EDA of Titanic Dataset

Import Libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

The Data

In [2]:

```
train = pd.read_csv('titanic_train.csv')
```

In [3]:

```
train.head()
```

Out[3]:

```
   PassengerId  Survived  Pclass                                     Name     Sex   Age  SibSp  Parch            Ticket     Fare  Cabin  Embarked
0            1         0       3                  Braund, Mr. Owen Harris    male  22.0      1      0         A/5 21171   7.2500    NaN         S
                                         Cumings, Mrs. John Bradley (Florence
1            2         1       1                              Briggs Th...  female  38.0      1      0          PC 17599  71.2833    C85         C
                                                                                                                STON/O2.
2            3         1       3                    Heikkinen, Miss. Laina  female  26.0      0      0           3101282   7.9250    NaN         S
                                         Futrelle, Mrs. Jacques Heath (Lily May
3            4         1       1                                     Peel)  female  35.0      1      0            113803  53.1000   C123         S
4            5         0       3                 Allen, Mr. William Henry    male  35.0      0      0            373450   8.0500    NaN         S
```

Exploratory Data Analysis

Missing Data

Missing Data

In [4]:

```
train.isnull()
```

Out[4]:

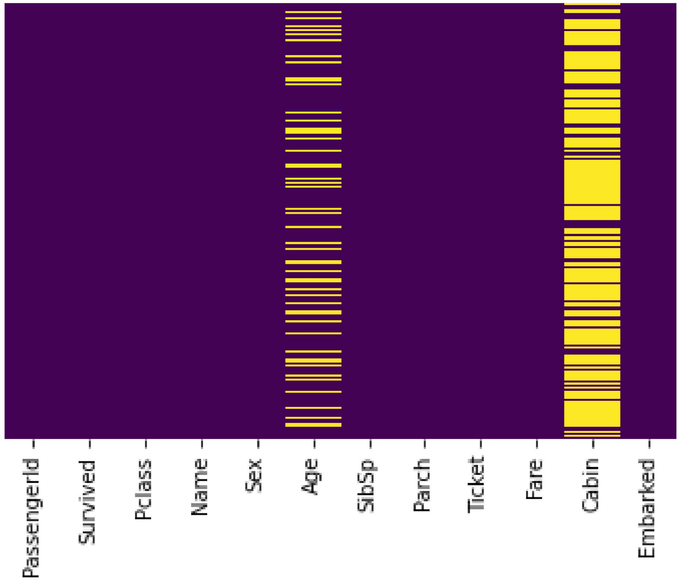
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	False	False	False	False	False	False	False	False	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True	False
887	False	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True	False
889	False	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True	False

