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

%matplotlib inline

#### Importing Libraries
```

In [2]:
```
#### Reading a data Set

titanic_data = pd.read_csv('titanic_full_data.csv')

### Checking columns name in dataset

titanic_data.columns
```

Out[2]:
```
Index(['passenger_id', 'pclass', 'name', 'sex', 'age', 'sibsp', 'parch',
       'ticket', 'fare', 'cabin', 'embarked', 'boat', 'body', 'home.dest',
       'survived'],
      dtype='object')
```

In [3]:
```
titanic_data.head(2)
```

Out[3]:

	passenger_id	pclass	name	sex	age	sibsp	parch	ticket	fare	cabin	embark
0	0	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	

In [4]:
```python
### Getting  dummies colums of required columns

titanic_data = pd.get_dummies(titanic_data,columns=['pclass','embarked','sex'])


titanic_data
```

```
0   0    PC 17483  221.7792  C97    8     NaN            NaN   1.0   1   0   0

                                                         San
0   0       13905   26.0000  NaN  NaN   148.0     Francisco,   0.0   1   0   0
                                                          CA

                                                   Dowagiac,
1   0       11967   91.0792  B49    7     NaN             MI   1.0   1   0   0

                                                   Dowagiac,
1   0       11967   91.0792  B49    7     NaN             MI   1.0   1   0   0

0   0    PC 17760  135.6333  C99    8     NaN            NaN   NaN   1   0   0

                                                  Stockholm,
                                                    Sweden /
0   0      110564   26.5500  C52    D     NaN    Washington,   1.0   1   0   0
                                                          DC

...  ...       ...       ...  ...  ...    ...            ...   ...  ...  ...  ...
```

In [5]:
```
### Dropping useless Columns

