Project: Machine Learning on Titanic Dataset by using Random Forest

Submitted By

ADAIKKAPPAN A (EBEON0522601032)

ABSTRACT

On April 15, 1912, the largest passenger liner ever made collided with an iceberg during her maiden voyage. When the Titanic sank it killed 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships. One of the reasons that the shipwreck resulted in such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others.

CONTENTS

TABLE OF CONTENTS

Chapter 1	Introductions: Scenario & Goals	4
Chapter 2	Features & Predictor	5
Chapter 3	Methodology: (i). Data Cleaning Pre-processing (iii). Implementation Steps	6
Chapter 4	Analysis of the Result	16
Chapter 5	Conclusions	19
	References	20

Chapter 1

Introduction

The original Titanic dataset, describing the survival status of individual passengers on the Titanic. The titanic data does not contain information from the crew, but it does contain actual ages of half of the passengers. The principal source for data about Titanic passengers is the Encyclopaedia Titanic. The datasets used here were begun by a variety of researchers. One of the original sources is Eaton & Haas (1994) Titanic: Triumph and Tragedy, Patrick Stephens Ltd, which includes a passenger list created by many researchers and edited by Michael A. Findlay.

Thomas Cason of UVa has greatly updated and improved the titanic data frame using the Encyclopaedia Titanic and created the dataset here. Now the titanic data was downloaded and used from the Kaggle website (Author Name: BRENDA N). Some Categorial features have been changed to continuous features, many errors corrected, many missing ages filled in.

Scenario:

As a Data Analyst Intern, I am interested to learn machine learning algorithms that enables me to use titanic dataset to predict the passenger survived or not in the titanic.

Goal:

- Examine the percentage of the male passenger survived vs female passenger.
- Examine which class of passenger having the highest no. of survived count.
- Examine which Embarked having the highest no. of survived count.
- Find the overall accuracy of the titanic dataset.

Chapter 2

Features & Predictor

StatewiseTestingDetails.csv Dataset

- PassengerId - int64
- Survived - int64
- Pclass - int64
- Name - object
- Sex - object
- Age - float64
- SibSp - int64
- Parch - int64
- Ticket - object
- Fare - float64
- Cabin - object
- Embarked - object

Note

- Total - 12 columns
- Decimal – 7 - Continuous: Which is quantitative data that can be measured.
- String – 5 - Ordinal Data: Categorical data that has an order to it.

Chapter 3
Methodology

Data Cleaning and Pre-processing:

The datasets which were collected from Titanic Dataset from Kaggle website contain unfiltered data which must be filtered before the final data set can be used to do analysis. Also, data has some categorical variables which must be modified into numerical values for which we used Panda's library of Python. In data cleaning step, first we checked whether there are any missing or junk values in the dataset for which we used the is null () function.

Implementation Steps:

As we already discussed in the methodology section about some of the implementation details. So, the language used in this project is Python programming. We're running python code in anaconda navigator's Jupyter notebook. Jupyter notebook is much faster than Python IDE tools like PyCharm or Visual studio for implementing ML algorithms. The advantage of Jupyter notebook is that while writing code, it's really helpful for Data visualization and plotting some graphs like histogram and heatmap of correlated matrices. Let's revise implementation steps: a) Dataset collection. b) Importing Libraries: NumPy, Pandas, Matplotlib,Seaborn and sklearn libraries were used. c) Exploratory data analysis: For getting more insights about data. d) Data cleaning and pre-processing: Checked for null and junk values using isnull() and isna().sum() functions of python. In Pre-processing phase, we did feature engineering on our dataset. As we converted categorical variables into numerical variables using function of Pandas library. All our datasets contains some categorical variables.