891 rows × 12 columns

Heat Map

In [5]:
```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[5]:
```
<AxesSubplot:>
```

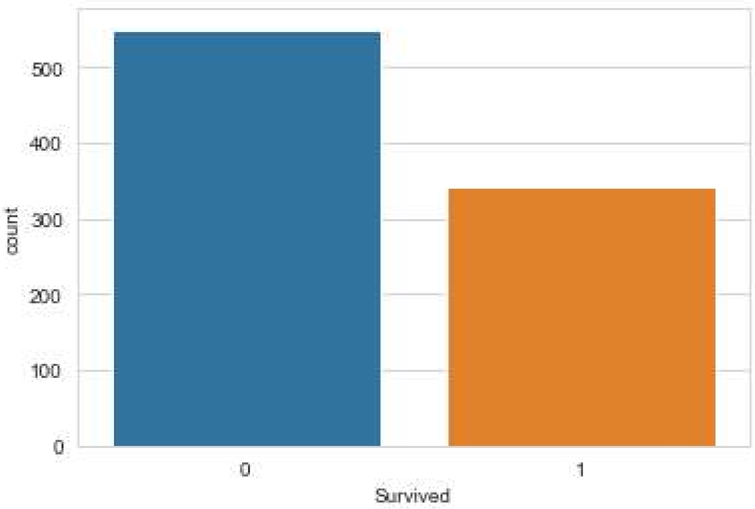


Roughly 20% of the age data is missing. The proportion of age missing is likely small enough for reasonable replacement with some form of imputation looking at the cabin column, it looks like we are just missing too much of that data to do something usweful with at a baic level. We will probably drop this later, or change it to another feature like "Cabin known: 1 or 0"

Classification

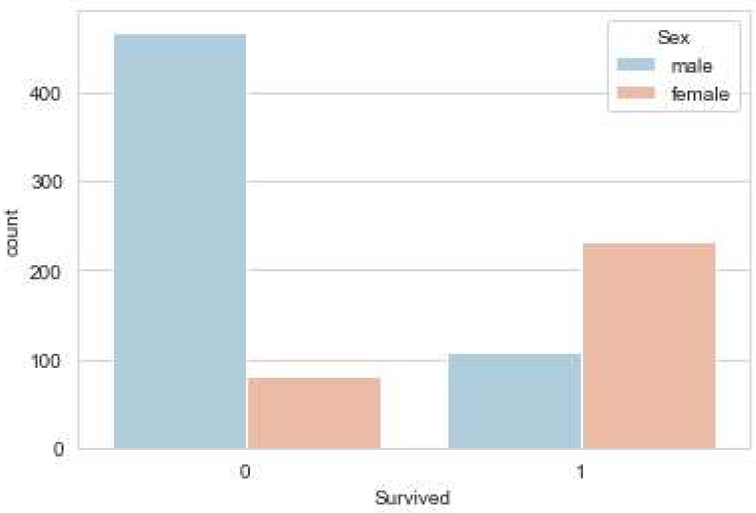
```
In [6]: sns.set_style("whitegrid")
        sns.countplot(x="Survived",data=train)
```

```
Out[6]: <AxesSubplot:xlabel='Survived', ylabel='count'>
```



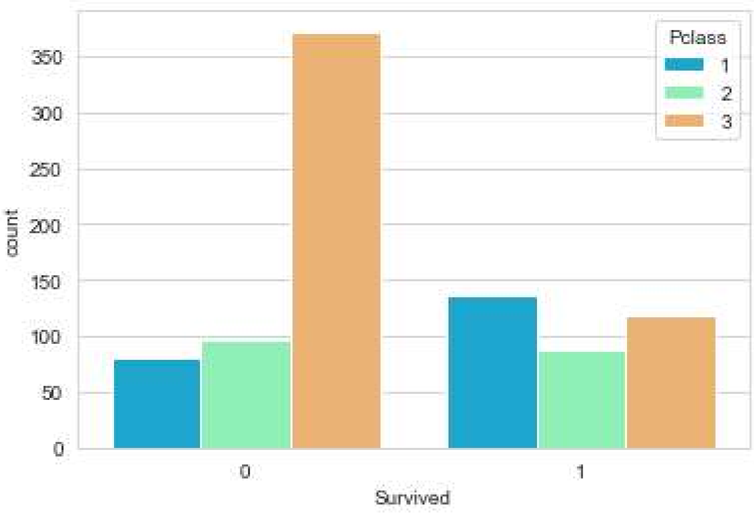
In [7]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',data=train,hue="Sex",palette='RdBu_r')
```

Out[7]:
```
<AxesSubplot:xlabel='Survived', ylabel='count'>
```



```
In [8]: sns.set_style('whitegrid')
        sns.countplot(x='Survived',hue='Pclass',data=train,palette='rainbow')
```

```
Out[8]: <AxesSubplot:xlabel='Survived', ylabel='count'>
```



