In [2]:

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

In [3]:

```python
train = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
```

In [4]:

```python
train.head()
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0

In [5]:

```python
train.isnull()
```

Out[5]:

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

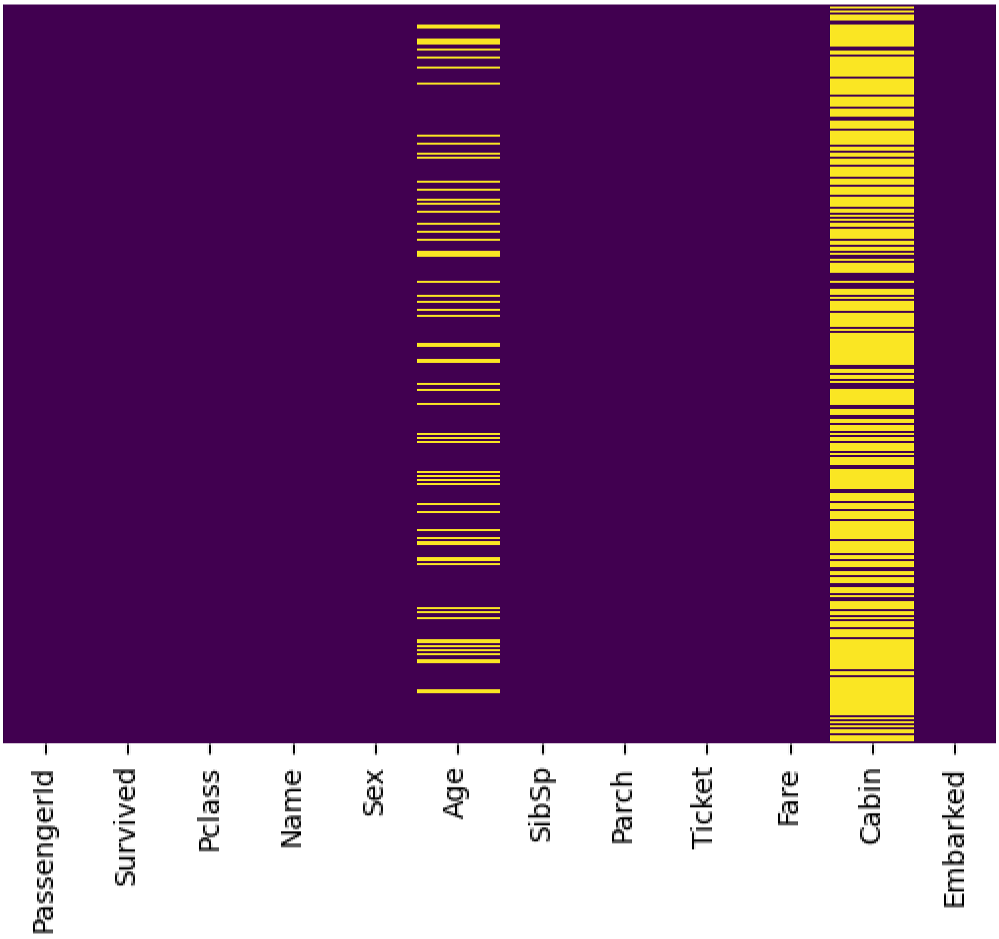
891 rows × 12 columns

In [6]:

```python
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

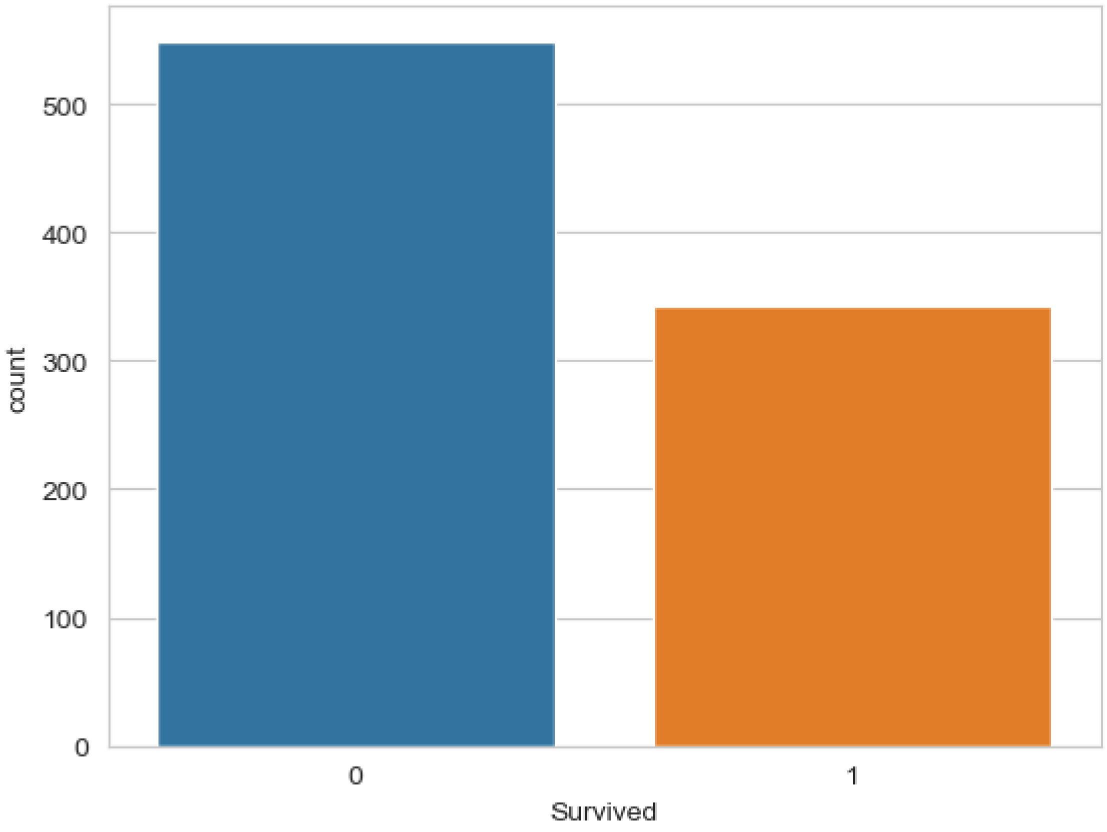
Out[6]:

```
<Axes: >
```



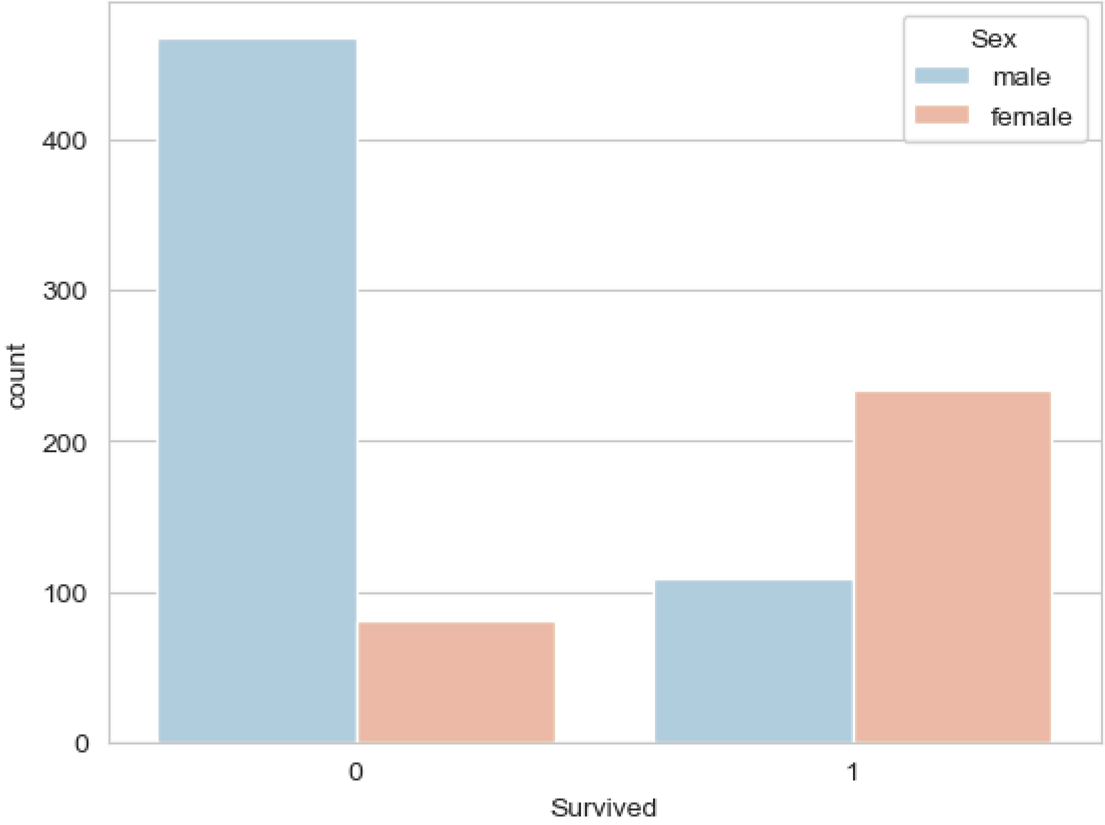
In [7]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',data=train)
```

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

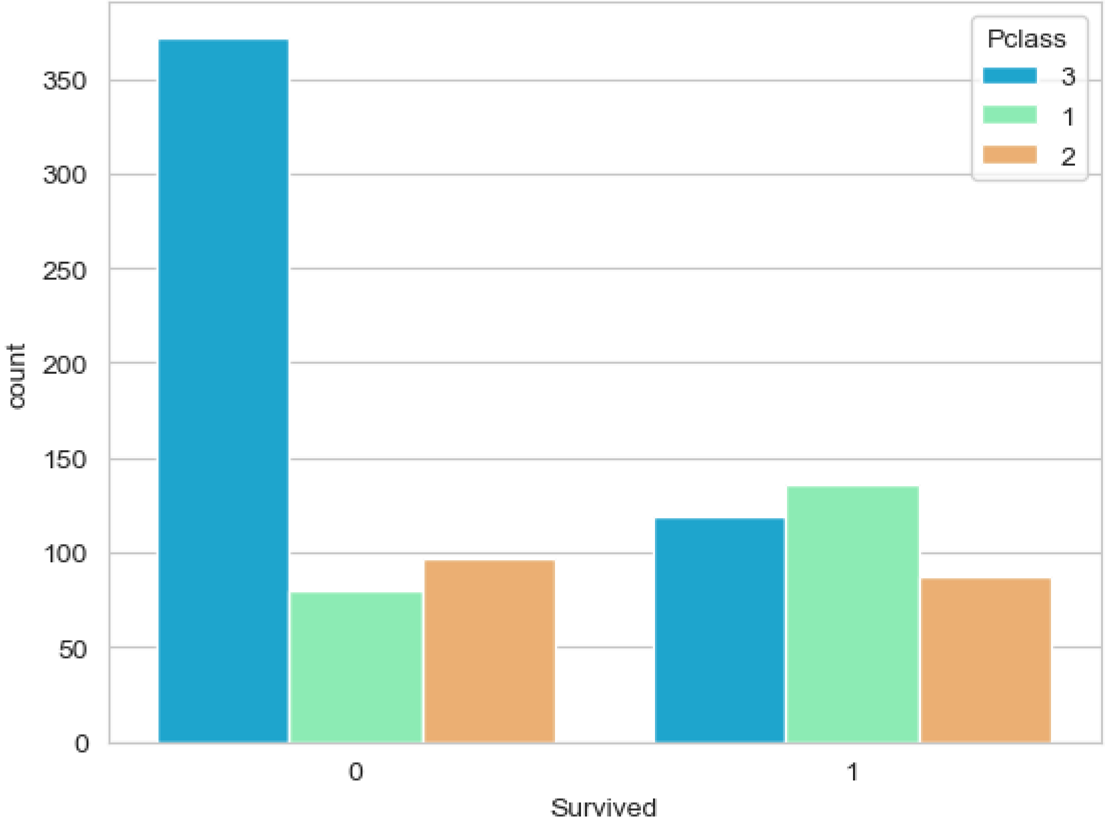


In [8]:
```
sns.set_style('whitegrid')
sns.countplot(x='Survived',hue='Sex',data=train,palette='RdBu_r')
```

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



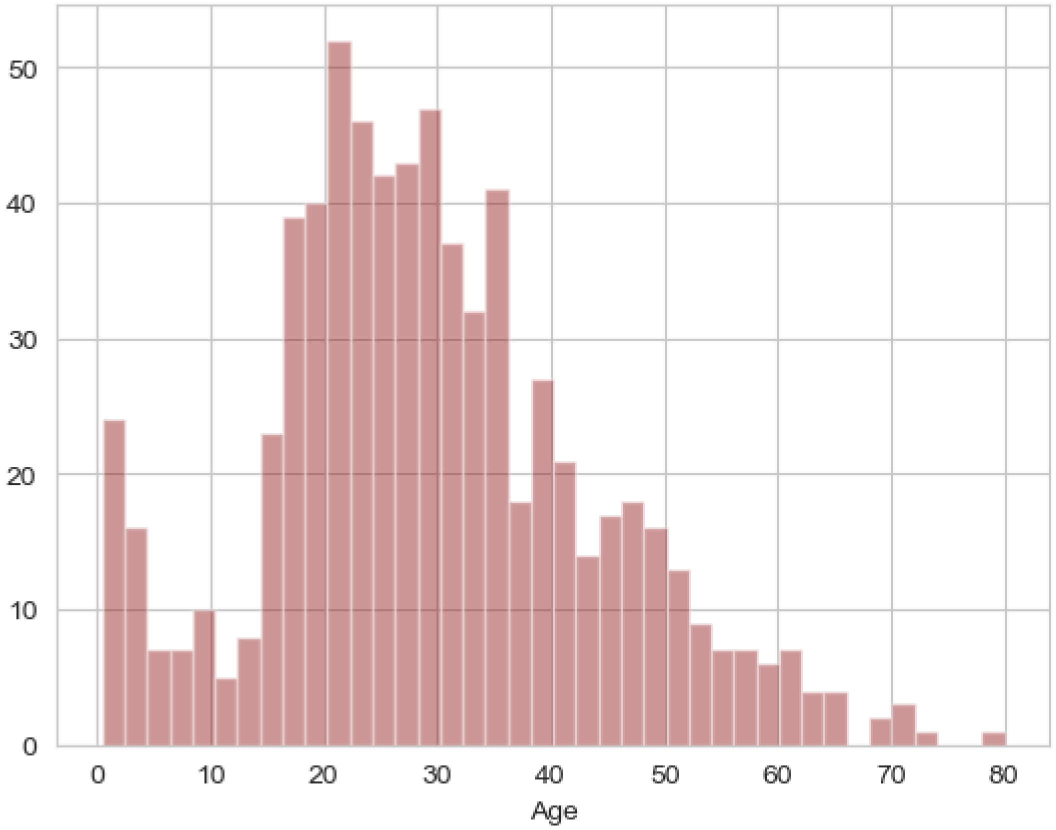
In [9]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Ensure the 'Pclass' column is of type string
train['Pclass'] = train['Pclass'].astype(str)

sns.set_style('whitegrid')
sns.countplot(x='Survived', hue='Pclass', data=train, palette='rainbow')
plt.show()
```