titanic_data.drop(['ticket','pclass_1','embarked_C','sex_female','name','passeng

titanic_data
```

Out[5]:

	age	sibsp	fare	survived	pclass_2	pclass_3	embarked_Q	embarked_S	sex_male
0	29.0000	0	211.3375	0.0	0	0	0	1	0
1	0.9167	1	151.5500	1.0	0	0	0	1	1
2	2.0000	1	151.5500	0.0	0	0	0	1	0
3	30.0000	1	151.5500	0.0	0	0	0	1	1
4	25.0000	1	151.5500	0.0	0	0	0	1	0
5	48.0000	0	26.5500	1.0	0	0	0	1	1
6	63.0000	1	77.9583	NaN	0	0	0	1	0
7	39.0000	0	0.0000	NaN	0	0	0	1	1
8	53.0000	2	51.4792	1.0	0	0	0	1	0
9	71.0000	0	49.5042	NaN	0	0	0	0	1
10	47.0000	1	227.5250	0.0	0	0	0	0	1
11	18.0000	1	227.5250	1.0	0	0	0	0	0
12	24.0000	0	69.3000	1.0	0	0	0	0	0
13	26.0000	0	78.8500	1.0	0	0	0	1	0
14	80.0000	0	30.0000	1.0	0	0	0	1	1
15	NaN	0	25.9250	0.0	0	0	0	1	1
16	24.0000	0	247.5208	NaN	0	0	0	0	1
17	50.0000	0	247.5208	1.0	0	0	0	0	0
18	32.0000	0	76.2917	1.0	0	0	0	0	0
19	36.0000	0	75.2417	0.0	0	0	0	0	1
20	37.0000	1	52.5542	NaN	0	0	0	1	1
21	47.0000	1	52.5542	1.0	0	0	0	1	0
22	26.0000	0	30.0000	NaN	0	0	0	0	1
23	42.0000	0	227.5250	1.0	0	0	0	0	0
24	29.0000	0	221.7792	1.0	0	0	0	1	0
25	25.0000	0	26.0000	0.0	0	0	0	0	1
26	25.0000	1	91.0792	1.0	0	0	0	0	1
27	19.0000	1	91.0792	1.0	0	0	0	0	0
28	35.0000	0	135.6333	NaN	0	0	0	1	0
29	28.0000	0	26.5500	1.0	0	0	0	1	1
...	...	...	...	...	...	...	...	...	...

	age	sibsp	fare	survived	pclass_2	pclass_3	embarked_Q	embarked_S	sex_male
1279	14.0000	0	7.8542	0.0	0	1	0	1	0
1280	22.0000	0	7.8958	NaN	0	1	0	1	1
1281	22.0000	0	9.0000	0.0	0	1	0	1	1
1282	NaN	0	8.0500	0.0	0	1	0	1	1
1283	NaN	0	7.5500	NaN	0	1	0	1	1
1284	NaN	0	8.0500	0.0	0	1	0	1	1
1285	32.5000	0	9.5000	NaN	0	1	0	1	1
1286	38.0000	0	7.2292	1.0	0	1	0	0	0
1287	51.0000	0	7.7500	0.0	0	1	0	1	1
1288	18.0000	1	6.4958	0.0	0	1	0	1	1
1289	21.0000	1	6.4958	NaN	0	1	0	1	1
1290	47.0000	1	7.0000	NaN	0	1	0	1	0
1291	NaN	0	8.7125	0.0	0	1	0	1	1
1292	NaN	0	7.5500	NaN	0	1	0	1	1
1293	NaN	0	8.0500	0.0	0	1	0	1	1
1294	28.5000	0	16.1000	0.0	0	1	0	1	1
1295	21.0000	0	7.2500	NaN	0	1	0	1	1
1296	27.0000	0	8.6625	NaN	0	1	0	1	1
1297	NaN	0	7.2500	NaN	0	1	0	1	1
1298	36.0000	0	9.5000	0.0	0	1	0	1	1
1299	27.0000	1	14.4542	0.0	0	1	0	0	1
1300	15.0000	1	14.4542	1.0	0	1	0	0	0
1301	45.5000	0	7.2250	0.0	0	1	0	0	1
1302	NaN	0	7.2250	0.0	0	1	0	0	1
1303	NaN	0	14.4583	0.0	0	1	0	0	1
1304	14.5000	1	14.4542	0.0	0	1	0	0	0
1305	NaN	1	14.4542	NaN	0	1	0	0	0
1306	26.5000	0	7.2250	0.0	0	1	0	0	1
1307	27.0000	0	7.2250	0.0	0	1	0	0	1
1308	29.0000	0	7.8750	NaN	0	1	0	1	1

1309 rows × 9 columns

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

Out[6]:
```
age            263
sibsp            0
fare             1
survived       458
pclass_2         0
pclass_3         0
embarked_Q       0
embarked_S       0
sex_male         0
dtype: int64
```

In [7]:
```
titanic_data=titanic_data.fillna(method='ffill')
```

In [8]:
```
titanic_data.tail(5)
```

Out[8]:

	age	sibsp	fare	survived	pclass_2	pclass_3	embarked_Q	embarked_S	sex_male
1304	14.5	1	14.4542	0.0	0	1	0	0	0
1305	14.5	1	14.4542	0.0	0	1	0	0	0
1306	26.5	0	7.2250	0.0	0	1	0	0	1
1307	27.0	0	7.2250	0.0	0	1	0	0	1
1308	29.0	0	7.8750	0.0	0	1	0	1	1

In [9]:
```
titanic_data.isnull().sum()
```

Out[9]:
```
age          0
sibsp        0
fare         0
survived     0
pclass_2     0
pclass_3     0
embarked_Q   0
embarked_S   0
sex_male     0
dtype: int64
```

In [10]:
```
Y = titanic_data['survived']
X = titanic_data.drop('survived',axis=1)
X
```

Out[10]:

	age	sibsp	fare	pclass_2	pclass_3	embarked_Q	embarked_S	sex_male
0	29.0000	0	211.3375	0	0	0	1	0
1	0.9167	1	151.5500	0	0	0	1	1
2	2.0000	1	151.5500	0	0	0	1	0
3	30.0000	1	151.5500	0	0	0	1	1
4	25.0000	1	151.5500	0	0	0	1	0
5	48.0000	0	26.5500	0	0	0	1	1
6	63.0000	1	77.9583	0	0	0	1	0
7	39.0000	0	0.0000	0	0	0	1	1
8	53.0000	2	51.4792	0	0	0	1	0
9	71.0000	0	49.5042	0	0	0	0	1
10	47.0000	1	227.5250	0	0	0	0	1
11	18.0000	1	227.5250	0	0	0	0	0
12	24.0000	0	69.3000	0	0	0	0	0
13	26.0000	0	78.8500	0	0	0	1	0
14	80.0000	0	30.0000	0	0	0	1	1
15	80.0000	0	25.9250	0	0	0	1	1
16	24.0000	0	247.5208	0	0	0	0	1
17	50.0000	0	247.5208	0	0	0	0	0
18	32.0000	0	76.2917	0	0	0	0	0
19	36.0000	0	75.2417	0	0	0	0	1
20	37.0000	1	52.5542	0	0	0	1	1
21	47.0000	1	52.5542	0	0	0	1	0
22	26.0000	0	30.0000	0	0	0	0	1
23	42.0000	0	227.5250	0	0	0	0	0
24	29.0000	0	221.7792	0	0	0	1	0
25	25.0000	0	26.0000	0	0	0	0	1
26	25.0000	1	91.0792	0	0	0	0	1
27	19.0000	1	91.0792	0	0	0	0	0
28	35.0000	0	135.6333	0	0	0	1	0
29	28.0000	0	26.5500	0	0	0	1	1
...	...	...	...	...	...	...	...	...
1279	14.0000	0	7.8542	0	1	0	1	0
1280	22.0000	0	7.8958	0	1	0	1	1

	age	sibsp	fare	pclass_2	pclass_3	embarked_Q	embarked_S	sex_male
1281	22.0000	0	9.0000	0	1	0	1	1
1282	22.0000	0	8.0500	0	1	0	1	1
1283	22.0000	0	7.5500	0	1	0	1	1
1284	22.0000	0	8.0500	0	1	0	1	1
1285	32.5000	0	9.5000	0	1	0	1	1
1286	38.0000	0	7.2292	0	1	0	0	0
1287	51.0000	0	7.7500	0	1	0	1	1
1288	18.0000	1	6.4958	0	1	0	1	1
1289	21.0000	1	6.4958	0	1	0	1	1
1290	47.0000	1	7.0000	0	1	0	1	0
1291	47.0000	0	8.7125	0	1	0	1	1
1292	47.0000	0	7.5500	0	1	0	1	1
1293	47.0000	0	8.0500	0	1	0	1	1
1294	28.5000	0	16.1000	0	1	0	1	1
1295	21.0000	0	7.2500	0	1	0	1	1
1296	27.0000	0	8.6625	0	1	0	1	1
1297	27.0000	0	7.2500	0	1	0	1	1
1298	36.0000	0	9.5000	0	1	0	1	1
1299	27.0000	1	14.4542	0	1	0	0	1
1300	15.0000	1	14.4542	0	1	0	0	0
1301	45.5000	0	7.2250	0	1	0	0	1
1302	45.5000	0	7.2250	0	1	0	0	1
1303	45.5000	0	14.4583	0	1	0	0	1
1304	14.5000	1	14.4542	0	1	0	0	0
1305	14.5000	1	14.4542	0	1	0	0	0
1306	26.5000	0	7.2250	0	1	0	0	1
1307	27.0000	0	7.2250	0	1	0	0	1
1308	29.0000	0	7.8750	0	1	0	1	1

1309 rows × 8 columns

In [11]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

X_train , X_test , Y_train , Y_test = train_test_split(X,Y,test_size = .3)

logreg = LogisticRegression()

logreg.fit(X_train,Y_train)
```

```
C:\Users\Lenovo\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:43
3: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a
solver to silence this warning.
  FutureWarning)
```