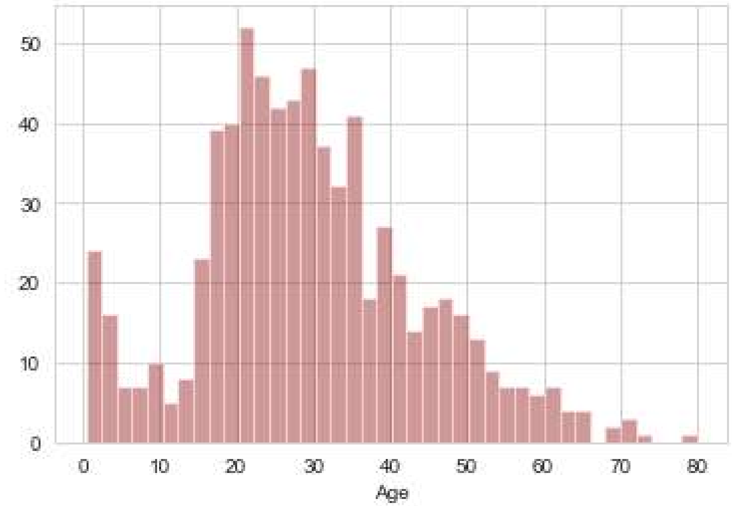
In [9]:

```
sns.distplot(train['Age'].dropna(),kde=False,color='darkred',bins=40)
```

```
C:\Users\MOHD. RAEES\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecat
ed function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

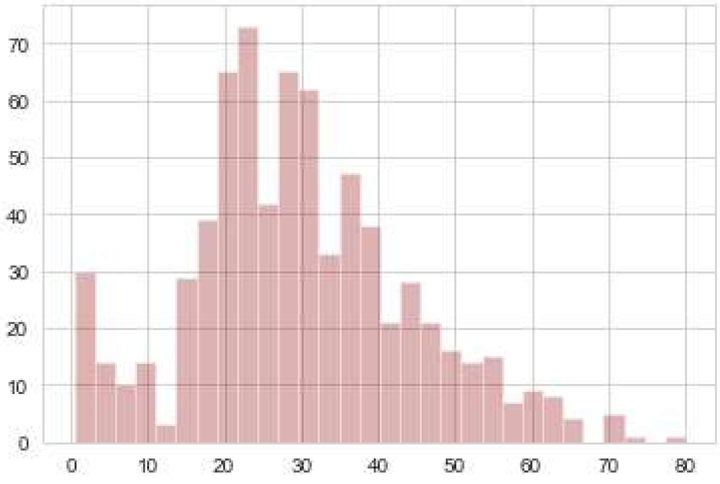
Out[9]:

```
<AxesSubplot:xlabel='Age'>
```



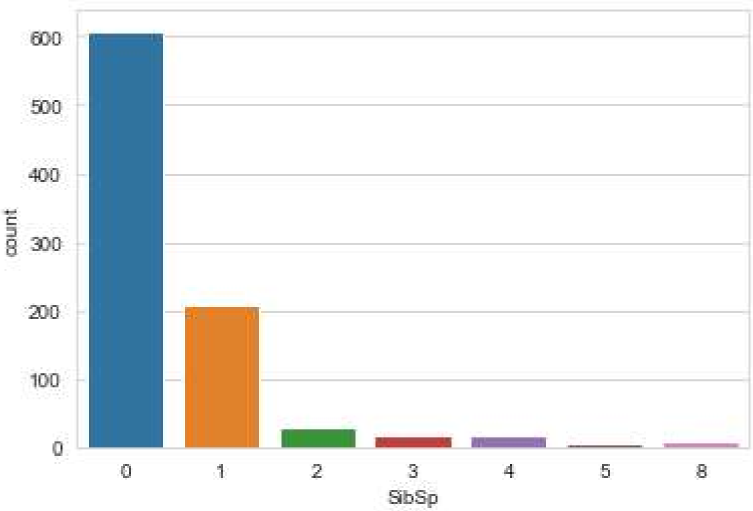
```
In [10]: train['Age'].hist(bins=30,color='darkred',alpha=0.3)
```

```
Out[10]: <AxesSubplot:>
```



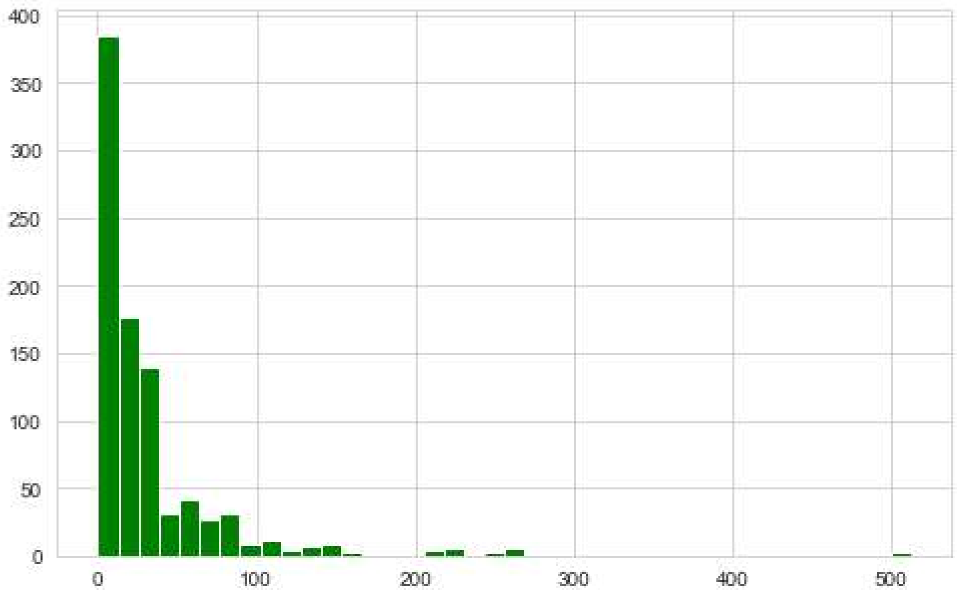
```
In [11]: sns.countplot(x='SibSp',data=train)
```

```
Out[11]: <AxesSubplot:xlabel='SibSp', ylabel='count'>
```



In [12]:
```
train['Fare'].hist(color='green',bins=40,figsize=(8,5))
```

Out[12]:
```
<AxesSubplot:>
```



Cufflinks for plots

In [18]:
```
import cufflinks as cf
cf.go_offline()
```

In [17]:

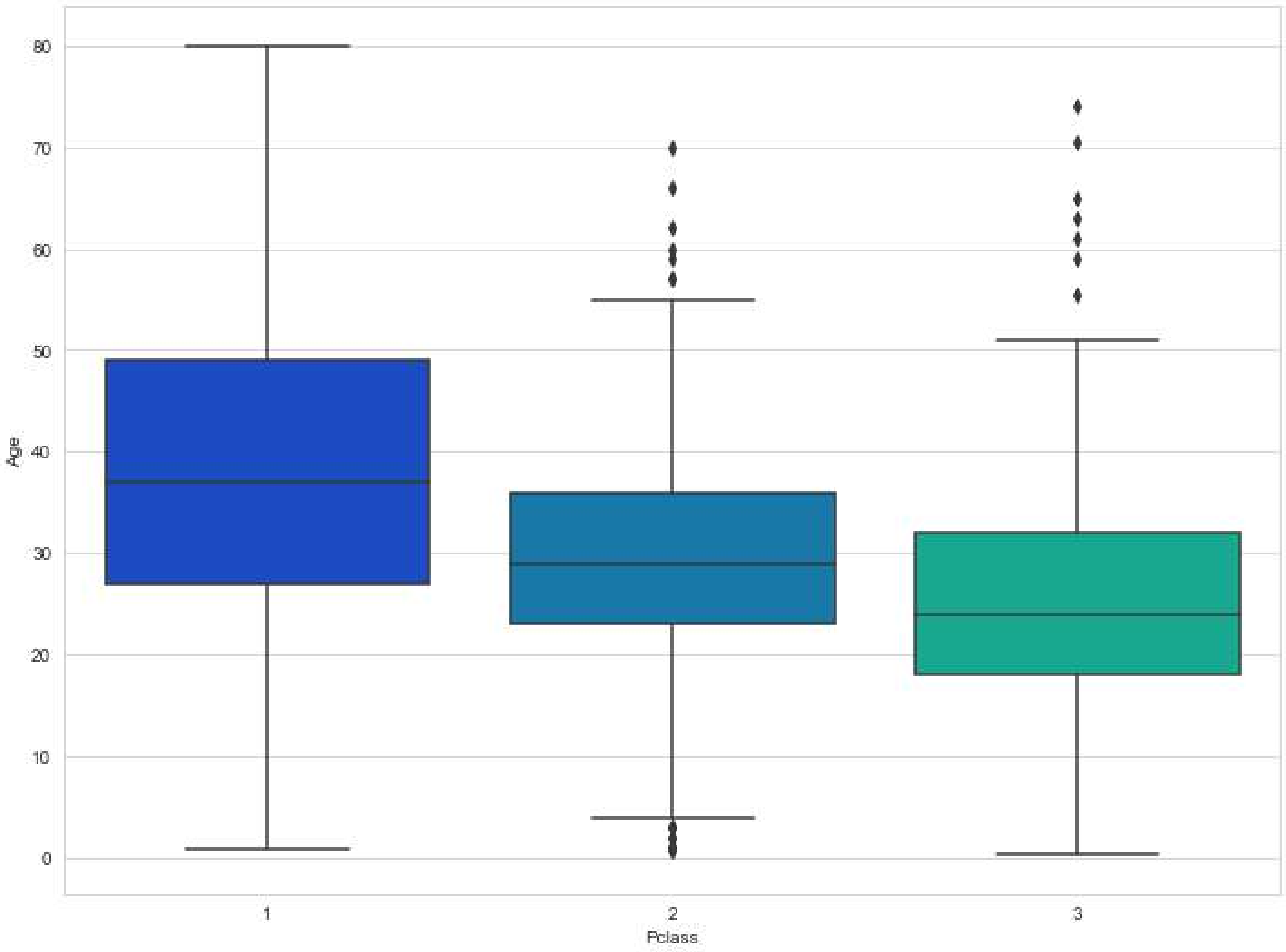
```
train['Fare'].iplot(kind='hist',bins=30,color='green')
```

Data cleaning

We want to fill the missing data instead of just dropping the missing age data rows. One way to do this is by filling in the mean age of all the passengers. however we can check the average age by passenger class.