In [19]:
```
sns.distplot(train['Age'].dropna(),kde=False,color='darkred',bins=40)
```

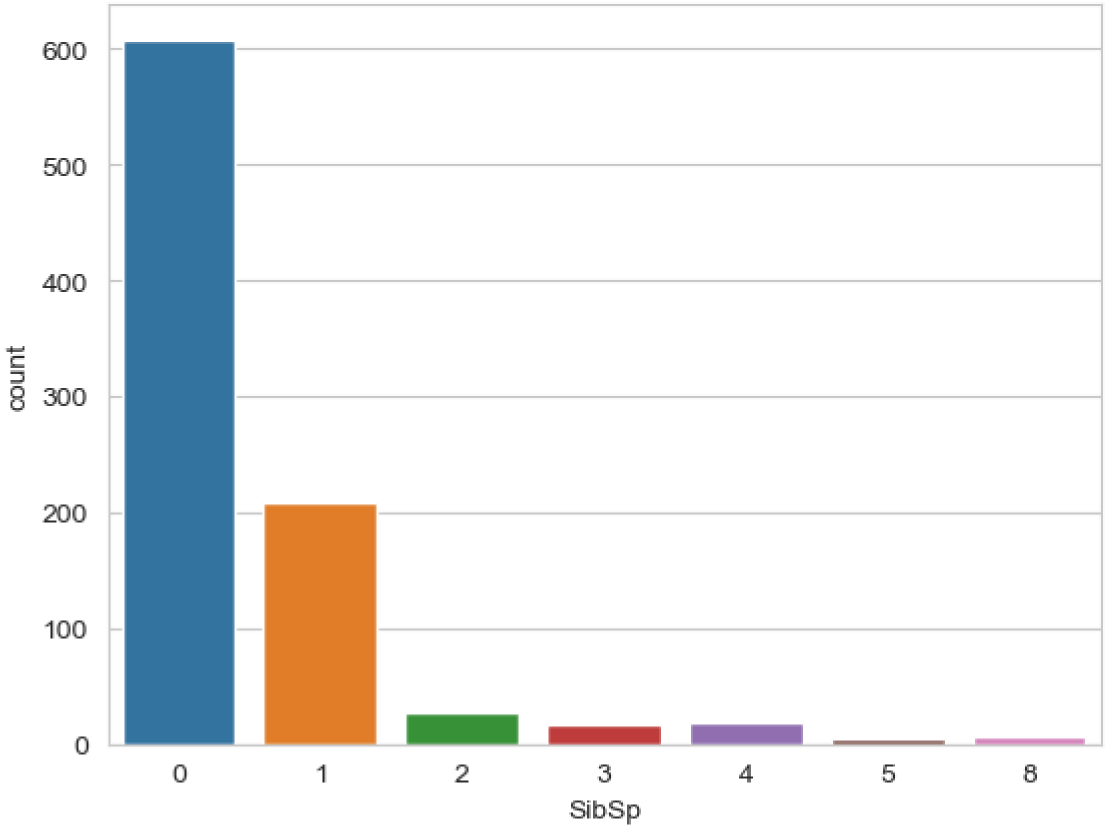
```
C:\Users\user\AppData\Local\Temp\ipykernel_23296\2002818437.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

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

Out[19]:
```
<Axes: xlabel='Age'>
```



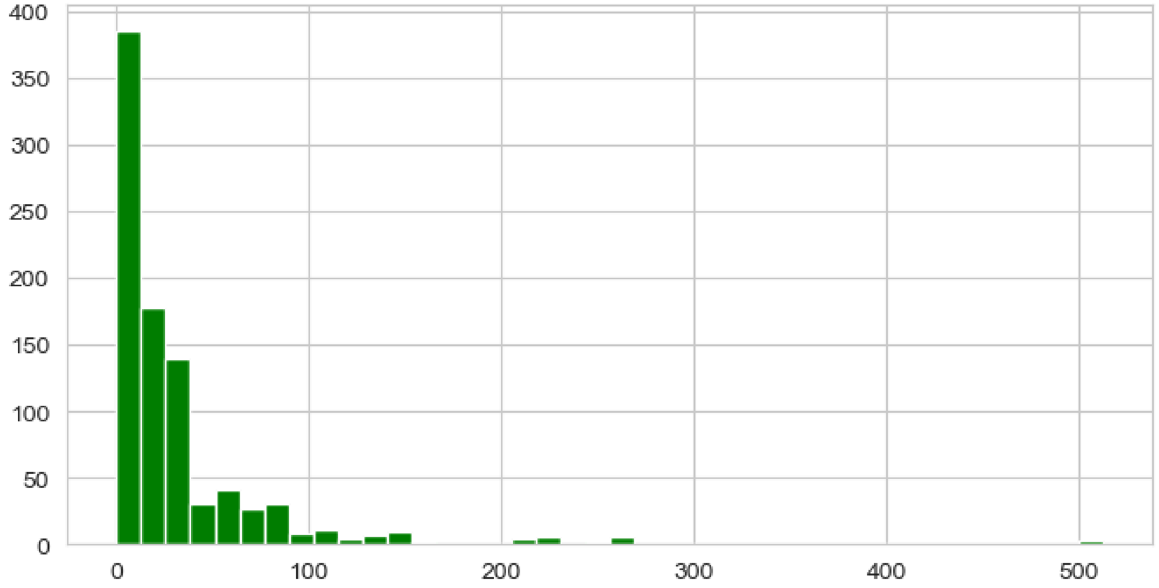
```
In [20]: sns.countplot(x='SibSp',data=train)
```