LIBRARIES:

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import plot_confusion_matrix
```

Data Lookup & Cleaning:

Reading the Dataset and assigning that to the variable df:

```
df=pd.read_csv('tested.csv')
```

Access the first 5 rows of a dataframe:

```
df.head()
```

```
In [3]: 1 df.head()
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

Access the last 5 rows of a dataframe:

df.tail()

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

Prints information about the DataFrame:

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Survived     418 non-null    int64
 2   Pclass       418 non-null    int64
 3   Name         418 non-null    object
 4   Sex          418 non-null    object
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64
 7   Parch        418 non-null    int64
 8   Ticket       418 non-null    object
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object
 11  Embarked     418 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

Remove the unwanted columns:

df.drop(["PassengerId","Cabin","Ticket","Name"], inplace=True, axis=1)

Check the number of missing values in each column:

df.isna().sum()

```
Survived     0
Pclass       0
Sex          0
Age         86
SibSp        0
Parch        0
Fare         1
Embarked     0
dtype: int64
```

Identify the numeric and categorical columns:

numeric_cols = df.select_dtypes(include=np.number).columns.tolist()

categorical_cols = df.select_dtypes('object').columns.tolist()

from sklearn.impute import SimpleImputer

imputer=SimpleImputer(strategy='mean').fit(df[numeric_cols])

df[numeric_cols]=imputer.transform(df[numeric_cols])

Check the number of missing values in each column:

```
In [13]:  1 df.isna().sum()

Out[13]: Survived    0
         Pclass      0
         Sex         0
         Age         0
         SibSp       0
         Parch       0
         Fare        0
         Embarked    0
         dtype: int64
```

Check the datatype of each column:

```
In [14]: 1 df.dtypes

Out[14]: Survived    float64
         Pclass      float64
         Sex          object
         Age         float64
         SibSp       float64
         Parch       float64
         Fare        float64
         Embarked     object
         dtype: object
```

Replacing the value M to 0 and F to 1 in the Sex column:

df.Sex=df.Sex.replace({"male":0.0,"female":1.0})

Replacing the value S to 0 , C to 1 and Q to 2 in the Embarked column:

df.Embarked=df.Embarked.replace({"S":0.0,"C":1.0,"Q":2.0})

```
In [17]: 1 df.head()
```

Out[17]:

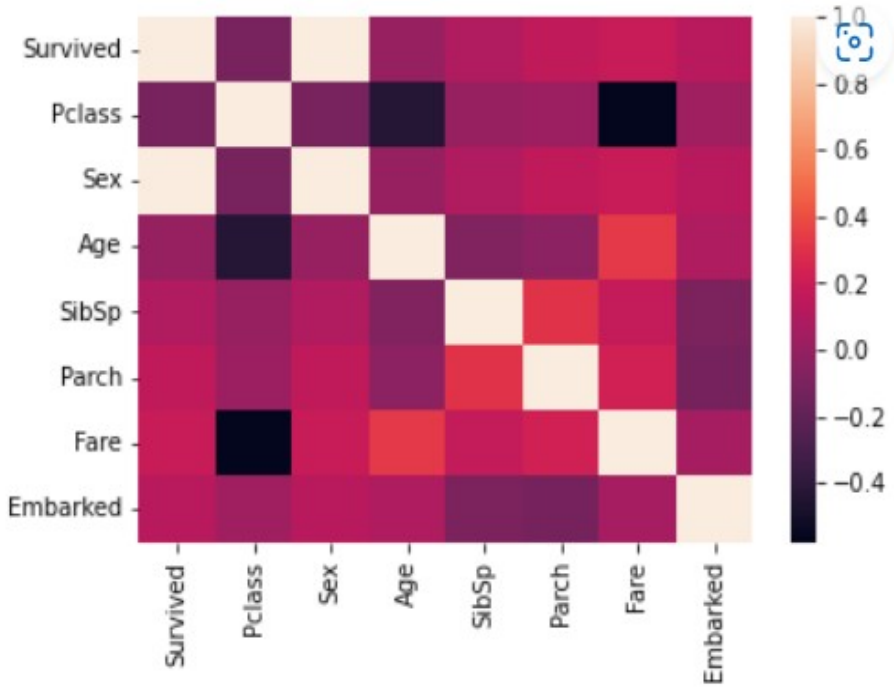
	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0.0	3.0	0.0	34.5	0.0	0.0	7.8292	2.0
1	1.0	3.0	1.0	47.0	1.0	0.0	7.0000	0.0
2	0.0	2.0	0.0	62.0	0.0	0.0	9.6875	2.0
3	0.0	3.0	0.0	27.0	0.0	0.0	8.6625	0.0
4	1.0	3.0	1.0	22.0	1.0	1.0	12.2875	0.0