```
In [15]: plt.figure(figsize=(12,9))
         sns.boxplot(x='Pclass',y='Age',data=train,palette='winter')
```

```
Out[15]: <AxesSubplot:xlabel='Pclass', ylabel='Age'>
```



We can see that whether passengers in the higher classes tend to be older, which makes sense. We will use these average age values to impute based on Pclass for Age

In [19]:

```
def input_age(cols):
    Age = cols[0]
    Pclass = cols[1]

    if pd.isnull(Age):
        if Pclass == 1:
            return 37
        elif Pclass == 2:
            return 29
        else:
            return 24

    else:
        return Age
```

Applying the above function

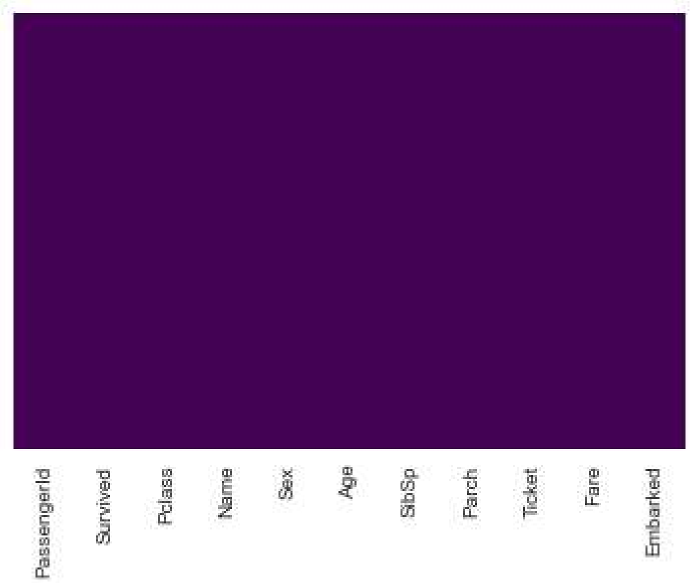
In [20]:

```
train['Age'] = train[['Age','Pclass']].apply(input_age,axis=1)
```

Check the heatmap again

In [26]:
```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[26]:
```
<AxesSubplot:>
```



Drop the Cabin column and the row in Embarked that is NaN

In [23]:
```
train.drop('Cabin',axis=1,inplace=True)
```

In [24]:
```
train.head()
```

Out[24]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S

In [25]:
```
train.dropna(inplace=True)
```

Converting Categorical Features

We will need to convert the categorical features to dummy variables using pandas! Otherwise our machine learning algorithm would not be able to directly take in those features as inputs

In [27]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  889 non-null    int64  
 1   Survived     889 non-null    int64  
 2   Pclass       889 non-null    int64  
 3   Name         889 non-null    object 
 4   Sex          889 non-null    object 
 5   Age          889 non-null    float64
 6   SibSp        889 non-null    int64  
 7   Parch        889 non-null    int64  
 8   Ticket       889 non-null    object 
 9   Fare         889 non-null    float64
 10  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(4)
memory usage: 83.3+ KB
```

In [28]:
```
pd.get_dummies(train['Embarked'],drop_first=True).head()
```

Out[28]:

	Q	S
0	0	1
1	0	0
2	0	1
3	0	1
4	0	1

In [29]:
```
sex = pd.get_dummies(train['Sex'],drop_first=True)
embark = pd.get_dummies(train['Embarked'],drop_first=True)
```

In [30]:
```
train.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)
```

In [31]:
```
train.head()
```

Out[31]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
0	1	0	3	22.0	1	0	7.2500
1	2	1	1	38.0	1	0	71.2833
2	3	1	3	26.0	0	0	7.9250
3	4	1	1	35.0	1	0	53.1000
4	5	0	3	35.0	0	0	8.0500

In [32]:
```
train = pd.concat([train,sex,embark],axis=1)
```

In [33]:
```
train.head()
```

Out[33]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	1	0	3	22.0	1	0	7.2500	1	0	1
1	2	1	1	38.0	1	0	71.2833	0	0	0
2	3	1	3	26.0	0	0	7.9250	0	0	1
3	4	1	1	35.0	1	0	53.1000	0	0	1
4	5	0	3	35.0	0	0	8.0500	1	0	1

Our data is ready for our model

Building a Logistic Regression model

Splitting the data into training set and test set

Train Test Split

In [34]:
```
train.drop('Survived',axis=1).head()
```

Out[34]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	1	3	22.0	1	0	7.2500	1	0	1
1	2	1	38.0	1	0	71.2833	0	0	0
2	3	3	26.0	0	0	7.9250	0	0	1
3	4	1	35.0	1	0	53.1000	0	0	1
4	5	3	35.0	0	0	8.0500	1	0	1

In [35]:
```
train['Survived'].head()
```

Out[35]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [36]:
```
from sklearn.model_selection import train_test_split
```

In [37]:
```
x_train,x_test,y_train,y_test = train_test_split(train.drop('Survived',axis=1),
                                                 train['Survived'],test_size=0.30,
                                                 random_state=101)
```

Accuracy, Training and Predicting

In [61]:
```
from sklearn.linear_model import LogisticRegression
```

In [62]:

```
logmodel = LogisticRegression()
logmodel.fit(x_train,y_train)
```

```
C:\Users\MOHD. RAEES\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:444: ConvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.h
tml)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modu
les/linear_model.html#logistic-regression)
```

Out[62]:

```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [63]:

```
predictions = logmodel.predict(x_test)
```

In [64]:

```
from sklearn.metrics import confusion_matrix
```

In [65]:

```
accuracy = confusion_matrix(y_test,predictions)
```

In [66]:

```
accuracy
```

Out[66]:

```
array([[149,  14],
       [ 39,  65]], dtype=int64)
```

In [67]:

```
from sklearn.metrics import accuracy_score
```

In [68]:
```
accuracy = accuracy_score(y_test,predictions)
accuracy
```

Out[68]:
```
0.8014981273408239
```

In [70]:
```
predictions
```

Out[70]:
```
array([0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
       1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
       0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0,
       0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 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, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0,
       0, 1, 1], dtype=int64)
```

In [ ]:
```

```

Analyzed by

Md Raiesh, Enrollment number : **19UME116**, Registration number : **1911345**, B Tech,**7th** semester,Section : **A**, Mechanical Engineering Department, National Institute of Technology Agartala, Tripura 799046,