```
Out[20]: <Axes: xlabel='SibSp', ylabel='count'>
```



```
In [21]: train['Fare'].hist(color='green',bins=40,figsize=(8,4))
```

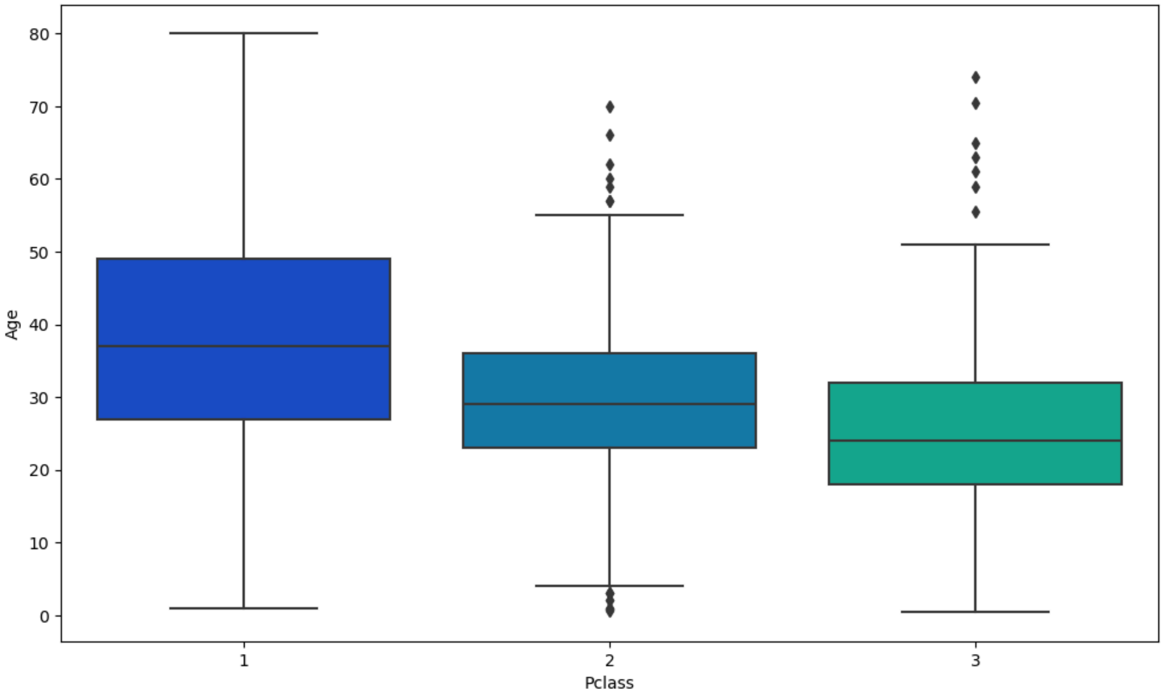
```
Out[21]: <Axes: >
```



```
In [ ]: #DATA CLEANING
```

```
In [4]: plt.figure(figsize=(12, 7))
        sns.boxplot(x='Pclass',y='Age',data=train,palette='winter')
```

```
Out[4]: <Axes: xlabel='Pclass', ylabel='Age'>
```



```
In [5]: def impute_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
```

In [ ]:

```
#Now apply that function!
```

In [6]:

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