Out[11]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [12]:
```
print(X_train.shape)
print(Y_train.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(916, 8)
(916,)
(393, 8)
(393,)
```

In [13]:
```
predictions = logreg.predict(X_test)
predictions
```

Out[13]:
```
array([0., 1., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
       1., 0., 1., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 1., 0., 0., 0.,
       0., 0., 0., 0., 1., 0., 1., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1.,
       1., 1., 0., 1., 1., 1., 1., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 1., 0., 1., 1., 0., 1., 0., 0., 0., 1., 1., 0., 0., 1., 1.,
       1., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 1., 1., 0., 0., 1., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.,
       0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 1., 0., 1., 1., 0., 0., 0., 0., 0., 0., 1., 0., 1., 1., 1.,
       0., 1., 0., 1., 0., 0., 0., 0., 1., 0., 1., 1., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 1., 0., 0., 1., 0., 0.,
       0., 0., 0., 0., 1., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 0., 0.,
       0., 0., 0., 0., 1., 0., 0., 1., 1., 0., 0., 0., 0., 1., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 0.,
       0., 1., 0., 1., 0., 0., 0., 1., 1., 0., 0., 1., 0., 1., 0., 0., 0.,
       0., 0., 0., 1., 0., 0., 1., 1., 1., 0., 0., 0., 1., 0., 0., 0., 0.,
       1., 0., 0., 1., 1., 1., 0., 1., 0., 0., 1., 0., 0., 1., 0., 0., 0.,
       0., 0., 0., 1., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       1., 0., 0., 1., 0., 0., 1., 0., 0., 0., 1., 1., 1., 0., 1., 0., 0.,
       0., 0., 0., 1., 0., 1., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 1., 1., 1.,
       1., 0., 1., 0., 0., 0., 1., 0., 1., 0., 0., 0., 1., 0., 1., 0., 0.,
       1., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
       1., 0.])
```

In [14]:
```
from sklearn.metrics import confusion_matrix,r2_score

print(r2_score(predictions,Y_test))

confusion_matrix(Y_test,predictions)
```

```
-0.3288583863408103
```

Out[14]:
```
array([[217,  28],
       [ 73,  75]], dtype=int64)
```

In [15]:
```
from sklearn.metrics import classification_report,accuracy_score
print(accuracy_score(Y_test,predictions))

classification_report(Y_test,predictions)
```

```
0.7430025445292621
```

Out[15]:
```
'              precision    recall  f1-score   support\n\n         0.0       0.
75      0.89      0.81       245\n         1.0       0.73      0.51      0.60
148\n\n   micro avg       0.74      0.74      0.74       393\n   macro avg
0.74      0.70      0.70       393\nweighted avg       0.74      0.74      0.73
393\n'
```

In [ ]: