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

Load Data and shows first 5 rows of data.

```
data = pd.read_csv(r"C:\Users\Pavilion-14\OneDrive\Desktop\Titanic-dataset.csv")
data.head()

   PassengerId  Survived  Pclass  \
0            1         0       3   
1            2         1       1   
2            3         1       3   
3            4         1       1   
4            5         0       3   

                                                Name     Sex   Age
SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0
1   
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0
1   
2                             Heikkinen, Miss. Laina  female  26.0
0   
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0
1   
4                           Allen, Mr. William Henry    male  35.0
0   

   Parch            Ticket     Fare Cabin Embarked  
0      0         A/5 21171   7.2500   NaN        S  
1      0          PC 17599  71.2833   C85        C  
2      0  STON/O2. 3101282   7.9250   NaN        S  
3      0            113803  53.1000  C123        S  
4      0            373450   8.0500   NaN        S  
```

checking the datatype.

```
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Calculates the number of rows and columns.

```
data.shape

(891, 12)
```

Calculates statistical values.

```
data.describe()

       PassengerId    Survived      Pclass         Age       SibSp  \
count   891.000000  891.000000  891.000000  714.000000  891.000000
mean    446.000000    0.383838    2.308642   29.699118    0.523008
std     257.353842    0.486592    0.836071   14.526497    1.102743
min       1.000000    0.000000    1.000000    0.420000    0.000000
25%     223.500000    0.000000    2.000000   20.125000    0.000000
50%     446.000000    0.000000    3.000000   28.000000    0.000000
75%     668.500000    1.000000    3.000000   38.000000    1.000000
max     891.000000    1.000000    3.000000   80.000000    8.000000

            Parch        Fare
count  891.000000  891.000000
mean     0.381594   32.204208
std      0.806057   49.693429
min      0.000000    0.000000
25%      0.000000    7.910400
50%      0.000000   14.454200
75%      0.000000   31.000000
max      6.000000  512.329200
```

checking the number of rows and column of new data.

```
data.shape

(891, 12)
```

Mark null values as True and returns sum of number of True values in each column in new df.

```
data.isnull().sum()

PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

Calculates statistical values for new data.

```
data.describe()

       PassengerId    Survived      Pclass         Age       SibSp  \
count   891.000000  891.000000  891.000000  714.000000  891.000000   
mean    446.000000    0.383838    2.308642   29.699118    0.523008   
std     257.353842    0.486592    0.836071   14.526497    1.102743   
min       1.000000    0.000000    1.000000    0.420000    0.000000   
25%     223.500000    0.000000    2.000000   20.125000    0.000000   
50%     446.000000    0.000000    3.000000   28.000000    0.000000   
75%     668.500000    1.000000    3.000000   38.000000    1.000000   
max     891.000000    1.000000    3.000000   80.000000    8.000000   

            Parch        Fare  
count  891.000000  891.000000  
mean     0.381594   32.204208  
std      0.806057   49.693429  
min      0.000000    0.000000  
25%      0.000000    7.910400  
50%      0.000000   14.454200  
75%      0.000000   31.000000  
max      6.000000  512.329200  
```

data prepoccessing.

```
scaler = StandardScaler()
data[['Age','Fare']] = scaler.fit_transform(data[['Age','Fare']])
data.head()


   PassengerId  Survived  Pclass  \
0            1         0       3   
1            2         1       1   
```

```
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex        Age
SibSp  \
0                            Braund, Mr. Owen Harris    male -0.530377
1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  0.571831
1
2                             Heikkinen, Miss. Laina  female -0.254825
0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  0.365167
1
4                           Allen, Mr. William Henry    male  0.365167
0

   Parch            Ticket      Fare Cabin Embarked
0      0         A/5 21171 -0.502445   NaN        S
1      0          PC 17599  0.786845   C85        C
2      0  STON/O2. 3101282 -0.488854   NaN        S
3      0            113803  0.420730  C123        S
4      0            373450 -0.486337   NaN        S
```

Mark null values as True and returns sum of number of True values in each column.

```
data['Age'].fillna(data['Age'].mean(), inplace=True)
data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
data.isnull().sum()

C:\Users\Pavilion-14\AppData\Local\Temp\
ipykernel_25380\846610028.py:1: FutureWarning: A value is trying to be
set on a copy of a DataFrame or Series through chained assignment
using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Age'].fillna(data['Age'].mean(), inplace=True)
C:\Users\Pavilion-14\AppData\Local\Temp\
ipykernel_25380\846610028.py:2: FutureWarning: A value is trying to be
set on a copy of a DataFrame or Series through chained assignment
using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
```

```
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         0
dtype: int64
```

>categorical data converter into numaric data using OneHotEncoder.

```
data.replace({'Sex':{'male':0,'female':1}}, inplace=True)
data.replace({'Embarked':{'S':0,'C':1,'Q':2}}, inplace=True)
data.head()
```

```
C:\Users\Pavilion-14\AppData\Local\Temp\
ipykernel_25380\1226164553.py:2: FutureWarning: Downcasting behavior
in `replace` is deprecated and will be removed in a future version. To
retain the old behavior, explicitly call
`result.infer_objects(copy=False)`. To opt-in to the future behavior,
set `pd.set_option('future.no_silent_downcasting', True)`
  data.replace({'Embarked':{'S':0,'C':1,'Q':2}}, inplace=True)
```

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name  Sex       Age
SibSp  \
0                            Braund, Mr. Owen Harris    0 -0.530377
1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...    1  0.571831
1
2                               Heikkinen, Miss. Laina    1 -0.254825
0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)    1  0.365167
1
4                             Allen, Mr. William Henry    0  0.365167
0

   Parch            Ticket      Fare Cabin  Embarked
0      0         A/5 21171 -0.502445   NaN         0
1      0          PC 17599  0.786845   C85         1
2      0  STON/O2. 3101282 -0.488854   NaN         0
3      0            113803  0.420730  C123         0
4      0            373450 -0.486337   NaN         0
```

check imbalance data and set into balance data.

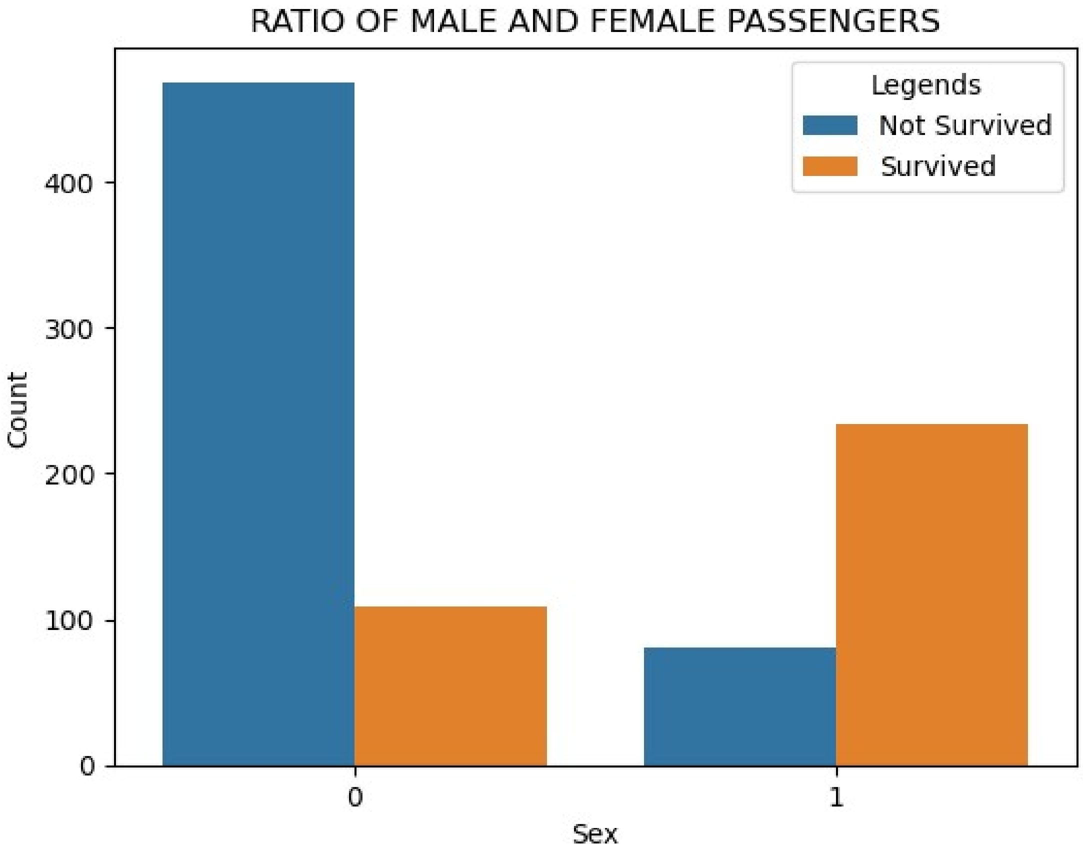
```
data['Survived'].value_counts()