Check the datatype of each column:

```
In [18]:  1  df.dtypes

Out[18]: Survived    float64
         Pclass      float64
         Sex         float64
         Age         float64
         SibSp       float64
         Parch       float64
         Fare        float64
         Embarked    float64
         dtype: object
```

Correlation:



Creation of a Model:

Take the X and y value:

In [22]:
```
X=df.drop('Survived',axis=1)
X
```

Out[22]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3.0	0.0	34.50000	0.0	0.0	7.8292	2.0
1	3.0	1.0	47.00000	1.0	0.0	7.0000	0.0
2	2.0	0.0	62.00000	0.0	0.0	9.6875	2.0
3	3.0	0.0	27.00000	0.0	0.0	8.6625	0.0
4	3.0	1.0	22.00000	1.0	1.0	12.2875	0.0
...	...	...	...	...	...	...	...
413	3.0	0.0	30.27259	0.0	0.0	8.0500	0.0
414	1.0	1.0	39.00000	0.0	0.0	108.9000	1.0
415	3.0	0.0	38.50000	0.0	0.0	7.2500	0.0
416	3.0	0.0	30.27259	0.0	0.0	8.0500	0.0
417	3.0	0.0	30.27259	1.0	1.0	22.3583	1.0

418 rows × 7 columns

In [23]:
```
y=df.Survived
y
```

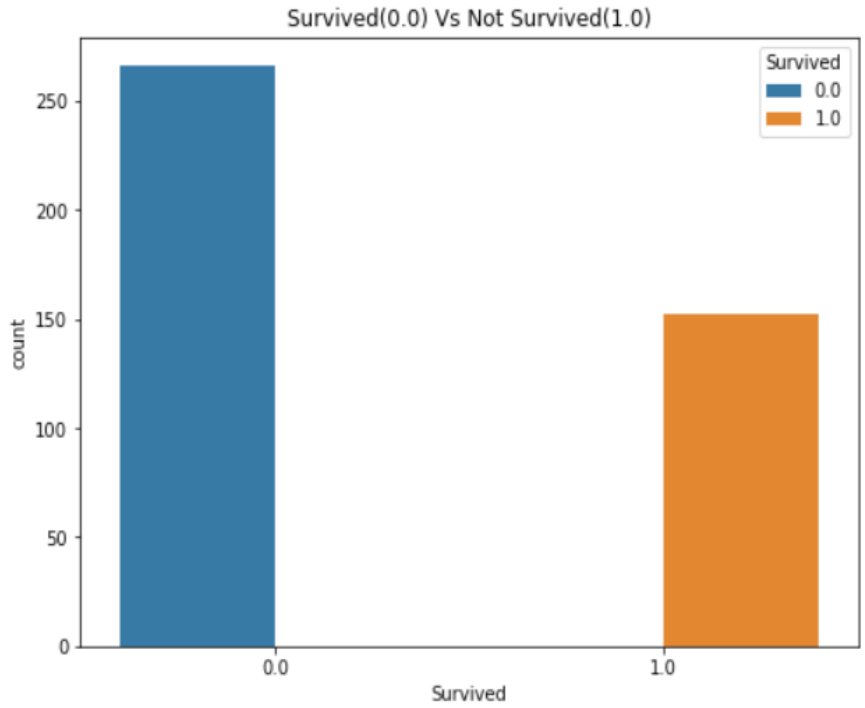
Out[23]:
```
0      0.0
1      1.0
2      0.0
3      0.0
4      1.0
      ...
413    0.0
414    1.0
415    0.0
416    0.0
417    0.0
Name: Survived, Length: 418, dtype: float64
```

In [24]:
```
y.value_counts()
```

Out[24]:
```
0.0    266
1.0    152
Name: Survived, dtype: int64
```

In [26]:
```
plt.figure(figsize=(8,6))
sns.countplot('Survived', data=df, hue = 'Survived')

plt.title("Survived(0.0) Vs Not Survived(1.0)")
plt.show()
```



Split the Training Dataset and Test Dataset:

In [28]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)
```

In [29]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [30]:
```
model=RandomForestClassifier()
```

Fitting the training data to the model:

In [31]:
```
model.fit(X,y)
```

Out[31]:
```
RandomForestClassifier()
```

Prediction:

```
In [32]: 1 y_predict=model.predict(X_test)

In [33]: 1 y_predict

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

Accuracy:

```
In [35]: 1 accuracy_score(y_test,y_predict)*100

Out[35]: 100.0
```

- Overall Accuracy = **100%**

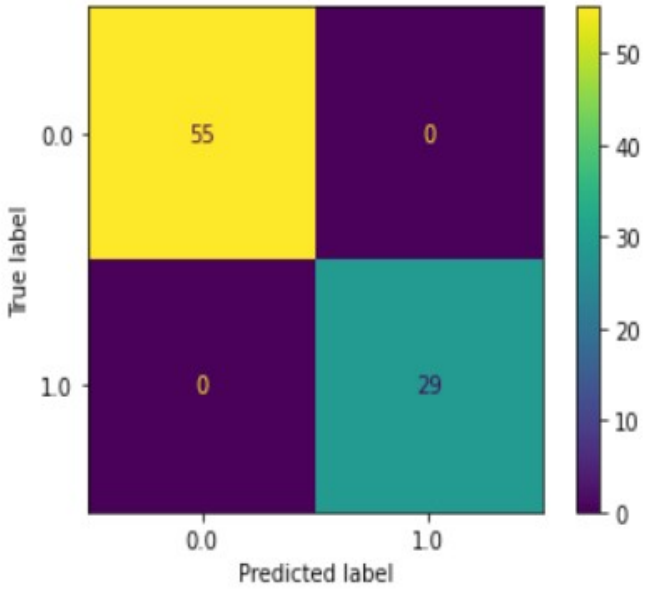
Confusion Matrix:

```
In [37]: 1 performance=confusion_matrix(y_test,y_predict)

In [38]: 1 performance

Out[38]: array([[55,  0],
                [ 0, 29]], dtype=int64)

In [40]: 1 plot_confusion_matrix(model,X_test,y_test)
         2 plt.show()
```


