LogReg_Titanic

October 15, 2024

[1]:
```
import pandas as pd, numpy as np
import matplotlib.pyplot as plt, seaborn as sns

titanic = sns.load_dataset('titanic')
titanic.head()
```

[1]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First
2         1       3  female  26.0      0      0   7.9250        S  Third
3         1       1  female  35.0      1      0  53.1000        S  First
4         0       3    male  35.0      0      0   8.0500        S  Third

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
2  woman       False  NaN  Southampton   yes   True
3  woman       False    C  Southampton   yes  False
4    man        True  NaN  Southampton    no   True
```

[2]:
```
titanic.shape
```

[2]:
```
(891, 15)
```

[3]:
```
titanic.isnull().sum()
```

[3]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
```

```
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

[4]:
```
titanic = titanic.dropna(subset=['age','embarked', 'deck','embark_town'])
```

[5]:
```
titanic.head()
```

[5]:
```
    survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
1          1       1  female  38.0      1      0  71.2833        C  First
3          1       1  female  35.0      1      0  53.1000        S  First
6          0       1    male  54.0      0      0  51.8625        S  First
10         1       3  female   4.0      1      1  16.7000        S  Third
11         1       1  female  58.0      0      0  26.5500        S  First

      who  adult_male deck  embark_town alive  alone
1   woman       False    C    Cherbourg   yes  False
3   woman       False    C  Southampton   yes  False
6     man        True    E  Southampton    no   True
10  child       False    G  Southampton   yes  False
11  woman       False    C  Southampton   yes   True
```

[6]:
```
titanic.shape
```

[6]:
```
(182, 15)
```

[7]:
```
titanic.sex.value_counts()
```

[7]:
```
male      94
female    88
Name: sex, dtype: int64
```

[8]:
```
titanic.embarked.value_counts()
```

[8]:
```
S    115
C     65
Q      2
Name: embarked, dtype: int64
```

[9]:
```
# Converting the categorical cols into numerical

titanic['sex'] = titanic['sex'].map({'male':0, 'female':1})
titanic['embarked'] = titanic['embarked'].map({'S':0,'C':1,'Q':2})
```

[10]:
```
titanic.head()
```

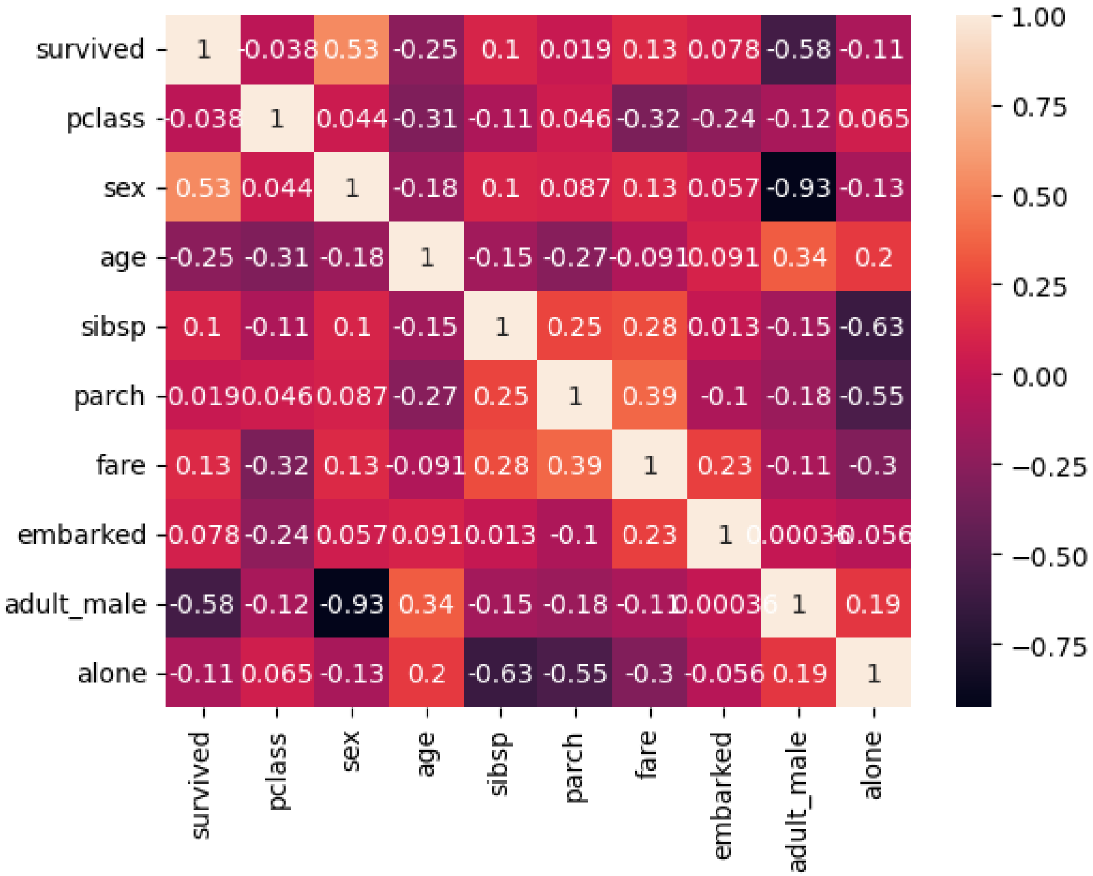
```
[10]:     survived  pclass  sex   age  sibsp  parch     fare  embarked  class  \
      1          1       1    1  38.0      1      0  71.2833         1  First
      3          1       1    1  35.0      1      0  53.1000         0  First
      6          0       1    0  54.0      0      0  51.8625         0  First
      10         1       3    1   4.0      1      1  16.7000         0  Third
      11         1       1    1  58.0      0      0  26.5500         0  First

            who  adult_male deck  embark_town alive  alone
      1   woman       False    C    Cherbourg   yes  False
      3   woman       False    C  Southampton   yes  False
      6     man        True    E  Southampton    no   True
      10  child       False    G  Southampton   yes  False
      11  woman       False    C  Southampton   yes   True
```