Survived
0    549
1    342
Name: count, dtype: int64
```

Data Visualization.

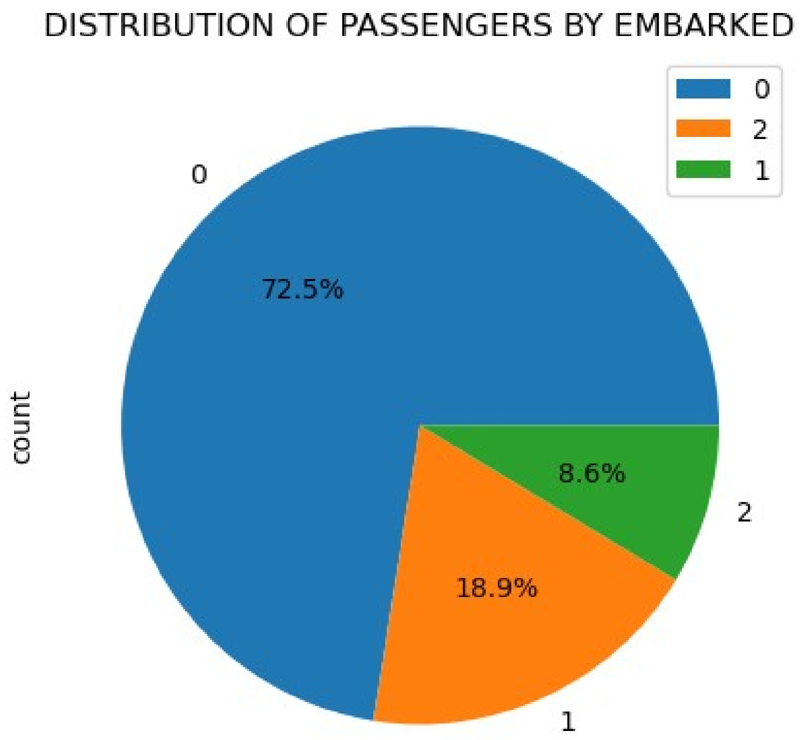
```
from imblearn.under_sampling import RandomUnderSampler
rus = RandomUnderSampler()
X_train_rus, Y_train_rus = 
rus.fit_resample(data.drop(['Survived'],axis=1), data['Survived'])
print("After undersampling the shape of train_X: ", X_train_rus.shape)
print("After undersampling the shape of train_Y: ", Y_train_rus.shape)
print("After undersampling the value counts of target variable: ", 
Y_train_rus.value_counts())

After undersampling the shape of train_X:  (684, 11)
After undersampling the shape of train_Y:  (684,)
After undersampling the value counts of target variable:  Survived
0    342
1    342
Name: count, dtype: int64
```

Data Visualization

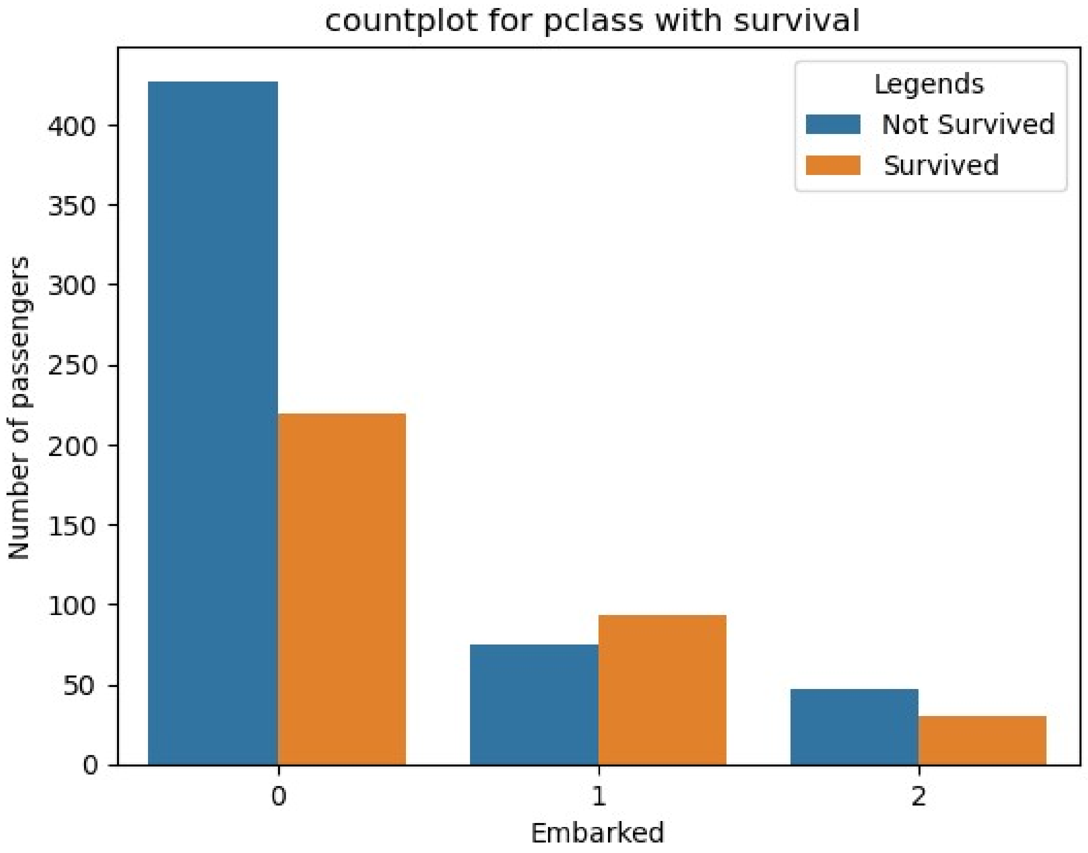
```
sns.countplot(x='Sex',data=data,hue='Survived')
plt.legend(title = "Legends", labels = ["Not Survived", "Survived"])
plt.title("RATIO OF MALE AND FEMALE PASSENGERS")
plt.ylabel("Count")
plt.xlabel("Sex")
plt.show()
```



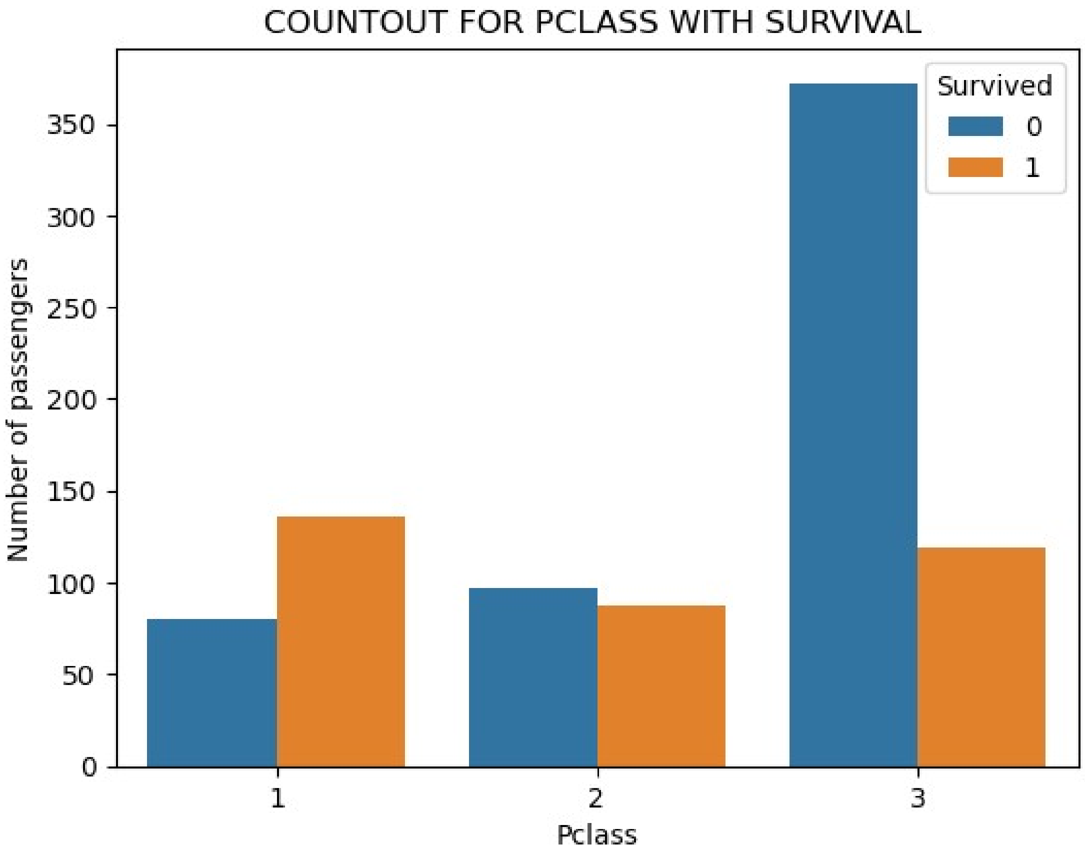
```
data['Embarked'].value_counts().plot(kind='pie',figsize=(6,5),autopct=
'%1.1f%%')
plt.title("DISTRIBUTION OF PASSENGERS BY EMBARKED")
plt.legend({'0','1','2'})
plt.show()
```



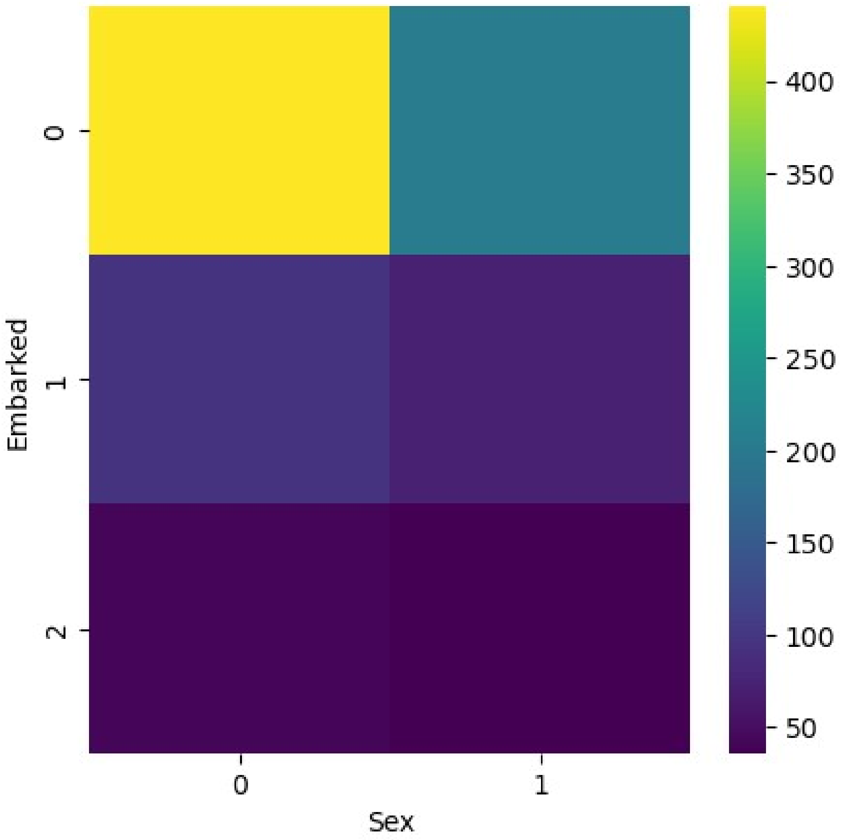
```
sns.countplot(data = data, x = "Embarked", hue = "Survived")