55
0
0
29
0.0
1.0
True label
0.0
1.0
Predicted label
50
40
30
20
10
0

Chapter 4

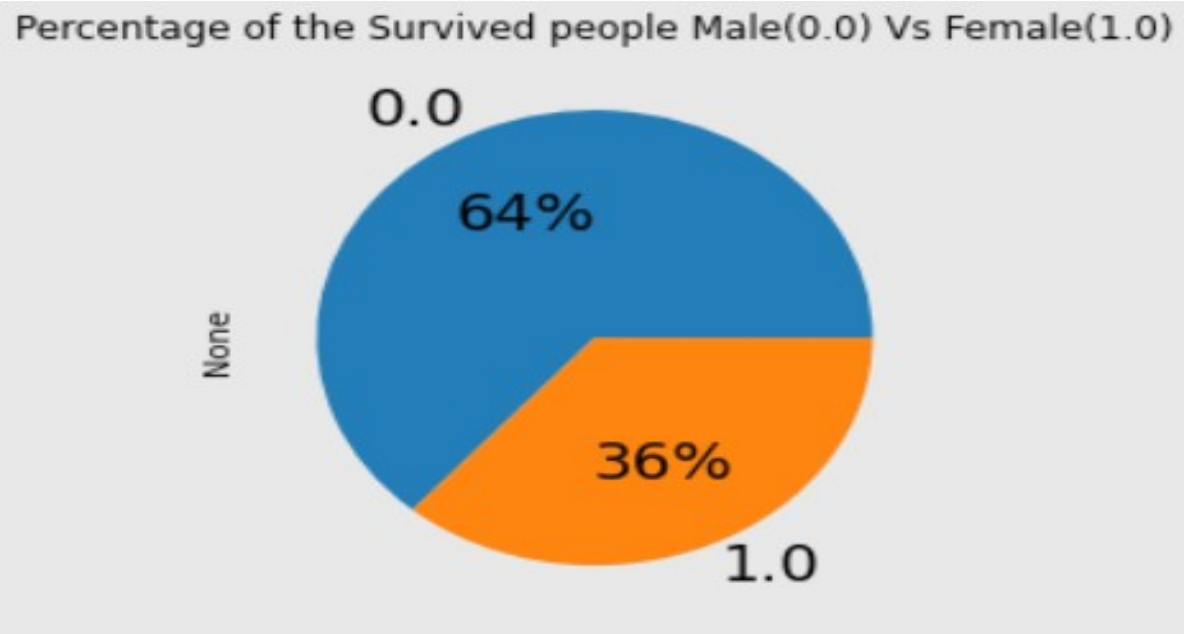
Analysis of the Result

Percentage of the Survived people Male(0.0) Vs Female(1.0):

```
df.groupby('Sex').size().plot(kind='pie',textprops={'fontsize': 20},autopct='%1.0f%%')

plt.title('Percentage of the Survived people Male(0.0) Vs Female(1.0)')

plt.show()
```



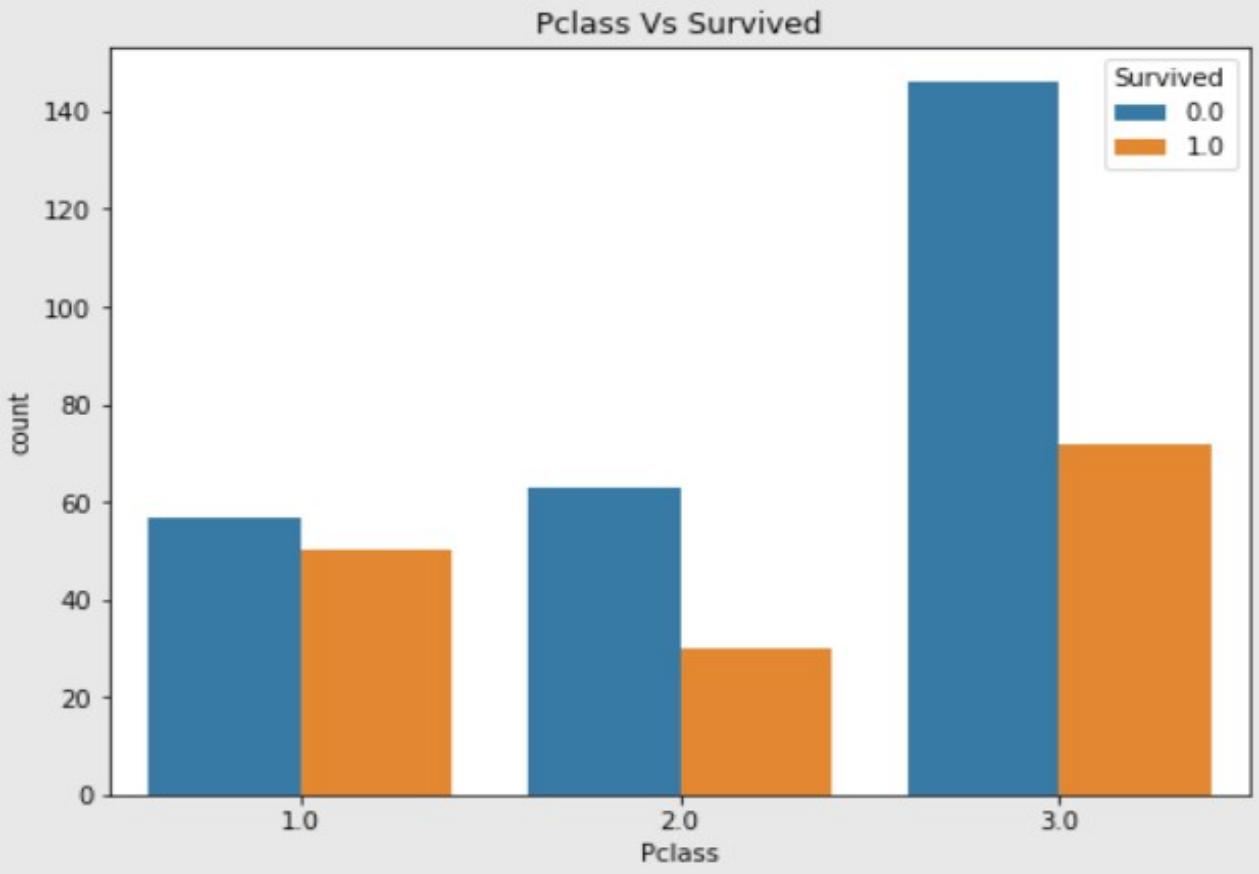
Observation:

- 64% of the Male's were Survived in the Titanic

- 36% of the Female's were Survived in the Titanic

Pclass Vs Survived:

```
plt.figure(figsize=(8,6))
sns.countplot('Pclass', data=df, hue = 'Survived')
plt.title("Pclass Vs Survived")
plt.show()
```

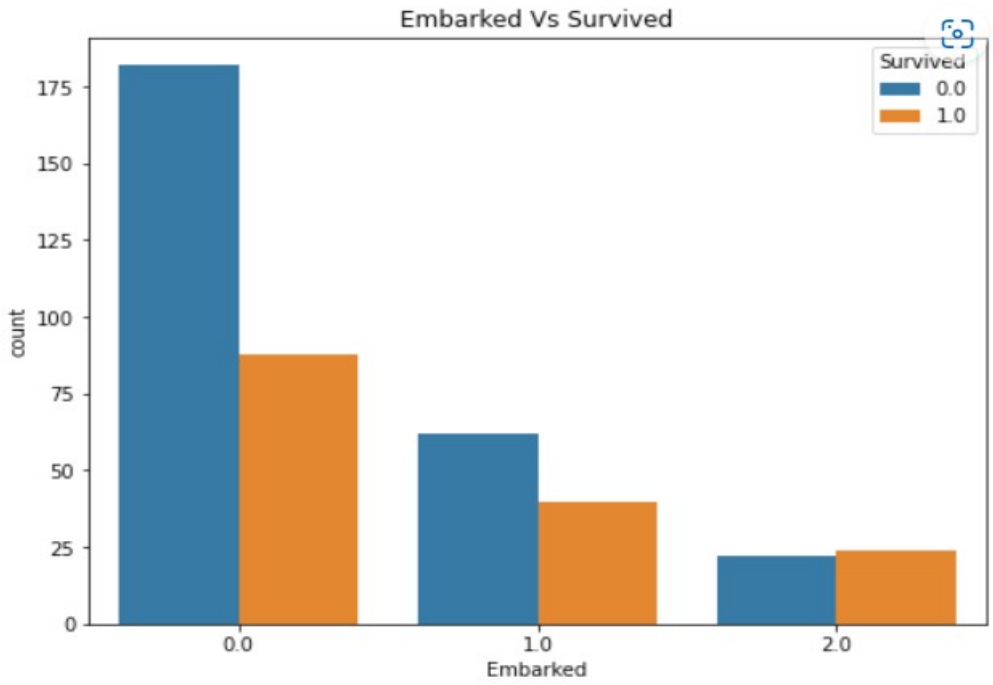


Observation:

- Total No. of Survived Passengers count are higher in 3.0 (Pclass)

Embarked Vs Survived:

```
plt.figure(figsize=(8,6))
sns.countplot('Embarked', data=df, hue = 'Survived')
plt.title("Embarked Vs Survived")
plt.show()
```



Observation:

- Total No. of Survived Passengers count are higher in 0.0 (Embarked) - S (Southampton)

Accuracy:

- Overall Accuracy of the model – 100%

Chapter 5

Conclusion

- Overall Accuracy of the model -100%
- Here we conclude that the 64% of the Males were Survived in the Titanic and 36% of the Females were Survived in the Titanic.
- Total No. of Survived Passengers count are higher in 3.0 (3rd class)
- Total No. of Survived Passengers count are higher in 0.0 (Embarked) - S (Southampton)

Here is access to the data set & code from my GitHub page:

https://github.com/Adaikkkappan/Machine-Learning-Titanic-Dataset

References

1. Titanic dataset from Kaggle Created by **BRENDA N.**

2. A Titanic Probability from the website:

https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/problem12.html

4. Titanic from openml website.

Author: Frank E. Harrell Jr., Thomas Cason

URL: https://www.openml.org/search?type=data&sort=runs&id=40945&status=active