```
C:\Users\user\AppData\Local\Temp\ipykernel_16528\822839471.py:2: FutureWarning: S
eries.__getitem__ treating keys as positions is deprecated. In a future version,
integer keys will always be treated as labels (consistent with DataFrame behavio
r). To access a value by position, use `ser.iloc[pos]`
  Age = cols[0]
C:\Users\user\AppData\Local\Temp\ipykernel_16528\822839471.py:3: FutureWarning: S
eries.__getitem__ treating keys as positions is deprecated. In a future version,
integer keys will always be treated as labels (consistent with DataFrame behavio
r). To access a value by position, use `ser.iloc[pos]`
  Pclass = cols[1]
```

In [ ]:

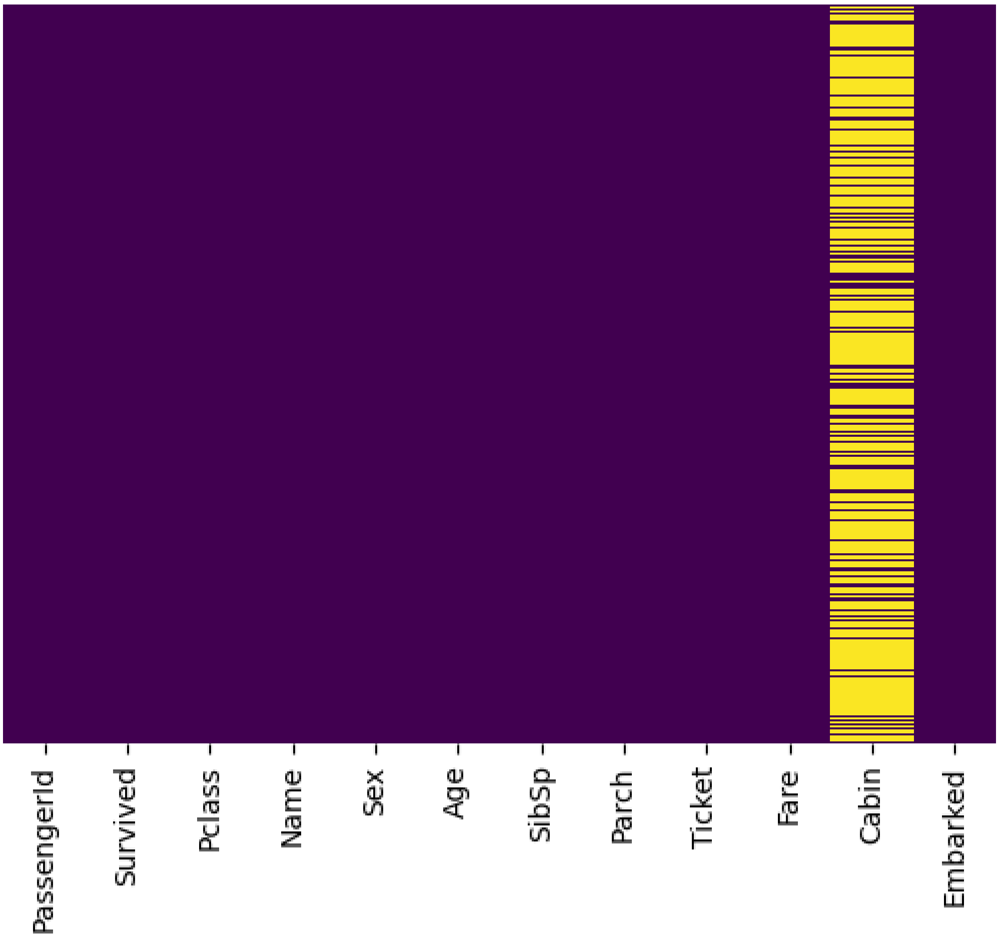
```
#Now let's check that heat map again!
```

In [7]:

```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[7]:

```
<Axes: >
```



In [60]:

```
train.drop('Cabin',axis=1,inplace=True)
train.head()
```

Out[60]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	I
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0

In [61]:

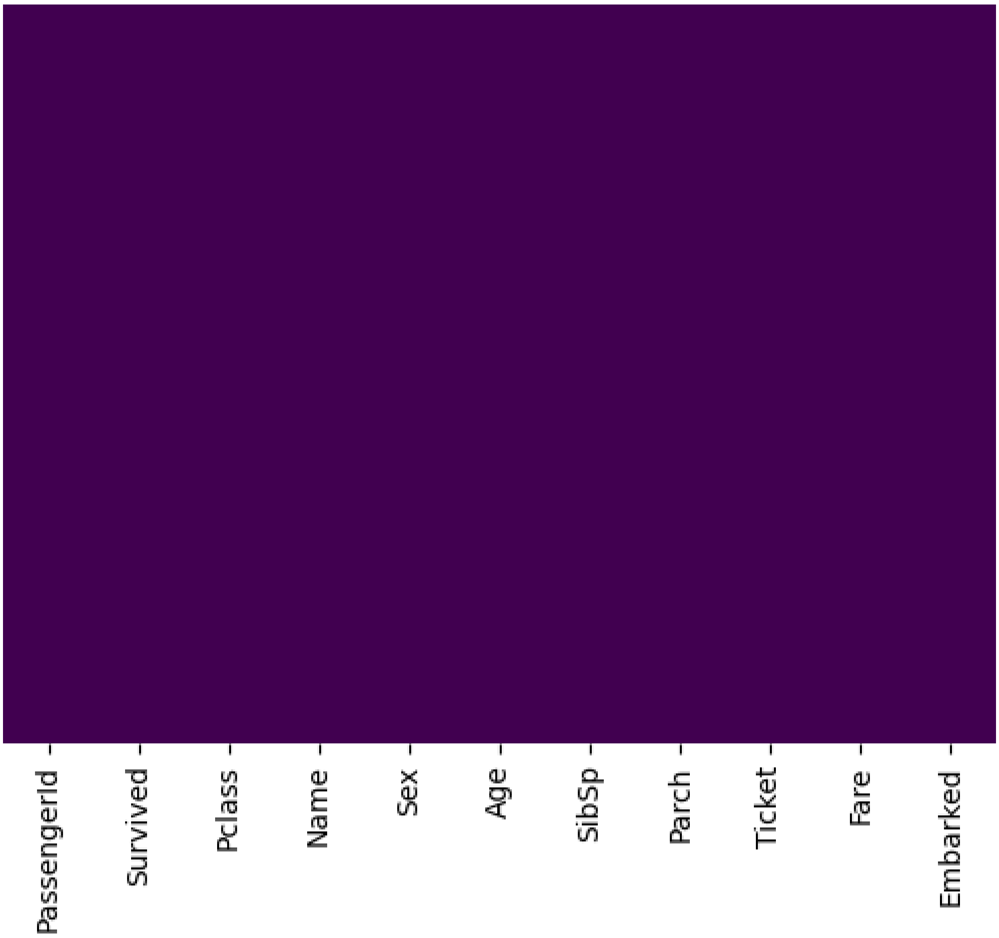
```
train.dropna(inplace=True)
```

In [10]:

```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[10]:

```
<Axes: >
```



In [11]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 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 [ ]:
```
#We'll need to convert categorical features to dummy variables using pandas! Oth
```

In [12]:
```
pd.get_dummies(train['Embarked'],drop_first=True).head()
```

Out[12]:

	Q	S
0	False	True
1	False	False
2	False	True
3	False	True
4	False	True

In [13]:
```
sex = pd.get_dummies(train['Sex'],drop_first=True)
embark = pd.get_dummies(train['Embarked'],drop_first=True)
```

In [14]:
```
train.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)
```

In [15]:
```
train.head()
```

Out[15]:

	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 [16]:
```
train = pd.concat([train,sex,embark],axis=1)
train.head()
```

Out[16]:

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

In [ ]:
```
#LOGISTIC REGRESSION
```

In [ ]:
```
#remove passenger id and name since it is unnecessary
```

In [17]:
```
train.drop(['PassengerId'], axis=1, inplace=True)
train.head()
```

Out[17]:

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

In [18]:

```
# Convert boolean to integers (True -> 1, False -> 0)
train[['male', 'Q', 'S']] = train[['male', 'Q', 'S']].astype(int)
```

In [40]:

```
#Define Features and Target Variable
#TRAIN TEST SPLIT
```

In [19]:

```
train.drop('Survived',axis=1).head()
```

Out[19]:

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

In [20]:

```
train['Survived'].head()
```

Out[20]:

```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [21]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(train.drop('Survived',axis=1
                                                    train['Survived'], test_size
                                                    random_state=101)
```

In [ ]:

```
#TRAINING AND PREDICTING
```

In [45]:

```
from sklearn.linear_model import LogisticRegression
```

In [46]:

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

```
D:\anaconda\Lib\site-packages\sklearn\linear_model\_logistic.py:458: ConvergenceW
arning: 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
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[46]:

```
▾ LogisticRegression

LogisticRegression()
```

In [47]:

```python
predictions = logmodel.predict(X_test)
```

In [ ]:

```python
#EVALUATION
```

In [48]:

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

In [49]:

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

Out[49]:

```
array([[147,  16],
       [ 30,  74]], dtype=int64)
```

In [50]:

```python
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(y_test,predictions)
accuracy
```

Out[50]:

```
0.8277153558052435
```

In [51]:

```python
predictions
```

Out[51]:

```
array([0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
       1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1,
       0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 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, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 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, 0, 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 [52]:

```python
from sklearn.metrics import classification_report
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.83      0.90      0.86       163
           1       0.82      0.71      0.76       104

    accuracy                           0.83       267
   macro avg       0.83      0.81      0.81       267
weighted avg       0.83      0.83      0.83       267
```

In [ ]:

```
#RANDOM FOREST
```

In [35]:

```
df = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
df
```

Out[35]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450
...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376

891 rows × 12 columns

In [45]:

```
median_age = df['Age'].median()  # Calculate median of the 'age' column
df['Age'].fillna(median_age, inplace=True)  # Replace NaNs with median
df
```

Out[45]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	28.0	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [46]:

```
y=df['Survived']
y
```

Out[46]:

```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [47]:

```
x = df[['Pclass', 'Sex', 'Age', 'Fare']]  # Step 1: Select relevant columns
x = pd.get_dummies(x, columns=['Sex'], drop_first=True)  # Step 2: One-hot encod
x['Sex_male'] = x['Sex_male'].astype(int)
x
```

Out[47]:

	Pclass	Age	Fare	Sex_male
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

In [48]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,
                                                    random_state=101)
```

In [49]:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, classification_rep
```

In [50]:

```
rf_model = RandomForestClassifier(random_state=101, n_estimators=100)
```

In [51]:

```
rf_model.fit(x_train, y_train)
```

Out[51]:

```
▾ RandomForestClassifier
RandomForestClassifier(random_state=101)
```

```
In [53]: rf_predictions = rf_model.predict(x_test)
```

```
In [55]: model.score(x_test,y_test)
```

```
Out[55]: 0.7985074626865671
```

```
In [56]: accuracy=confusion_matrix(y_test,predictions)
         accuracy
```

```
Out[56]: array([[135,  19],
                [ 35,  79]], dtype=int64)
```

```
In [57]: from sklearn.metrics import classification_report
         print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.79      0.88      0.83       154
           1       0.81      0.69      0.75       114

    accuracy                           0.80       268
   macro avg       0.80      0.78      0.79       268
weighted avg       0.80      0.80      0.80       268
```

```
In [ ]: #SVM
```

```
In [58]: df = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
         df
```

Out[58]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [59]:

```
median_age = df['Age'].median()  # Calculate median of the 'age' column
df['Age'].fillna(median_age, inplace=True)  # Replace NaNs with median
df
```

Out[59]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	28.0	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [60]:

```
y=df['Survived']
y
```

Out[60]:
```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [61]:
```python
x = df[['Pclass', 'Sex', 'Age', 'Fare']]  # Step 1: Select relevant columns
x = pd.get_dummies(x, columns=['Sex'], drop_first=True)  # Step 2: One-hot encod
x['Sex_male'] = x['Sex_male'].astype(int)
x
```

Out[61]:

	Pclass	Age	Fare	Sex_male
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

In [62]:
```python
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,
                                                    random_state=101)
```

In [73]:
```python
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

# Fit on training data and transform both train and test
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

In [75]:
```python
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, accuracy_score, classification_rep
```

In [76]:
```python
svm_model = SVC(kernel='rbf')  # You can also try 'rbf', 'poly', or 'sigmoid'
```

In [77]:
```
svm_model.fit(x_train_scaled, y_train)
```

Out[77]:
```
▾ SVC
SVC()
```

In [78]:
```
svm_predictions = svm_model.predict(x_test_scaled)
```

In [81]:
```
model.score(x_test_scaled,y_test)
```

```
D:\anaconda\Lib\site-packages\sklearn\base.py:439: UserWarning: X does not have v
alid feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

Out[81]:
```
0.23134328358208955
```

In [80]:
```
accuracy=confusion_matrix(y_test,predictions)
accuracy
```

Out[80]:
```
array([[135,  19],
       [ 35,  79]], dtype=int64)
```

In [82]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.79      0.88      0.83       154
           1       0.81      0.69      0.75       114

    accuracy                           0.80       268
   macro avg       0.80      0.78      0.79       268
weighted avg       0.80      0.80      0.80       268
```

In [71]:
```
#DECISION TREEE
```

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

In [3]:
```
df = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
df
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [5]:

```
y=df['Survived']
y
```

Out[5]:

```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [10]:

```python
import pandas as pd

# Assuming your DataFrame is named df
median_age = df['Age'].median()  # Calculate median of the 'age' column
df['Age'].fillna(median_age, inplace=True)  # Replace NaNs with median
df
```

Out[10]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	28.0	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [12]:

```
y=df['Survived']
y
```

Out[12]:

```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [21]:

```
x = df[['Pclass', 'Sex', 'Age', 'Fare']]  # Step 1: Select relevant columns
x = pd.get_dummies(x, columns=['Sex'], drop_first=True)  # Step 2: One-hot encod
x['Sex_male'] = x['Sex_male'].astype(int)
x
```

Out[21]:

	Pclass	Age	Fare	Sex_male
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

In [22]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,
                                                    random_state=101)
```

In [23]:

```
from sklearn import tree
model=tree.DecisionTreeClassifier()
model.fit(x_train,y_train)
```

Out[23]:

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

In [54]:

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

In [25]:

```
y_test
```

Out[25]:

```
331    0
700    1
748    0
751    1
481    0
      ..
388    0
416    1
407    1
482    0
829    1
Name: Survived, Length: 268, dtype: int64
```

In [26]:

```
model.score(x_test,y_test)
```

Out[26]:

```
0.7985074626865671
```

In [27]:

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

In [31]:

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

Out[31]:

```
array([[135,  19],
       [ 35,  79]], dtype=int64)
```

In [32]:

```
from sklearn.metrics import classification_report
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.79      0.88      0.83       154
           1       0.81      0.69      0.75       114

    accuracy                           0.80       268
   macro avg       0.80      0.78      0.79       268
weighted avg       0.80      0.80      0.80       268
```

In [ ]:

```
#NAIVE BAYES
```

In [83]:

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

In [84]:

```
df = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
df
```

Out[84]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [85]:

```
y=df['Survived']
y
```

Out[85]:

```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [86]:

```
# Assuming your DataFrame is named df
median_age = df['Age'].median()  # Calculate median of the 'age' column
df['Age'].fillna(median_age, inplace=True)  # Replace NaNs with median
df
```

Out[86]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	28.0	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [88]:

```
x = df[['Pclass', 'Sex', 'Age', 'Fare']]  # Step 1: Select relevant columns
x = pd.get_dummies(x, columns=['Sex'], drop_first=True)  # Step 2: One-hot encod
x['Sex_male'] = x['Sex_male'].astype(int)
x
```

Out[88]:

	Pclass	Age	Fare	Sex_male
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

In [101…

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,
                                                    random_state=101)
```

In [103…

```
from sklearn.naive_bayes import GaussianNB

model_nb = GaussianNB()              # ✅ Instantiate the model
model_nb.fit(x_train, y_train)       # ✅ Fit the model
```

Out[103…

```
▾ GaussianNB
GaussianNB()
```

In [104…

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

In [105…

```
model.score(x_test,y_test)
```

Out[105…

```
0.7910447761194029
```

In [106…

```
from sklearn.metrics import confusion_matrix
accuracy=confusion_matrix(y_test,predictions)
accuracy
```

Out[106…

```
array([[135,  19],
       [ 37,  77]], dtype=int64)
```

In [107…

```
from sklearn.metrics import classification_report
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.78      0.88      0.83       154
           1       0.80      0.68      0.73       114

    accuracy                           0.79       268
   macro avg       0.79      0.78      0.78       268
weighted avg       0.79      0.79      0.79       268
```

In [108…

```
#KNN
```

In [123…

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

In [124…

```
df = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
df
```

Out[124…

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [126…

```
y=df['Survived']
y
# Assuming your DataFrame is named df
median_age = df['Age'].median()  # Calculate median of the 'age' column
df['Age'].fillna(median_age, inplace=True)  # Replace NaNs with median
```

```
df
x = df[['Pclass', 'Sex', 'Age', 'Fare']]  # Step 1: Select relevant columns
x = pd.get_dummies(x, columns=['Sex'], drop_first=True)  # Step 2: One-hot encod
x['Sex_male'] = x['Sex_male'].astype(int)
x
```

Out[126…

	Pclass	Age	Fare	Sex_male
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

In [127…

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,
                                                    random_state=101)
```

In [128…

```
from sklearn.neighbors import KNeighborsClassifier
model_knn=KNeighborsClassifier(n_neighbors=2)
model_knn.fit(x_train, y_train)       # ✅ Fit the model
```

Out[128…

```
▾      KNeighborsClassifier
KNeighborsClassifier(n_neighbors=2)
```

In [129…

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

In [130…

```
model.score(x_test,y_test)
```

Out[130…

```
0.832089552238806
```

In [131…

```
from sklearn.metrics import confusion_matrix
accuracy=confusion_matrix(y_test,predictions)
accuracy
```

Out[131…

```
array([[143,  11],
       [ 34,  80]], dtype=int64)
```

In [132…

```
from sklearn.metrics import classification_report
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.81      0.93      0.86       154
           1       0.88      0.70      0.78       114

    accuracy                           0.83       268
   macro avg       0.84      0.82      0.82       268
weighted avg       0.84      0.83      0.83       268
```

In [133…

```
#XG BOOST
```

In [134…

```
df = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
df
```

Out[134…

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	1
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	3
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	2
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	3
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	

891 rows × 12 columns

In [135…

```
y=df['Survived']
y
# Assuming your DataFrame is named df
median_age = df['Age'].median()  # Calculate median of the 'age' column
df['Age'].fillna(median_age, inplace=True)  # Replace NaNs with median
```

```
df
x = df[['Pclass', 'Sex', 'Age', 'Fare']]  # Step 1: Select relevant columns
x = pd.get_dummies(x, columns=['Sex'], drop_first=True)  # Step 2: One-hot encod
x['Sex_male'] = x['Sex_male'].astype(int)
x
```

Out[135…

	Pclass	Age	Fare	Sex_male
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

In [136…

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,
                                                    random_state=101)
```

In [137…

```
!pip install xgboost
from xgboost import XGBClassifier
```

```
Requirement already satisfied: xgboost in d:\anaconda\lib\site-packages (3.0.1)
Requirement already satisfied: numpy in d:\anaconda\lib\site-packages (from xgboo
st) (1.26.4)
Requirement already satisfied: scipy in d:\anaconda\lib\site-packages (from xgboo
st) (1.11.4)
```

In [138…

```
model = XGBClassifier(use_label_encoder=False, eval_metric='logloss', random_sta
model.fit(x_train, y_train)

# Predict on test data
y_pred = model.predict(x_test)
```

```
D:\anaconda\Lib\site-packages\xgboost\training.py:183: UserWarning: [16:32:55] WA
RNING: C:\actions-runner\_work\xgboost\xgboost\src\learner.cc:738:
Parameters: { "use_label_encoder" } are not used.

  bst.update(dtrain, iteration=i, fobj=obj)
```

In [139…

```
model.score(x_test,y_test)
```

Out[139…

```
0.832089552238806
```

In [140…

```
from sklearn.metrics import confusion_matrix
accuracy=confusion_matrix(y_test,predictions)
accuracy
```

Out[140…

```
array([[143,  11],
       [ 34,  80]], dtype=int64)
```

In [141…

```
from sklearn.metrics import classification_report
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.81      0.93      0.86       154
           1       0.88      0.70      0.78       114

    accuracy                           0.83       268
   macro avg       0.84      0.82      0.82       268
weighted avg       0.84      0.83      0.83       268
```

In [ ]:

```
#ROC AND AUC CURVE FOR XG BOOST(OUR BEST MODEL)
```

In [10]:

```
# Step 1: Import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report, roc_curve,
from xgboost import XGBClassifier
```

In [11]:

```
# Step 2: Load the dataset
df = pd.read_csv('C:/Users/user/Downloads/titanic_train.csv')
```

In [12]:

```
# Step 3: Handle missing values (e.g., Age)
median_age = df['Age'].median()
df['Age'].fillna(median_age, inplace=True)
```

In [13]:

```
# Step 4: Define features (X) and target (y)
x = df[['Pclass', 'Sex', 'Age', 'Fare']]
x = pd.get_dummies(x, columns=['Sex'], drop_first=True)  # Convert 'Sex' to nume
y = df['Survived']
```

In [14]:

```
# Step 5: Split the data
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_
```

In [15]:

```
# Step 6: Train the XGBoost model
model = XGBClassifier(use_label_encoder=False, eval_metric='logloss', random_sta
model.fit(x_train, y_train)
```

```
D:\anaconda\Lib\site-packages\xgboost\training.py:183: UserWarning: [20:05:05] WA
RNING: C:\actions-runner\_work\xgboost\xgboost\src\learner.cc:738: 
Parameters: { "use_label_encoder" } are not used.

  bst.update(dtrain, iteration=i, fobj=obj)
```

Out[15]:

▾ XGBClassifier

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rou
nds=None,
              enable_categorical=False, eval_metric='logloss',
              feature_types=None, feature_weights=None, gamma=None,
              grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_
bin=None,
```

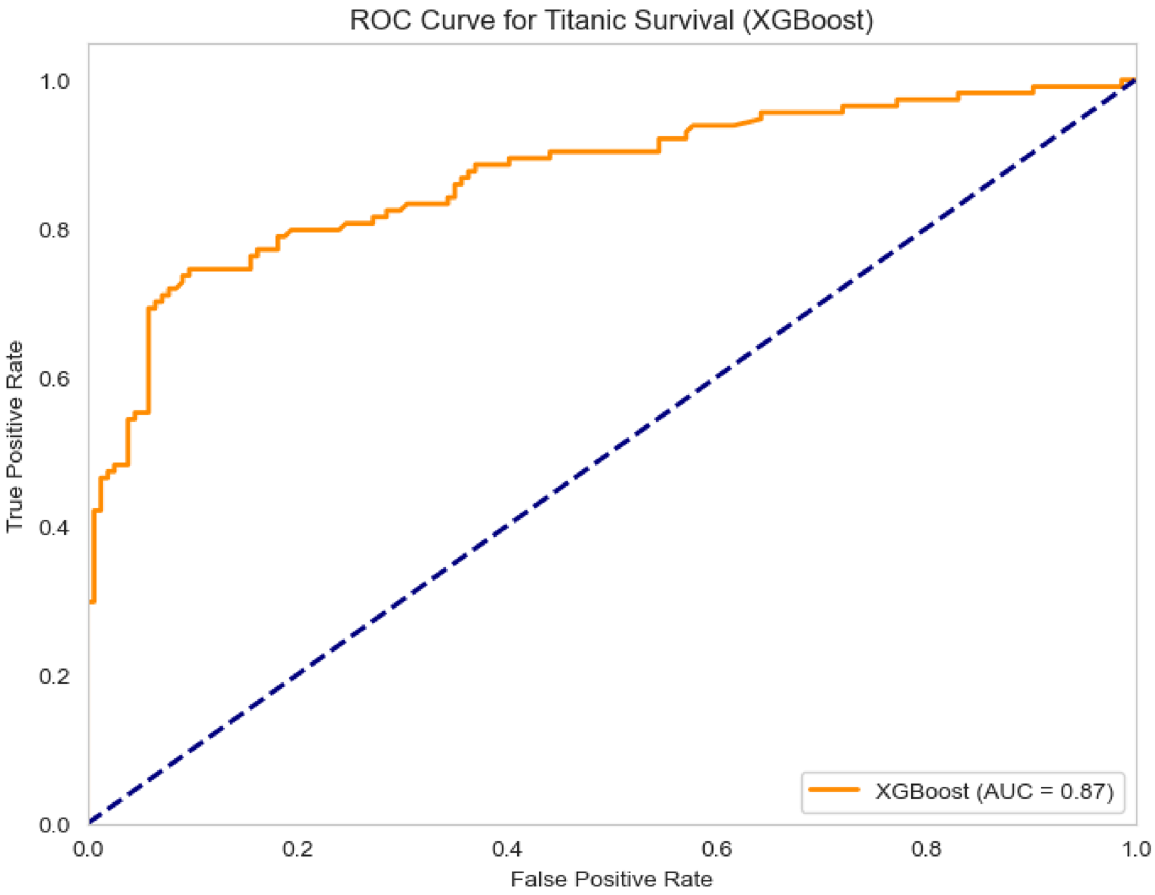
In [16]:

```python
# Step 7: Predictions and evaluation
y_pred = model.predict(x_test)

# Confusion matrix
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

# Classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Confusion Matrix:
[[143  11]
 [ 34  80]]

Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.93      0.86       154
           1       0.88      0.70      0.78       114

    accuracy                           0.83       268
   macro avg       0.84      0.82      0.82       268
weighted avg       0.84      0.83      0.83       268
```

In [17]:

```python
# Get predicted probabilities for the positive class (Survived=1)
y_prob = model.predict_proba(x_test)[:, 1]

# Calculate FPR, TPR, thresholds
fpr, tpr, thresholds = roc_curve(y_test, y_prob)

# Compute AUC
roc_auc = auc(fpr, tpr)

# Plot ROC Curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='XGBoost (AUC = %0.2f)' % roc
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')  # Diagonal line
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve for Titanic Survival (XGBoost)')
plt.legend(loc="lower right")
plt.grid()
plt.show()
```



In [18]:

```
print("AUC Score:", roc_auc)
```

```
AUC Score: 0.8703292321713374
```