	1	0.9167	1	2	151.5500	0	1	0	0	
3	1	2.0000	1	2	151.5500	1	0	0	0	
4	1	30.0000	1	2	151.5500	0	1	0	0	
5	1	25.0000	1	2	151.5500	1	0	0	0	

In [11]:

```
# Tạo lại dữ liệu huấn luyện và test sau khi bỏ đi các thuộc tính ít quan trọng
X_now = X[['age', 'fare', 'sex_female', 'sex_male', 'pclass']]
y_now = data['survived']
```

In [12]:

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X_now, y_now,
                                                    test_size=0.3,
                                                    random_state = 1)
```

Grid Search

In [13]:

```
# Dùng Grid Search
from sklearn.model_selection import GridSearchCV
```

In [14]:

```
param_grid = {
    'n_estimators': [50, 100, 200, 300],
    'max_features': ['auto', 'sqrt', 'log2']
}
```

Có thể dùng tham số đầy đủ như sau:

param_grid = {"max_depth": [2,3, None],
"n_estimators":[50,100,200,300],
"max_features": [1,2,3,4],
"min_samples_split": [2, 3, 10],
"min_samples_leaf": [1, 3, 10],
"bootstrap": [True, False],
"criterion": ["gini", "entropy"]}

In [15]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [16]:

```
CV_rfc = GridSearchCV(
                estimator=RandomForestClassifier(random_state=1),
                      param_grid=param_grid,
                      cv= 5)
```

In [17]:
```
CV_rfc.fit(X_train, y_train)
```

Out[17]:
```
GridSearchCV(cv=5, error_score='raise-deprecating',
             estimator=RandomForestClassifier(bootstrap=True, class_weight=Non
e,
                                              criterion='gini', max_depth=None,
                                              max_features='auto',
                                              max_leaf_nodes=None,
                                              min_impurity_decrease=0.0,
                                              min_impurity_split=None,
                                              min_samples_leaf=1,
                                              min_samples_split=2,
                                              min_weight_fraction_leaf=0.0,
                                              n_estimators='warn', n_jobs=None,
                                              oob_score=False, random_state=1,
                                              verbose=0, warm_start=False),
             iid='warn', n_jobs=None,
             param_grid={'max_features': ['auto', 'sqrt', 'log2'],
                         'n_estimators': [50, 100, 200, 300]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)
```

In [18]:
```
print(CV_rfc.best_params_)
```

```
{'max_features': 'auto', 'n_estimators': 100}
```

In [19]:
```
y_pred=CV_rfc.predict(X_test)
```

In [20]:
```
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.7735368956743003
```

In [21]:
```
# Kiểm tra độ chính xác
print("The Training R^2 score is: ",
      CV_rfc.score(X_train,y_train)*100,"%")
print("The Testing R^2 score is: ",
      CV_rfc.score(X_test,y_test)*100,"%")
```

```
The Training R^2 score is:  98.47161572052401 %
The Testing R^2 score is:  77.35368956743002 %
```

Random Search

In [22]:
```
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import randint as sp_randint
param_dist = {'n_estimators': [50, 100, 200, 300],
              'max_features': ['auto', 'sqrt', 'log2']}
```

In [23]:
```
forest_random = RandomizedSearchCV(
    estimator=RandomForestClassifier(random_state=1),
                                   param_distributions=param_dist,
                                   cv=5)
```

In [24]:
```
forest_random.fit(X_train,y_train)
```

Out[24]:
```
RandomizedSearchCV(cv=5, error_score='raise-deprecating',
                   estimator=RandomForestClassifier(bootstrap=True,
                                                    class_weight=None,
                                                    criterion='gini',
                                                    max_depth=None,
                                                    max_features='auto',
                                                    max_leaf_nodes=None,
                                                    min_impurity_decrease=0.0,
                                                    min_impurity_split=None,
                                                    min_samples_leaf=1,
                                                    min_samples_split=2,
                                                    min_weight_fraction_leaf=0.
0,
                                                    n_estimators='warn',
                                                    n_jobs=None,
                                                    oob_score=False,
                                                    random_state=1, verbose=0,
                                                    warm_start=False),
                   iid='warn', n_iter=10, n_jobs=None,
                   param_distributions={'max_features': ['auto', 'sqrt',
                                                         'log2'],
                                        'n_estimators': [50, 100, 200, 300]},
                   pre_dispatch='2*n_jobs', random_state=None, refit=True,
                   return_train_score=False, scoring=None, verbose=0)
```

In [25]:
```
forest_random_best = forest_random.best_estimator_
print("Best Model Parameter: ",forest_random.best_params_)
```

```
Best Model Parameter:  {'n_estimators': 100, 'max_features': 'auto'}
```

In [26]:
```
y_pred=forest_random.predict(X_test)
```

In [27]:
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.7735368956743003
```

In [28]:
```
# Kiểm tra độ chính xác
print("The Training R^2 score is: ",
      forest_random.score(X_train,y_train)*100,"%")
print("The Testing R^2 score is: ",
      forest_random.score(X_test,y_test)*100,"%")
```

```
The Training R^2 score is:  98.47161572052401 %
The Testing R^2 score is:  77.35368956743002 %
```

In [29]:
```
# Model vẫn bị overfitting
```

Bổ sung sau khi học chapter 9: Lựa chọn 1 model phù hợp cho dataset này dựa trên các model đã học.

In [ ]: