assignment_24.1

January 12, 2019

0.1 Predicting Survival using Titanic Dataset

In this assignment, I have used decision tree to predict survival of passengers using Titanic Dataset

0.2 Import libraries

```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sb
        import matplotlib.pyplot as plt
        import sklearn
        from pandas import Series, DataFrame
        from pylab import rcParams
        from sklearn import preprocessing
        from sklearn.linear_model import LogisticRegression
        from sklearn.cross_validation import train_test_split
        from sklearn import metrics
        from sklearn.metrics import classification_report
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.metrics import accuracy_score,recall_score,precision_score
```

```
E:\anaconda\lib\site-packages\sklearn\cross_validation.py:41: DeprecationWarning: This module w
  "This module will be removed in 0.20.", DeprecationWarning)
```

0.3 Load titanic dataset

```
In [2]: url= "https://raw.githubusercontent.com/BigDataGal/Python-for-Data-Science/master/titan
        titanic = pd.read_csv(url)

In [3]: titanic.head()
```

Out[3]:

	PassengerId	Survived	Pclass	\
0	1	0	3	
1	2	1	1	
2	3	1	3	
3	4	1	1	
4	5	0	3	

```
                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

0.4 Perform analysis of dataset (describe, event rate)

In [4]:
```
titanic.describe()
```

Out[4]:
```
        PassengerId    Survived      Pclass         Age       SibSp  \
count    891.000000  891.000000  891.000000  714.000000  891.000000
mean     446.000000    0.383838    2.308642   29.699118    0.523008
std      257.353842    0.486592    0.836071   14.526497    1.102743
min        1.000000    0.000000    1.000000    0.420000    0.000000
25%      223.500000    0.000000    2.000000   20.125000    0.000000
50%      446.000000    0.000000    3.000000   28.000000    0.000000
75%      668.500000    1.000000    3.000000   38.000000    1.000000
max      891.000000    1.000000    3.000000   80.000000    8.000000

             Parch        Fare
count  891.000000  891.000000
mean     0.381594   32.204208
std      0.806057   49.693429
min      0.000000    0.000000
25%      0.000000    7.910400
50%      0.000000   14.454200
75%      0.000000   31.000000
max      6.000000  512.329200
```

In [5]:
```
event_rate = (sum(titanic.loc[titanic['Survived']==1,'Survived'] )/titanic.shape[0])*1
print("Event_Rate: " + str(event_rate) + "%")
```

```
Event_Rate: 38.38383838383838%
```

0.5 Find missing values in columns and fill the missing values

In [6]:
```
titanic.isnull().sum()
```

```
Out[6]: PassengerId      0
        Survived         0
        Pclass           0
        Name             0
        Sex              0
        Age            177
        SibSp            0
        Parch            0
        Ticket           0
        Fare             0
        Cabin          687
        Embarked         2
        dtype: int64
```

```
In [7]: ## Fill mssing value in Age column with average Age
        mean_age = titanic['Age'].mean()
        titanic['Age'].fillna(mean_age, inplace=True)
```

```
In [8]: ## Fill mssing value in Cabin column with NaN
        titanic['Cabin'].fillna('NaN', inplace=True)
```

```
In [9]: ## Fill mssing value in Embarked column with NaN
        titanic['Embarked'].fillna('NA', inplace=True)
```

0.6 Perform one hot encoding of columns Sex, Embarked, Cabin

```
In [10]: titanic = pd.get_dummies(titanic, columns=['Sex'])
```

```
In [11]: titanic = pd.get_dummies(titanic, columns=['Embarked'])
```

0.7 Add a new column has_Cabin and populate with 0 if Cabin value is NaN, 1 otherwise

```
In [12]: titanic['has_Cabin'] =  titanic['Cabin'].apply(lambda x: 0 if x == 'NaN' else 1)
```

```
In [13]: titanic.head()
```

```
Out[13]:    PassengerId  Survived  Pclass  \
         0            1         0       3
         1            2         1       1
         2            3         1       3
         3            4         1       1
         4            5         0       3

                                                         Name   Age  SibSp  Parch  \
         0                             Braund, Mr. Owen Harris  22.0      1      0
         1  Cumings, Mrs. John Bradley (Florence Briggs Th...  38.0      1      0
         2                              Heikkinen, Miss. Laina  26.0      0      0
         3        Futrelle, Mrs. Jacques Heath (Lily May Peel)  35.0      1      0
```

```
4                            Allen, Mr. William Henry  35.0      0      0

             Ticket     Fare Cabin  Sex_female  Sex_male  Embarked_C  \
0         A/5 21171   7.2500   NaN           0         1           0
1          PC 17599  71.2833   C85           1         0           1
2  STON/O2. 3101282   7.9250   NaN           1         0           0
3            113803  53.1000  C123           1         0           0
4            373450   8.0500   NaN           0         1           0

   Embarked_NA  Embarked_Q  Embarked_S  has_Cabin
0            0           0           1          0
1            0           0           0          1
2            0           0           1          0
3            0           0           1          1
4            0           0           1          0
```

0.8 Praprare data for Decision Tree analysis

1. Create a dataframe titanic_features From titanic dataframe by dropping columns 'PassengerId','Name','Ticket','Survived','Cabin'
2. Create a dataframe titanic_target by taking colmn 'Survived' from titanic dataframe
3. Split the dataframes titanic_features and titanic_target into train and tests data

```
In [14]: titanic_features = titanic.drop(['PassengerId','Name','Ticket','Survived','Cabin'], ax
         # titanic_features=titanic[['Pclass', 'Sex_female', 'Sex_male', 'Age', 'SibSp' , 'Par
         titanic_target=titanic[['Survived']]
```

```
In [15]: # Import train_test_split
         from sklearn.cross_validation import train_test_split

         # Split the 'titanic_features' and 'titanic_target' data into training and testing se
         X_train, X_test, y_train, y_test = train_test_split(titanic_features,
                                                             titanic_target,
                                                             test_size = 0.3,
                                                             random_state = 0)


         # Show the results of the split
         print("Training set has {} samples.".format(X_train.shape[0]))
         print("Testing set has {} samples.".format(X_test.shape[0]))
```

```
Training set has 623 samples.
Testing set has 268 samples.
```

0.9 Train data using Devision Tree and find accuracy scores

```
In [16]: model = DecisionTreeClassifier()
         model.fit(X_train,y_train)
```

```
Out[16]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                                max_features=None, 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, presort=False, random_state=None,
                                splitter='best')

In [17]: y_predict = model.predict(X_test)
         acc_score1 = accuracy_score(y_test,y_predict)
         print('accuracy_score: '+str(acc_score1))

accuracy_score: 0.7835820895522388


In [18]: ## Final accuracy score of survival is 78.35%
```