plt.title("countplot for Embarked with survival")
plt.xlabel("Embarked")
plt.ylabel("Number of passengers")
plt.legend(title = "Legends", labels = ["Not Survived", "Survived"])
plt.show()
```



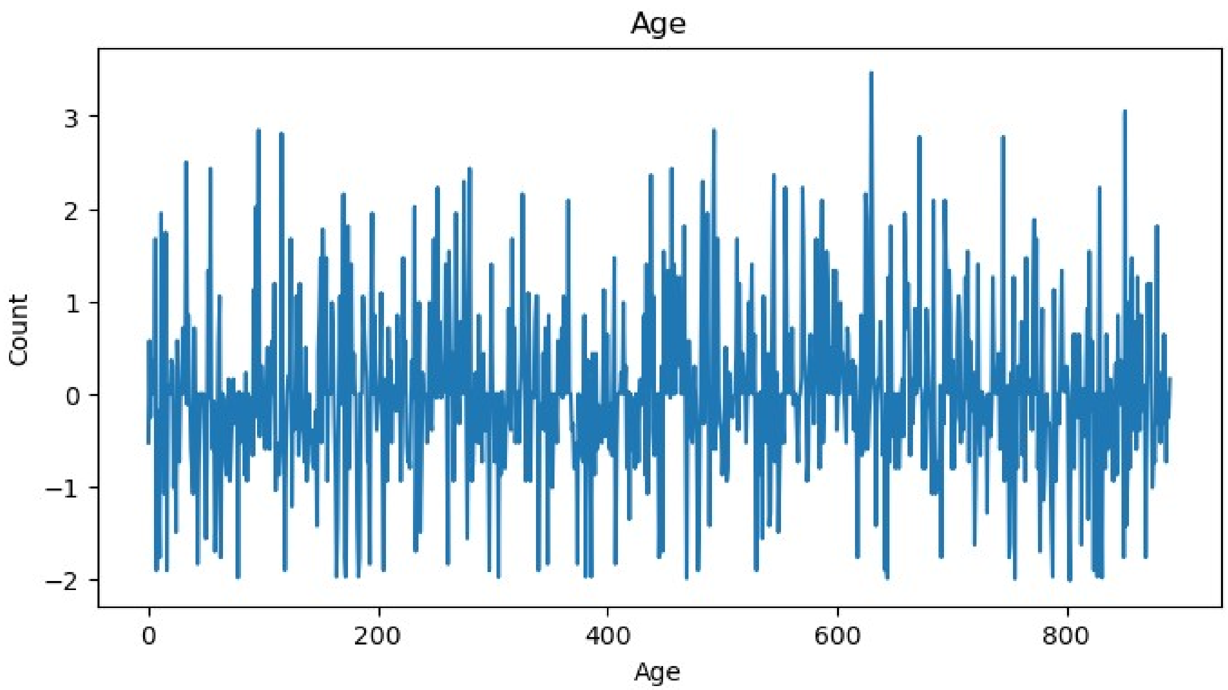
```
sns.countplot(data =data, x = "Pclass", hue = "Survived")
plt.title("COUNTOUT FOR PCLASS WITH SURVIVAL ")
plt.xlabel("Pclass")
plt.ylabel("Number of passengers")


Text(0, 0.5, 'Number of passengers')
```



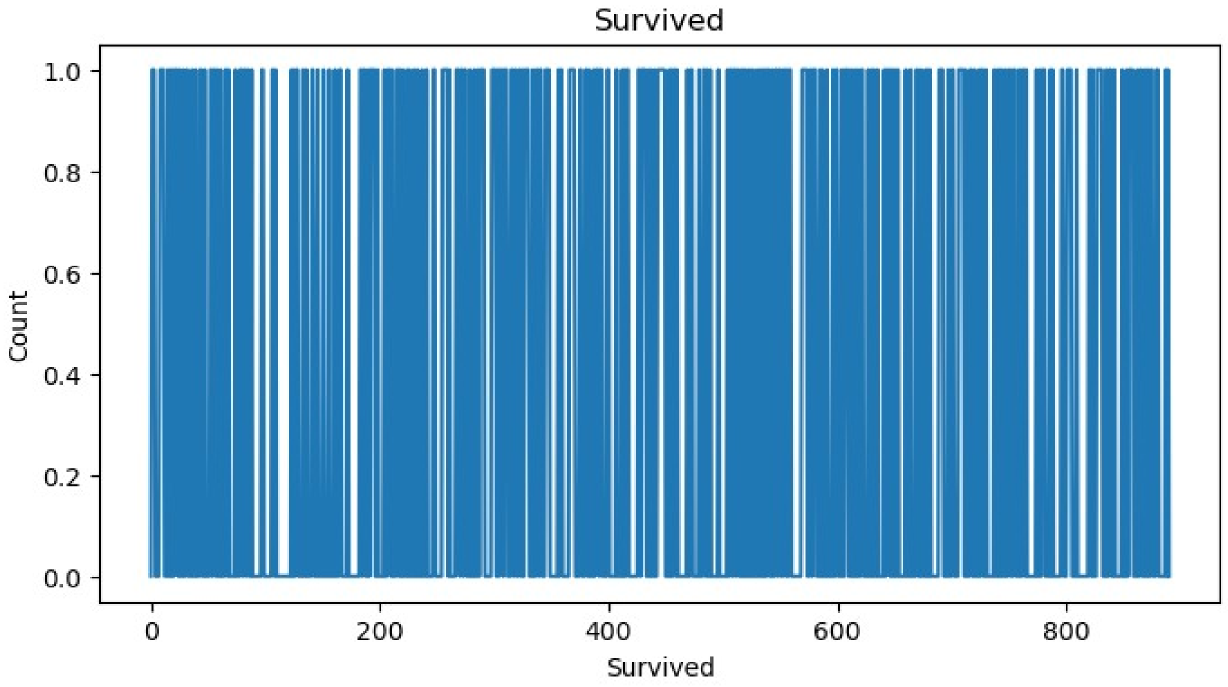
```
plt.subplots(figsize=(5, 5))
df_2dhist = pd.DataFrame({
    x_label: grp['Embarked'].value_counts()
    for x_label, grp in data.groupby('Sex')
})
sns.heatmap(df_2dhist, cmap='viridis')
plt.xlabel('Sex')
plt.ylabel('Embarked')
plt.show()
```



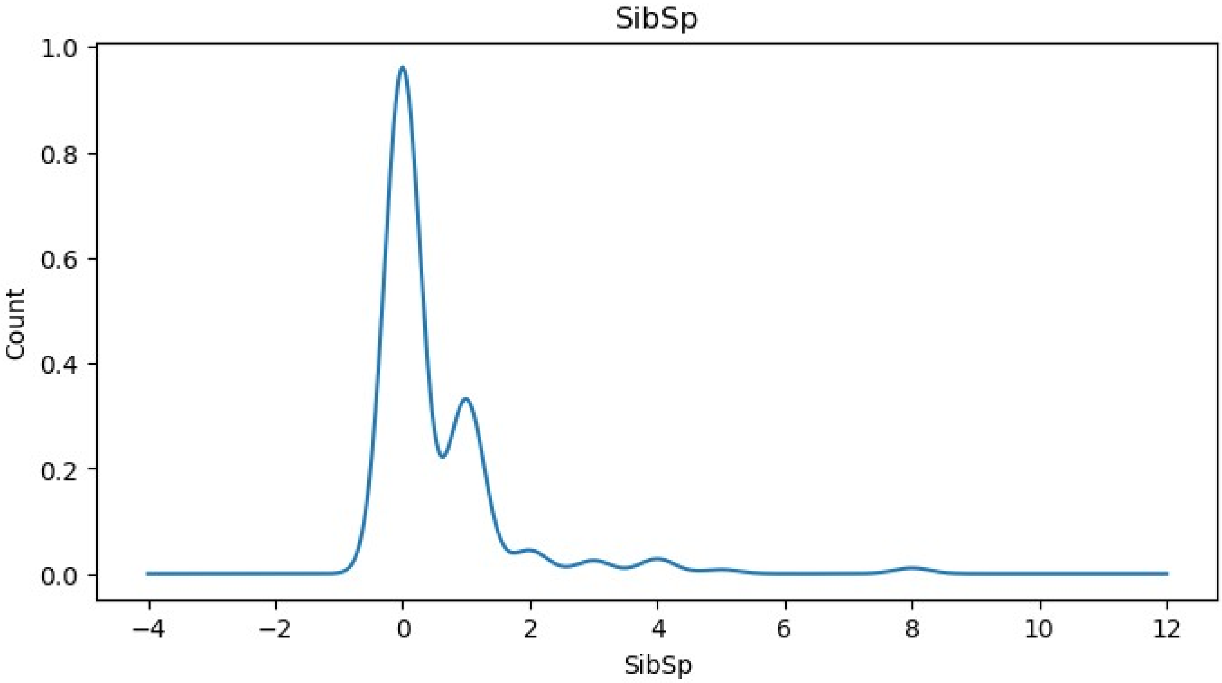
```
data['Age'].plot(kind='line', figsize=(8, 4), title='Age')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



```
data['Survived'].plot(kind='line', figsize=(8, 4), title='Survived')
plt.xlabel('Survived')
plt.ylabel('Count')
plt.show()
```



```
data['SibSp'].plot(kind='density', figsize=(8, 4), title='SibSp')
plt.xlabel('SibSp')
plt.ylabel('Count')
plt.show()
```



Deviding the data into Dependent and Independent variables.

```
x= data.drop(columns=['Survived'],axis=1)
y= data['Survived']


print(x)

     PassengerId  Pclass
Name  \
0              1       3                            Braund, Mr. Owen
Harris
1              2       1  Cumings, Mrs. John Bradley (Florence Briggs
Th...
2              3       3                             Heikkinen, Miss.
Laina
3              4       1       Futrelle, Mrs. Jacques Heath (Lily May
Peel)
4              5       3                           Allen, Mr. William
Henry
..           ...     ...
...
```

```
886          887       2                              Montvila, Rev.
Juozas
887          888       1                       Graham, Miss. Margaret
Edith
888          889       3           Johnston, Miss. Catherine Helen
"Carrie"
889          890       1                              Behr, Mr. Karl
Howell
890          891       3                                Dooley, Mr.
Patrick

     Sex           Age  SibSp  Parch            Ticket      Fare Cabin
\
0      0 -5.303766e-01      1      0         A/5 21171 -0.502445   NaN

1      1  5.718310e-01      1      0          PC 17599  0.786845   C85

2      1 -2.548247e-01      0      0  STON/O2. 3101282 -0.488854   NaN

3      1  3.651671e-01      1      0            113803  0.420730  C123

4      0  3.651671e-01      0      0            373450 -0.486337   NaN

..   ...           ...    ...    ...               ...       ...   ...

886    0 -1.859368e-01      0      0            211536 -0.386671   NaN

887    1 -7.370406e-01      0      0            112053 -0.044381   B42

888    1  2.388379e-16      1      2        W./C. 6607 -0.176263   NaN

889    0 -2.548247e-01      0      0            111369 -0.044381  C148

890    0  1.585031e-01      0      0            370376 -0.492378   NaN


     Embarked
0           0
1           1
2           0
3           0
4           0
..        ...
886         0
887         0
888         0
889         1
890         2

[891 rows x 11 columns]
```

```
print(y)

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
```

Deviding the cleaned data into training and testing sets and checking the null value in train and test data

```
x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.2,random_state=42)
x_train.isnull().sum()
x_test.isnull().sum()

PassengerId      0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          134
Embarked         0
dtype: int64
```

```
x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.2,random_state=42)
x_train.isnull().sum()
x_test.isnull().sum()

PassengerId      0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
```

```
Cabin          134
Embarked         0
dtype: int64
```

```
x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.2,random_state=42)
x_train.isnull().sum()
x_test.isnull().sum()
```

```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          134
Embarked         0
dtype: int64
```