```
[11]: sns.heatmap(titanic.corr(), annot = True)
```

```
/tmp/ipykernel_6220/3283234111.py:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  sns.heatmap(titanic.corr(), annot = True)
```

```
[11]: <Axes: >
```



[12]:
```
# identify which is X and y

X = titanic[['sex', 'age', 'pclass', 'sibsp', 'parch', 'fare','embarked']]
y = titanic['survived']
```

[13]:
```
# train test split

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,␣
  ↪classification_report
```

[14]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,␣
  ↪random_state = 42)
```

[15]:
```
# Initialize and fit the Logistic regression model

logreg = LogisticRegression(max_iter = 1000)
logreg.fit(X_train, y_train)
```

```
[15]: LogisticRegression(max_iter=1000)
```

[33]:

```
y_pred = logreg.predict(X_test)
y_pred
```

```
[33]: array([1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0,
             0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1])
```

[51]:

```
# Evalute the model

accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy : {accuracy *100:.2f}%')
```

```
Accuracy : 62.16%
```

[53]:

```
# confusion matrix

conf_matrix = confusion_matrix(y_test, y_pred)
print(conf_matrix)
```

```
[[ 8  5]
 [ 9 15]]
```

[55]:

```
print("Classification Report:",classification_report(y_test, y_pred) )
```

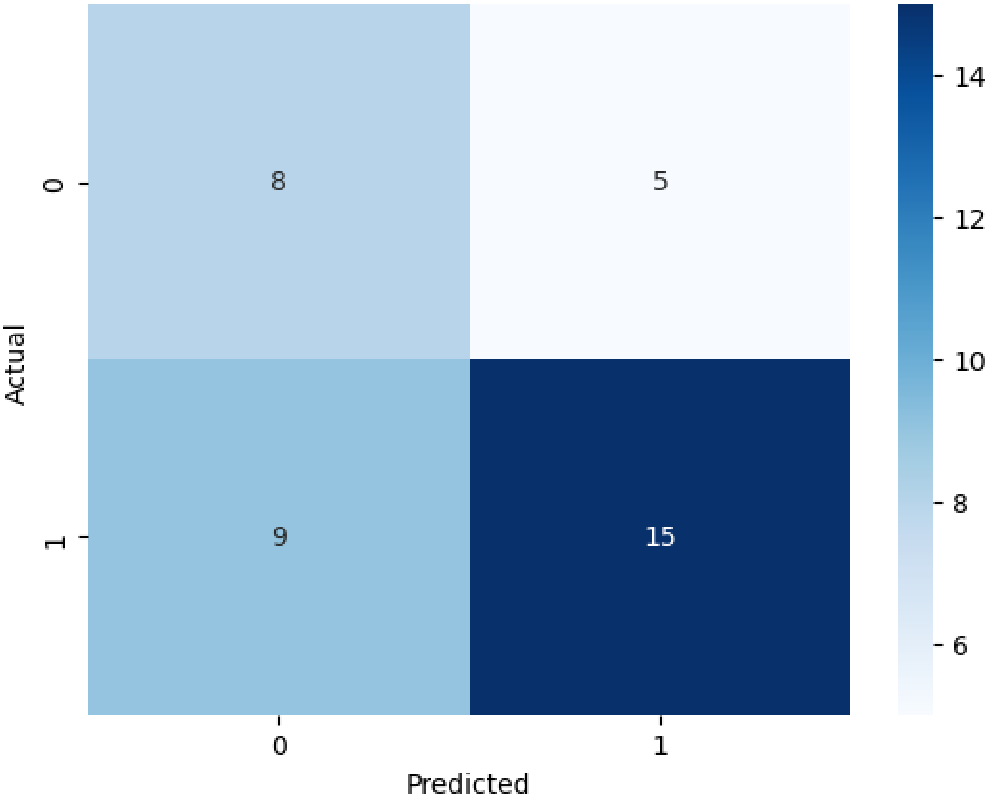
```
Classification Report:               precision    recall  f1-score   support

           0       0.47      0.62      0.53        13
           1       0.75      0.62      0.68        24

    accuracy                           0.62        37
   macro avg       0.61      0.62      0.61        37
weighted avg       0.65      0.62      0.63        37
```

[63]:

```
sns.heatmap(conf_matrix, annot = True, fmt = 'd', cmap = 'Blues')
plt.ylabel("Actual")
plt.xlabel("Predicted")
plt.show()
```


8
5
9
15
0
1
Actual
0
1
Predicted
14
12
10
8
6

```
[ ]:
```