ASSIGNMENT - 3

Name : R.Nikhila Manogna

Reg no : 21BCE7281

Performing data preprocessing on titanic dataset

Data Preprocessing

```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Import the dataset
df=pd.read_csv("Titanic-Dataset.csv")

df.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

df.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  
```

```
df.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
```

```
df.shape
```

```
(891, 12)
```

```
df.corr()
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_19624\1134722465.py:1:
FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only
```

```
valid columns or specify the value of numeric_only to silence this
warning.
  df.corr()

             PassengerId  Survived    Pclass       Age     SibSp
Parch  \
PassengerId     1.000000 -0.005007 -0.035144  0.036847 -0.057527 -
0.001652
Survived       -0.005007  1.000000 -0.338481 -0.077221 -0.035322
0.081629
Pclass         -0.035144 -0.338481  1.000000 -0.369226  0.083081
0.018443
Age             0.036847 -0.077221 -0.369226  1.000000 -0.308247 -
0.189119
SibSp          -0.057527 -0.035322  0.083081 -0.308247  1.000000
0.414838
Parch          -0.001652  0.081629  0.018443 -0.189119  0.414838
1.000000
Fare            0.012658  0.257307 -0.549500  0.096067  0.159651
0.216225

                 Fare
PassengerId  0.012658
Survived     0.257307
Pclass      -0.549500
Age          0.096067
SibSp        0.159651
Parch        0.216225
Fare         1.000000
```

```
df.corr().Fare.sort_values(ascending=False)
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_19624\60082530.py:1:
FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only
valid columns or specify the value of numeric_only to silence this
warning.
  df.corr().Fare.sort_values(ascending=False)

Fare           1.000000
Survived       0.257307
Parch          0.216225
SibSp          0.159651
Age            0.096067
PassengerId    0.012658
Pclass        -0.549500
Name: Fare, dtype: float64
```

```
# Checking for null values
df.isnull().any()
```

```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age             True
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin           True
Embarked        True
dtype: bool
```

```
df.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
```

```
df[df['Age'].isnull()]
```

```
     PassengerId  Survived  Pclass
Name  \
5              6         0       3                          Moran, Mr.
James
17            18         1       2              Williams, Mr. Charles
Eugene
19            20         1       3                   Masselmani, Mrs.
Fatima
26            27         0       3                   Emir, Mr. Farred
Chehab
28            29         1       3             O'Dwyer, Miss. Ellen
"Nellie"
..           ...       ...     ...
...
859          860         0       3                          Razi, Mr.
Raihed
863          864         0       3         Sage, Miss. Dorothy Edith
"Dolly"
```

```
868          869         0       3                van Melkebeke, Mr.
Philemon
878          879         0       3                         Laleff, Mr.
Kristo
888          889         0       3  Johnston, Miss. Catherine Helen
"Carrie"

        Sex  Age  SibSp  Parch      Ticket     Fare Cabin Embarked
5      male  NaN      0      0      330877   8.4583   NaN        Q
17     male  NaN      0      0      244373  13.0000   NaN        S
19   female  NaN      0      0        2649   7.2250   NaN        C
26     male  NaN      0      0        2631   7.2250   NaN        C
28   female  NaN      0      0      330959   7.8792   NaN        Q
..      ...  ...    ...    ...         ...      ...   ...      ...
859    male  NaN      0      0        2629   7.2292   NaN        C
863  female  NaN      8      2    CA. 2343  69.5500   NaN        S
868    male  NaN      0      0      345777   9.5000   NaN        S
878    male  NaN      0      0      349217   7.8958   NaN        S
888  female  NaN      1      2  W./C. 6607  23.4500   NaN        S

[177 rows x 12 columns]
```

```
mean_age=round(df['Age'].mean(), 1)
mean_age
```

```
29.7
```

```
# Mean imputation for null values in age column
df['Age'].replace(np.nan,mean_age,inplace=True)
```

```
# Null values in age column have been imputed by mean
df.isnull().sum()
```

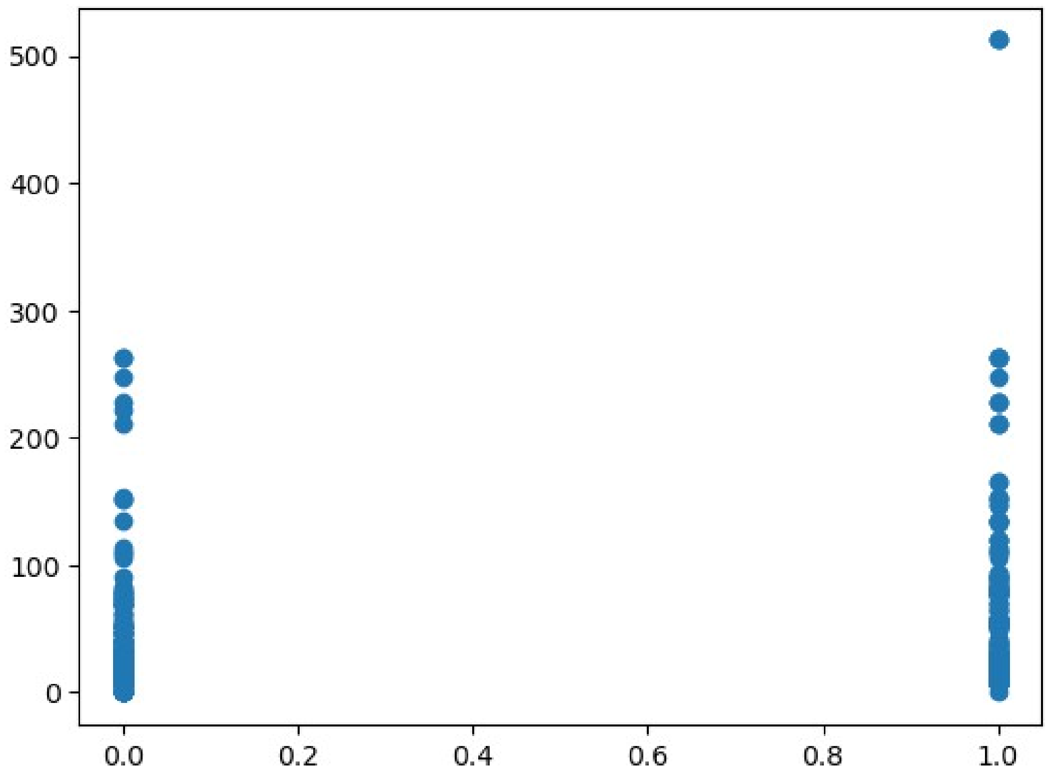
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

```
mode_embarked=df['Embarked'].mode()[0]
mode_embarked
```

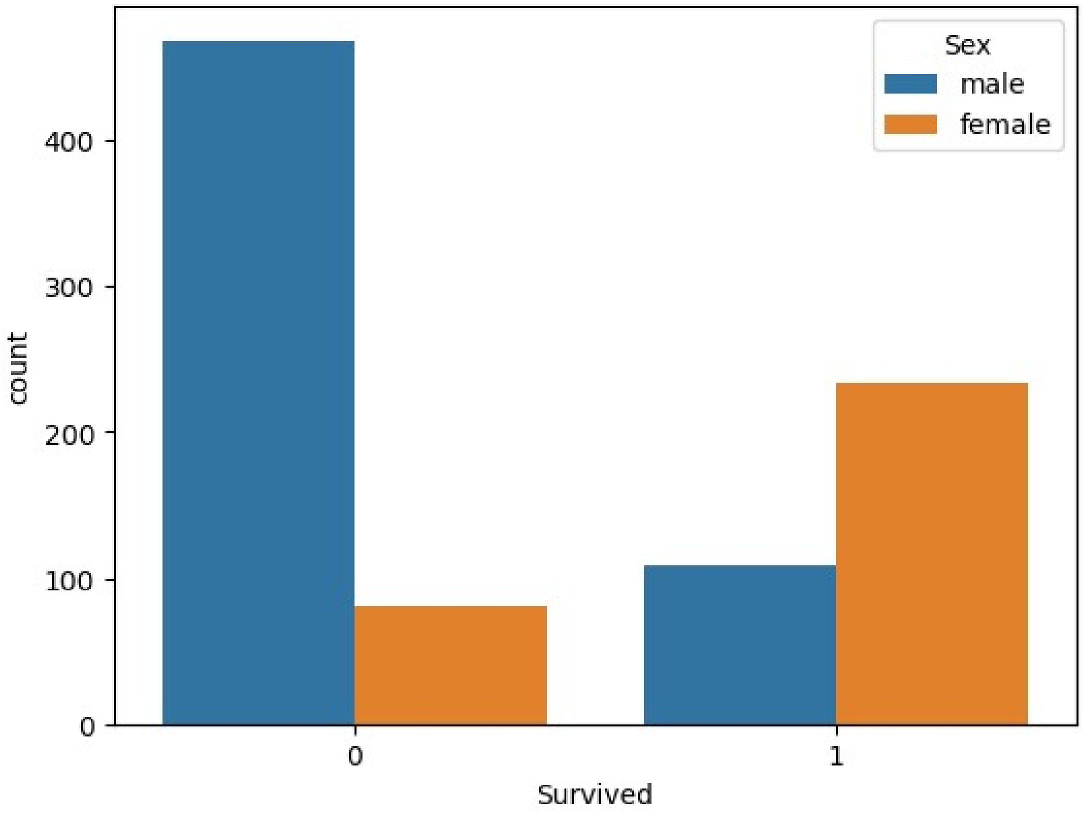
```
'S'
```

```
# Mode imputation for null values in embarked column
df['Embarked'].replace(np.nan, mode_embarked, inplace=True)

# Null values in embarked column have been imputed by mode
df.isnull().sum()

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

# Dropping cabin columns because it contains almost 80% of null values
df.drop(columns='Cabin',inplace=True)

df.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 Embarked
0      0         A/5 21171   7.2500        S
1      0          PC 17599  71.2833        C
2      0  STON/O2. 3101282   7.9250        S
3      0            113803  53.1000        S
4      0            373450   8.0500        S
```

```
# Data visualisation
plt.scatter(df["Survived"],df["Fare"])

<matplotlib.collections.PathCollection at 0x1c2382e6ed0>
```

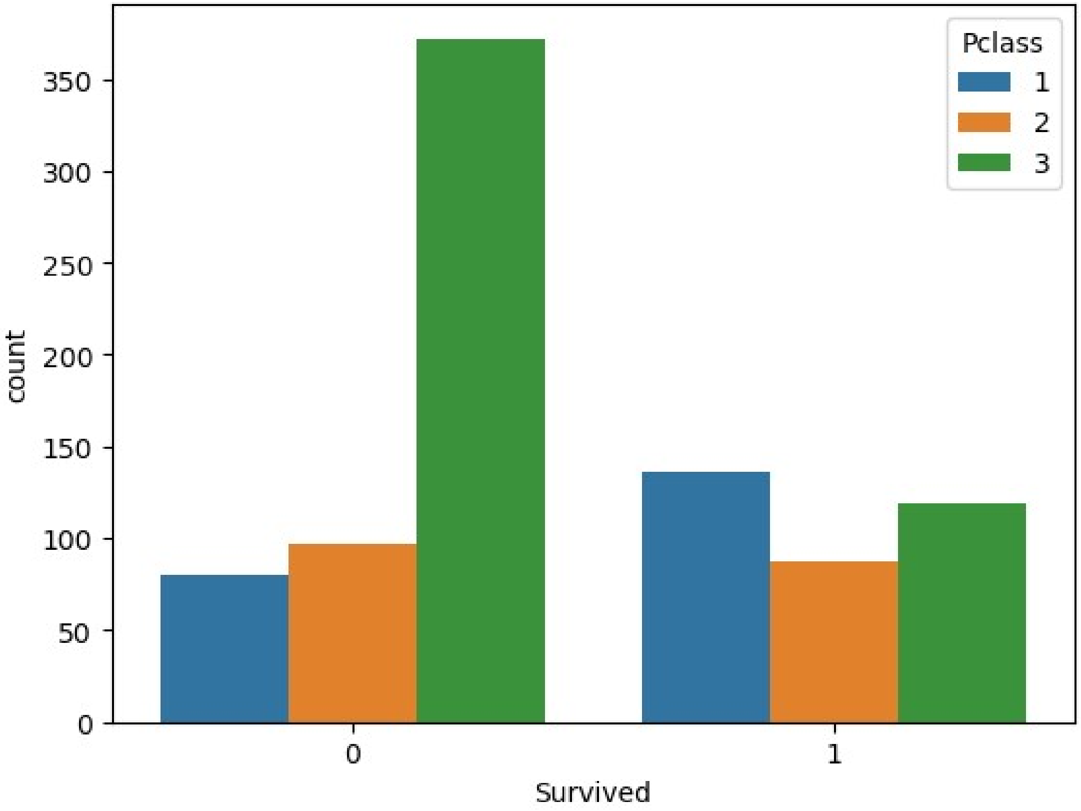


```
sns.countplot(x="Survived",data=df,hue="Sex")

<Axes: xlabel='Survived', ylabel='count'>
```



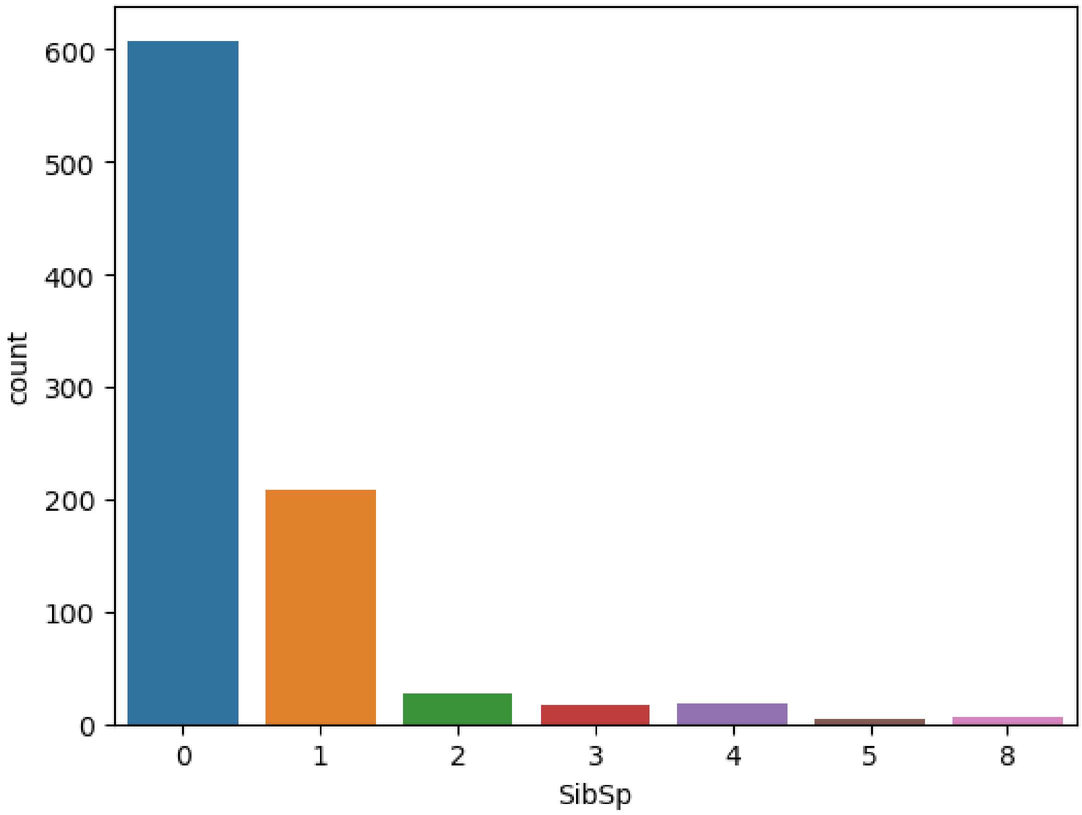
```
sns.countplot(x="Survived",data=df,hue="Pclass")
```

```
<Axes: xlabel='Survived', ylabel='count'>
```



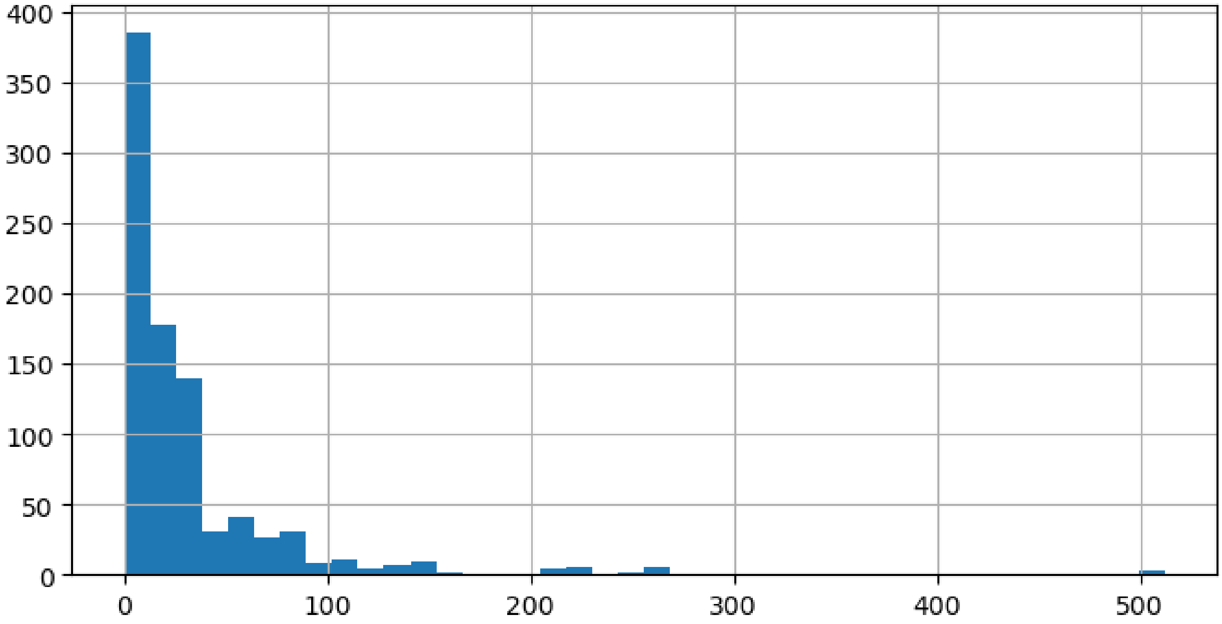
```
sns.countplot(x="SibSp",data=df)
```

```
<Axes: xlabel='SibSp', ylabel='count'>
```



```
df["Fare"].hist(bins=40,figsize=(8,4))
```

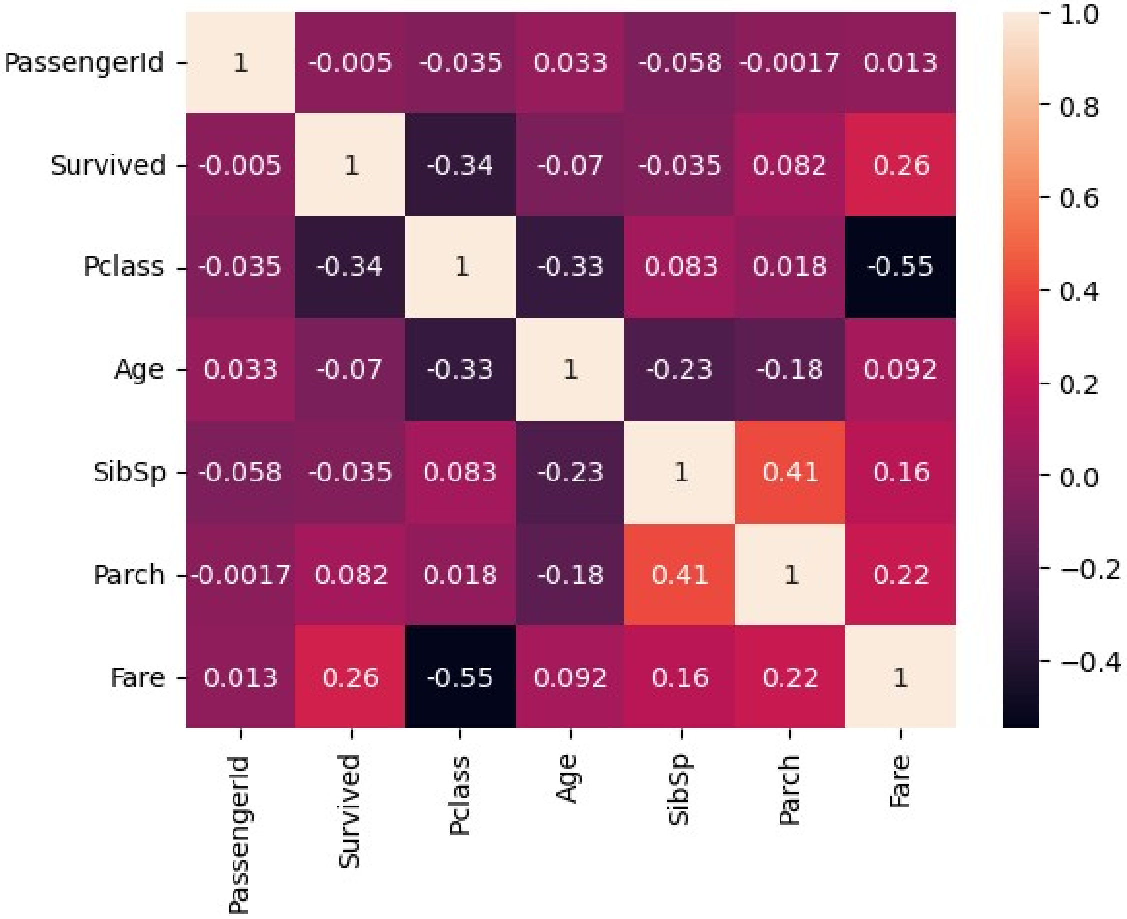
```
<Axes: >
```



```
sns.heatmap(df.corr(),annot=True)
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_19624\4277794465.py:1:
FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only
valid columns or specify the value of numeric_only to silence this
warning.
  sns.heatmap(df.corr(),annot=True)
```

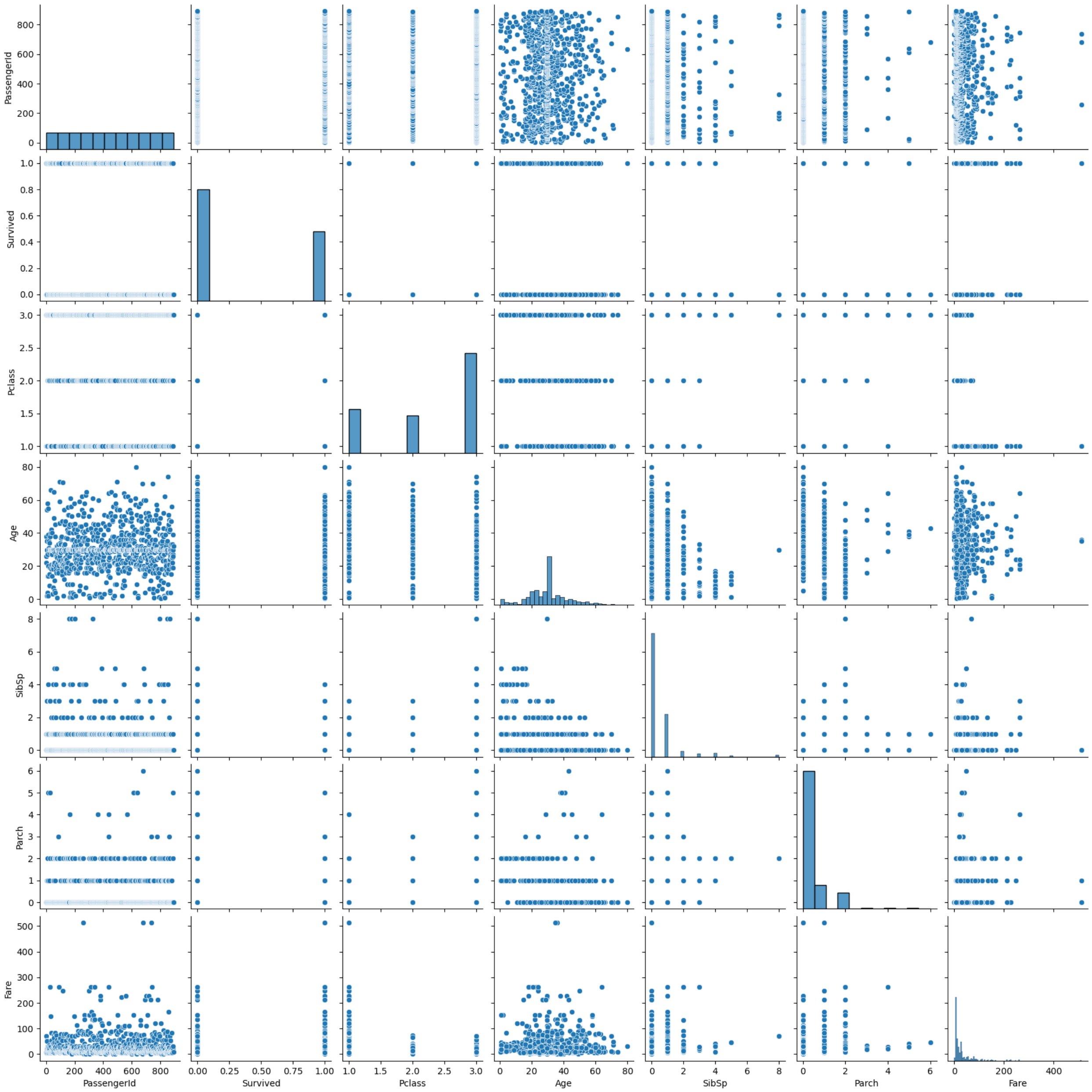
```
<Axes: >
```



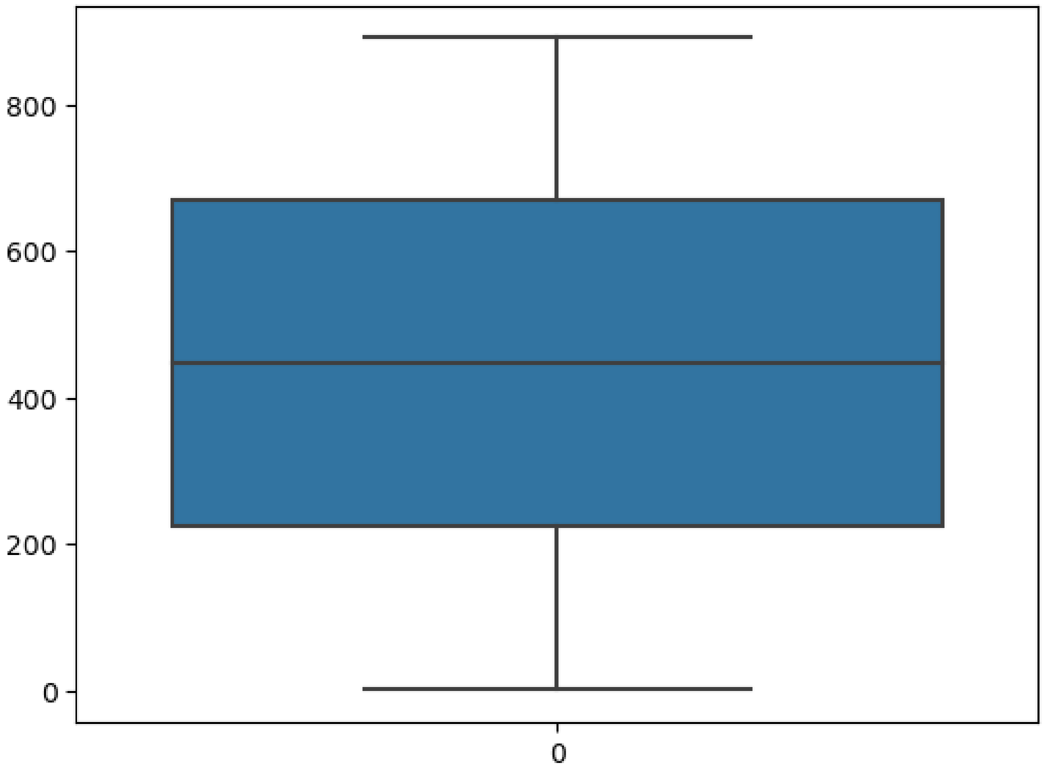
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1	-0.005	-0.035	0.033	-0.058	-0.0017	0.013
Survived	-0.005	1	-0.34	-0.07	-0.035	0.082	0.26
Pclass	-0.035	-0.34	1	-0.33	0.083	0.018	-0.55
Age	0.033	-0.07	-0.33	1	-0.23	-0.18	0.092
SibSp	-0.058	-0.035	0.083	-0.23	1	0.41	0.16
Parch	-0.0017	0.082	0.018	-0.18	0.41	1	0.22
Fare	0.013	0.26	-0.55	0.092	0.16	0.22	1

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x1c23833ded0>
```

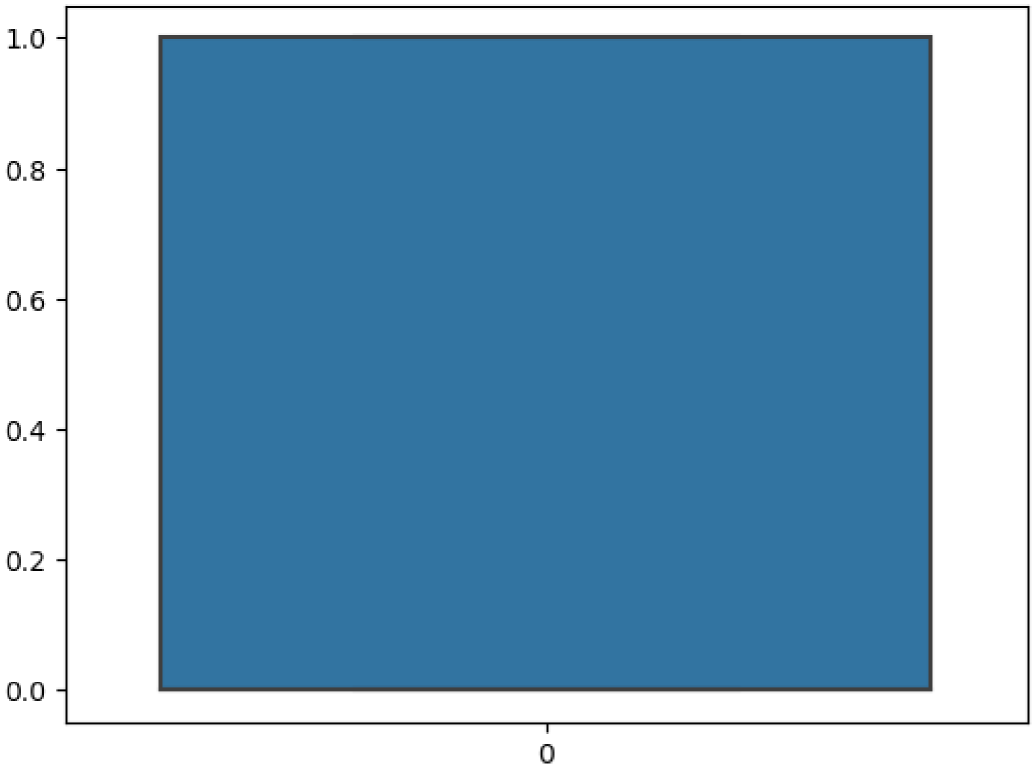


```
# Outlier detection
sns.boxplot(df["PassengerId"])

<Axes: >
```



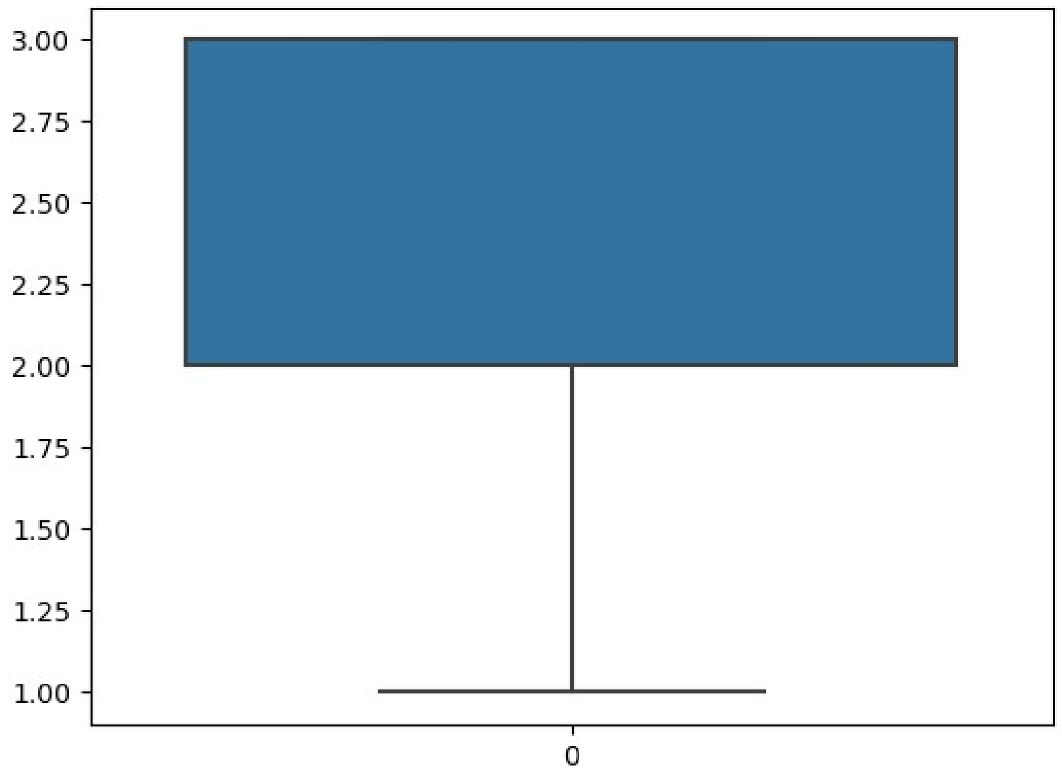
```
sns.boxplot(df["Survived"])
```

```
<Axes: >
```



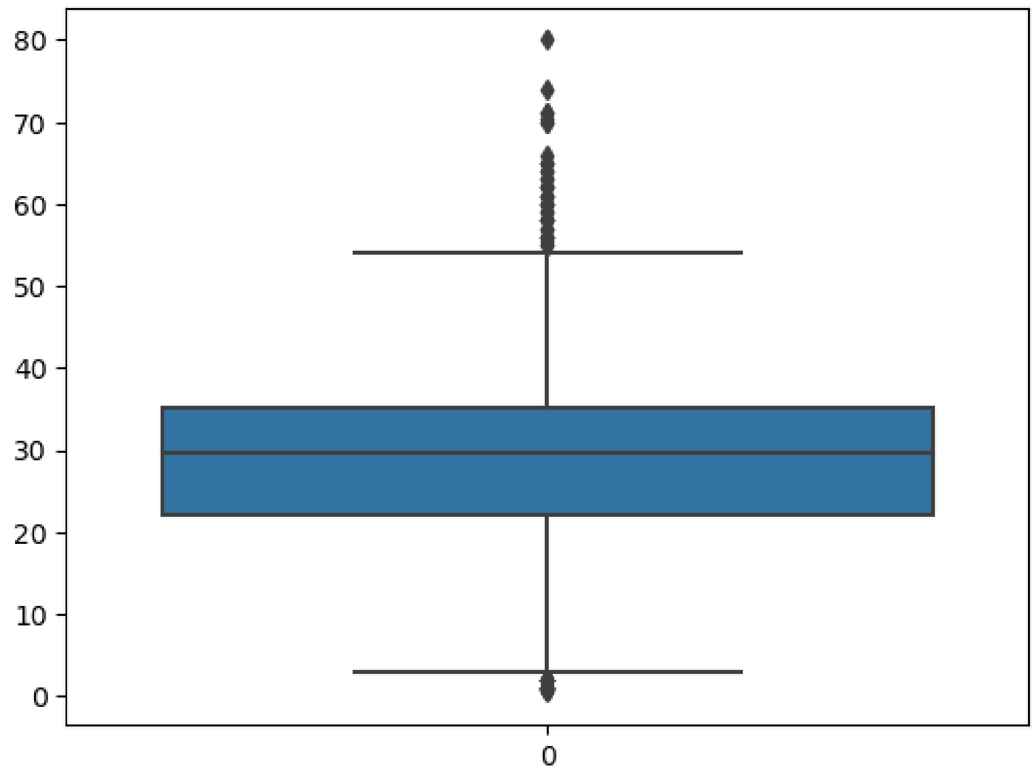
```
sns.boxplot(df["Pclass"])
```

```
<Axes: >
```



```
sns.boxplot(df["Age"])
```

```
<Axes: >
```



```
# Outlier removal by replacement with median
q1=df.Age.quantile(0.25)
q3=df.Age.quantile(0.75)
```

```
q1
```

```
22.0
```

```
q3
```

```
35.0
```

```
IQR=q3-q1
IQR
```

```
13.0
```

```
upper_limit=q3+1.5*IQR
upper_limit
```

```
54.5
```

```
lower_limit=q1-1.5*IQR
lower_limit
```

```
2.5
```

```
df.median()
```

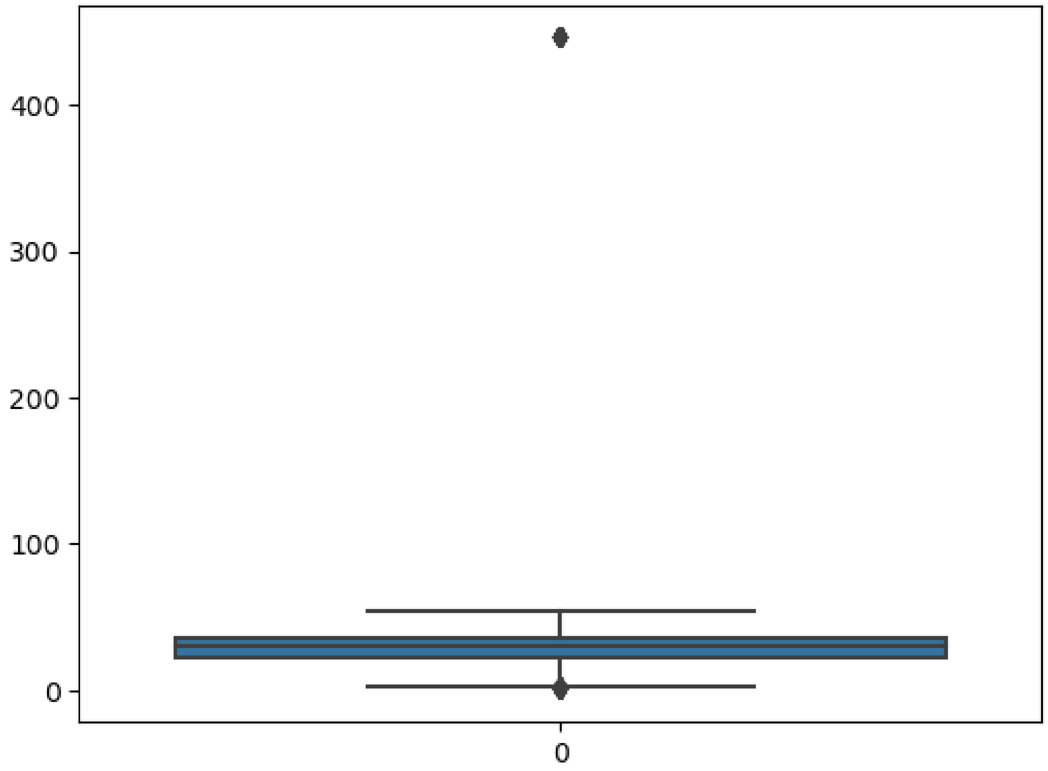
```
C:\Users\DELL\AppData\Local\Temp\ipykernel_19624\530051474.py:1:
FutureWarning: The default value of numeric_only in DataFrame.median
is deprecated. In a future version, it will default to False. In
addition, specifying 'numeric_only=None' is deprecated. Select only
valid columns or specify the value of numeric_only to silence this
warning.
  df.median()
```

```
PassengerId    446.0000
Survived         0.0000
Pclass           3.0000
Age             29.7000
SibSp            0.0000
Parch            0.0000
Fare            14.4542
dtype: float64
```

```
df['Age']=np.where(df['Age']>upper_limit,446.0000,df['Age'])
```

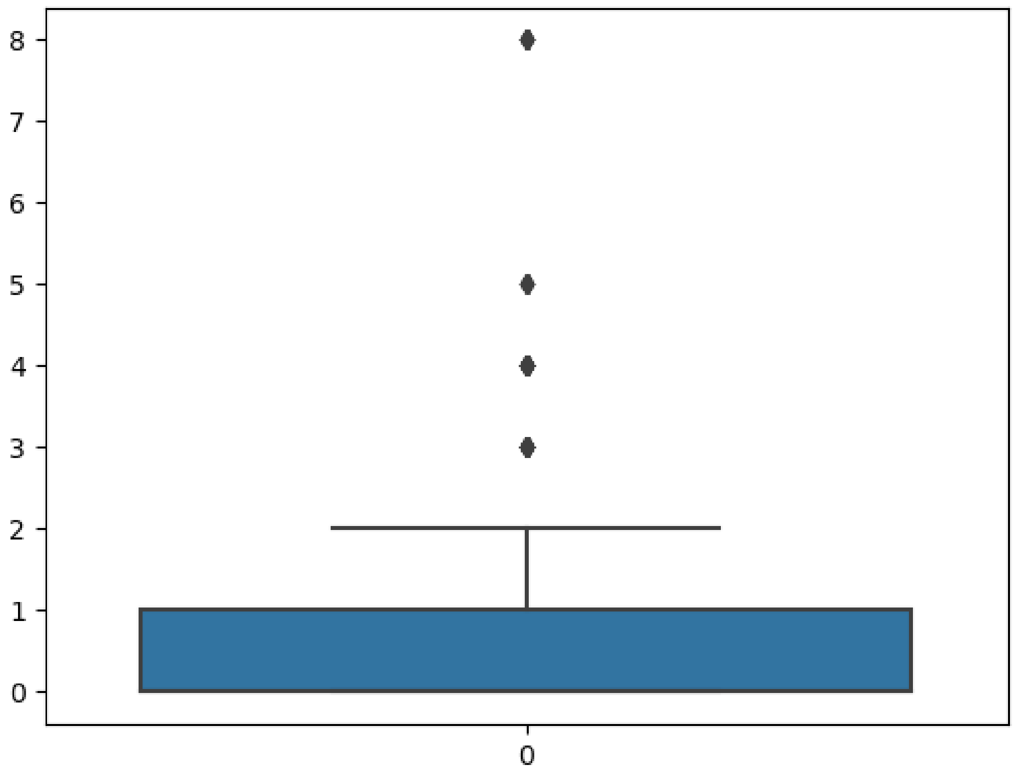
```
sns.boxplot(df.Age)
```

```
<Axes: >
```



```
sns.boxplot(df["SibSp"])
```

```
<Axes: >
```



```
# Outlier removal by replacement with median
q1=df.SibSp.quantile(0.25)
q3=df.SibSp.quantile(0.75)

q1

0.0

q3

1.0

IQR=q3-q1
IQR

1.0

upper_limit=q3+1.5*IQR
upper_limit

2.5

lower_limit=q1-1.5*IQR
lower_limit

-1.5

df.median()
```

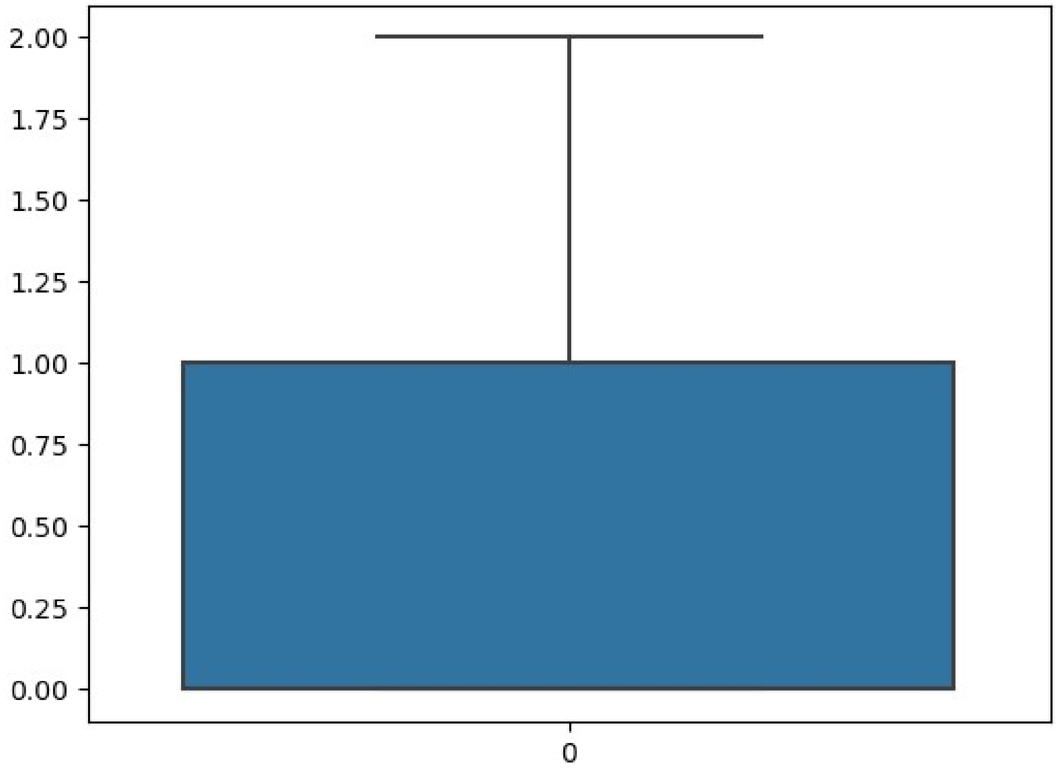
```
C:\Users\DELL\AppData\Local\Temp\ipykernel_19624\530051474.py:1:
FutureWarning: The default value of numeric_only in DataFrame.median
is deprecated. In a future version, it will default to False. In
addition, specifying 'numeric_only=None' is deprecated. Select only
valid columns or specify the value of numeric_only to silence this
warning.
  df.median()
```

```
PassengerId    446.0000
Survived         0.0000
Pclass           3.0000
Age             29.7000
SibSp            0.0000
Parch            0.0000
Fare            14.4542
dtype: float64
```

```
df['SibSp']=np.where(df['SibSp']>upper_limit,0.0000,df['SibSp'])
```

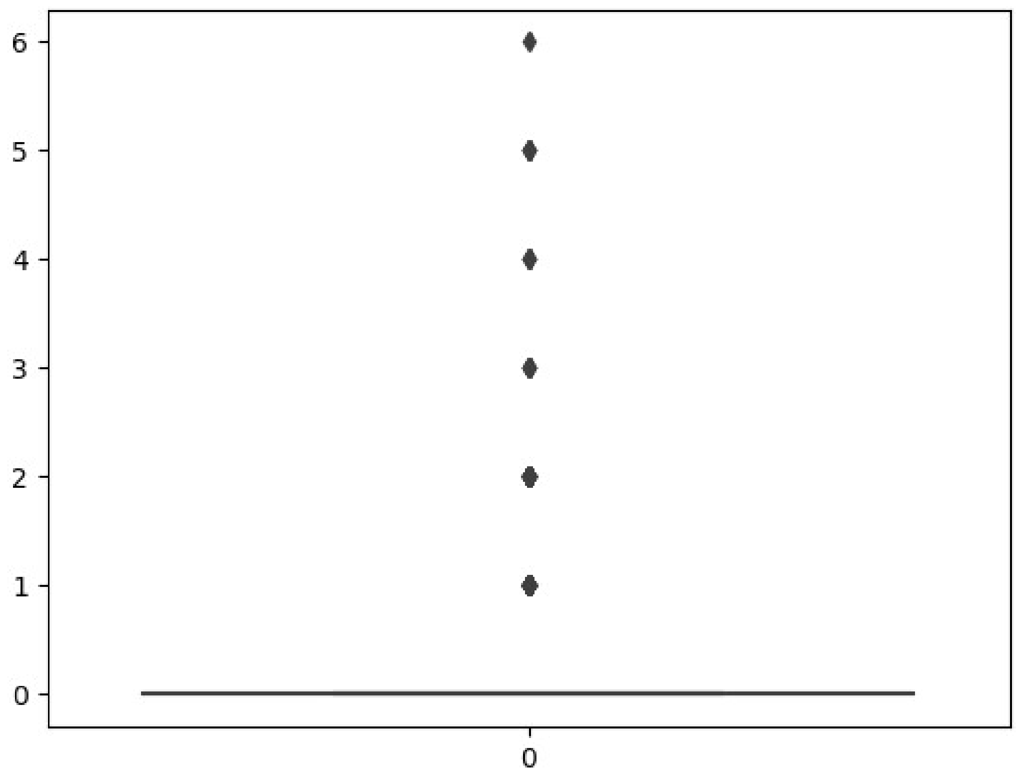
```
sns.boxplot(df["SibSp"])
```

```
<Axes: >
```



```
sns.boxplot(df["Parch"])
```

```
<Axes: >
```



```
# Outlier removal by replacement with median
q1=df.Parch.quantile(0.25)
q3=df.Parch.quantile(0.75)

q1

0.0

q3

0.0

IQR=q3-q1
IQR

0.0

upper_limit=q3+1.5*IQR
upper_limit

0.0

lower_limit=q1-1.5*IQR
lower_limit

0.0

df.median()
```

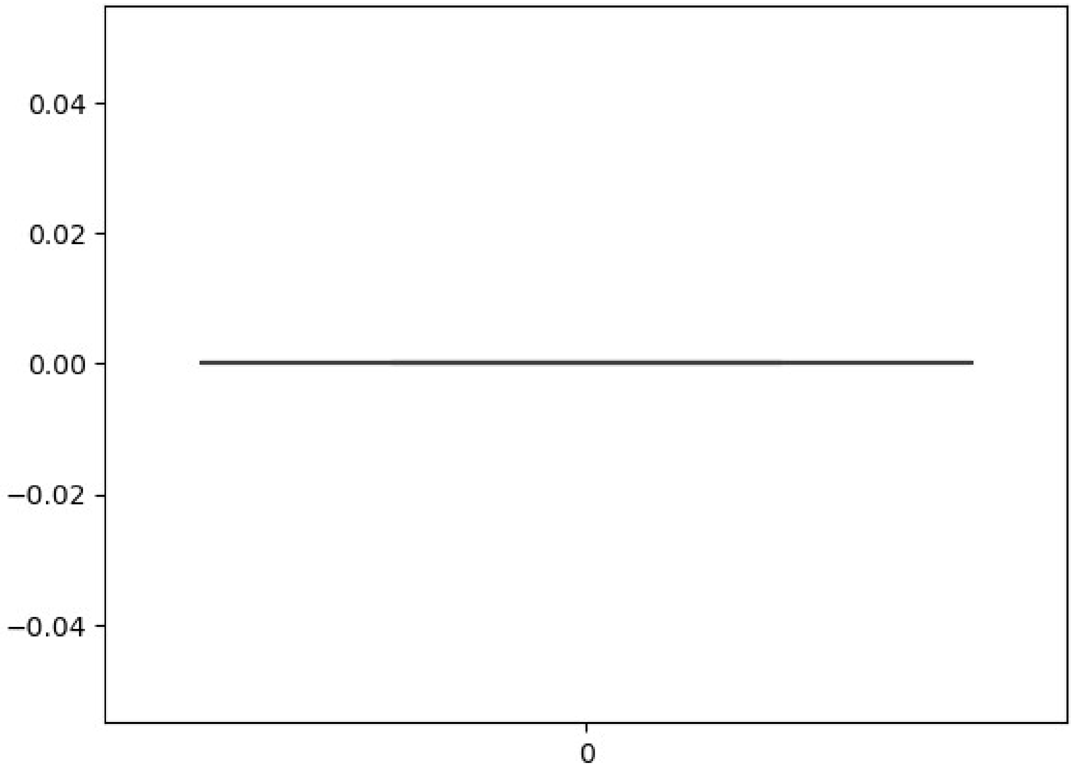
```
C:\Users\DELL\AppData\Local\Temp\ipykernel_19624\530051474.py:1:
FutureWarning: The default value of numeric_only in DataFrame.median
is deprecated. In a future version, it will default to False. In
addition, specifying 'numeric_only=None' is deprecated. Select only
valid columns or specify the value of numeric_only to silence this
warning.
  df.median()
```

```
PassengerId    446.0000
Survived         0.0000
Pclass           3.0000
Age             29.7000
SibSp            0.0000
Parch            0.0000
Fare            14.4542
dtype: float64
```

```
df['Parch']=np.where(df['Parch']>upper_limit,0.0000,df['Parch'])
```

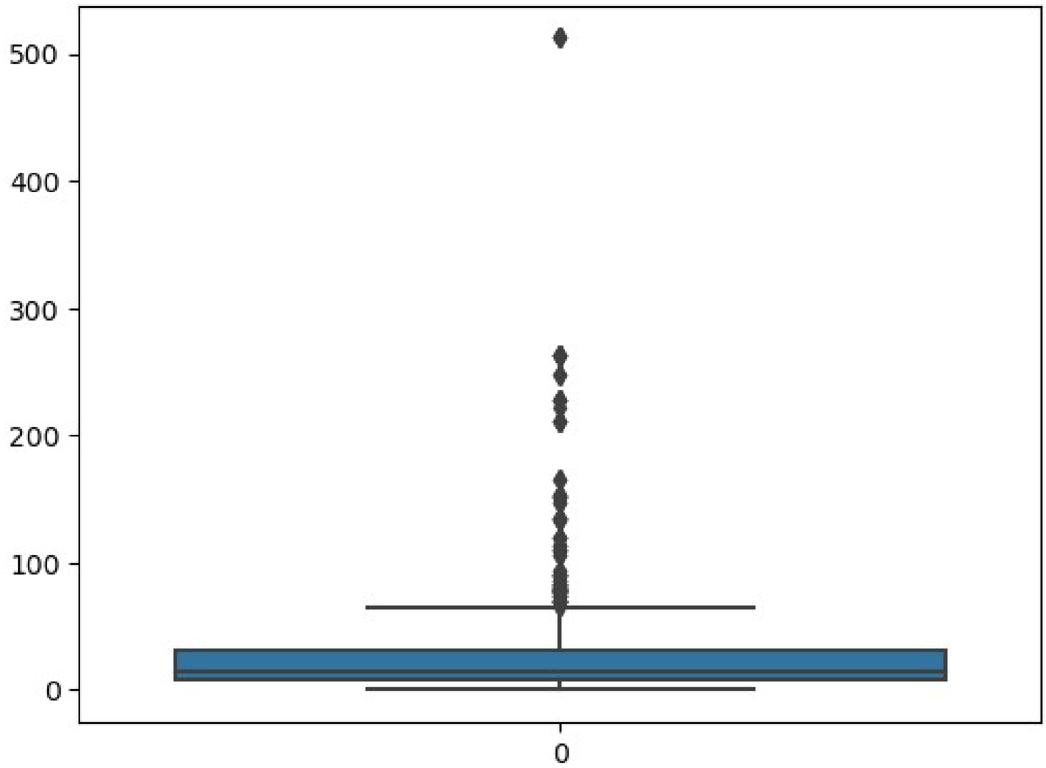
```
sns.boxplot(df["Parch"])
```

```
<Axes: >
```



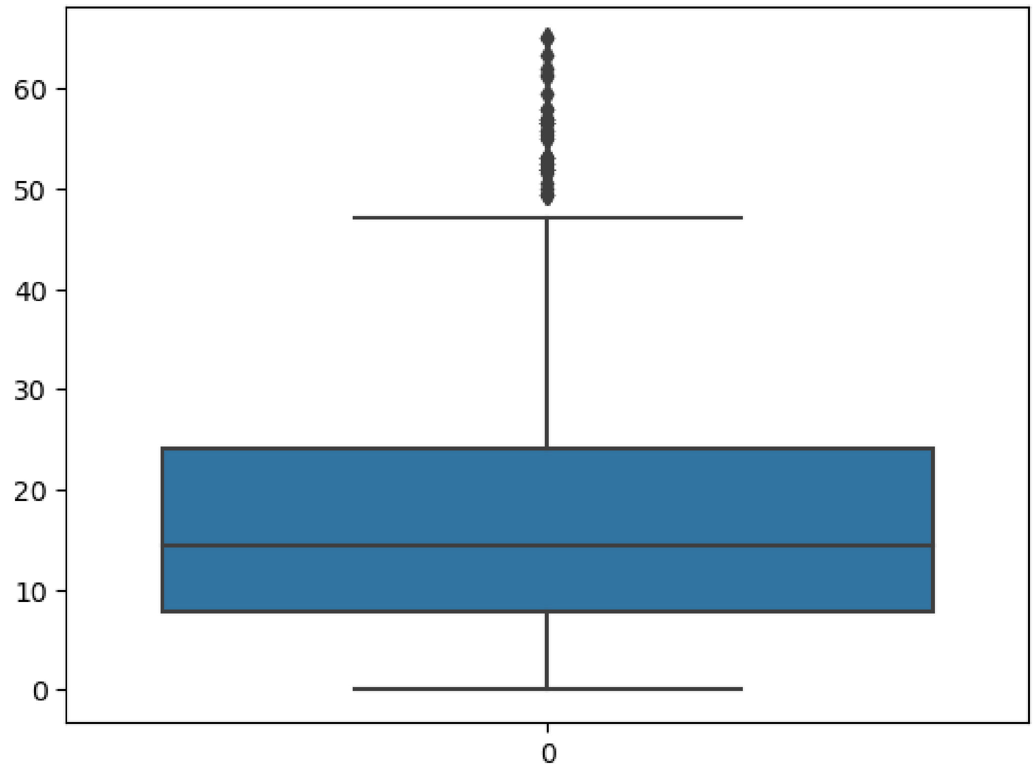
```
sns.boxplot(df["Fare"])
```

```
<Axes: >
```

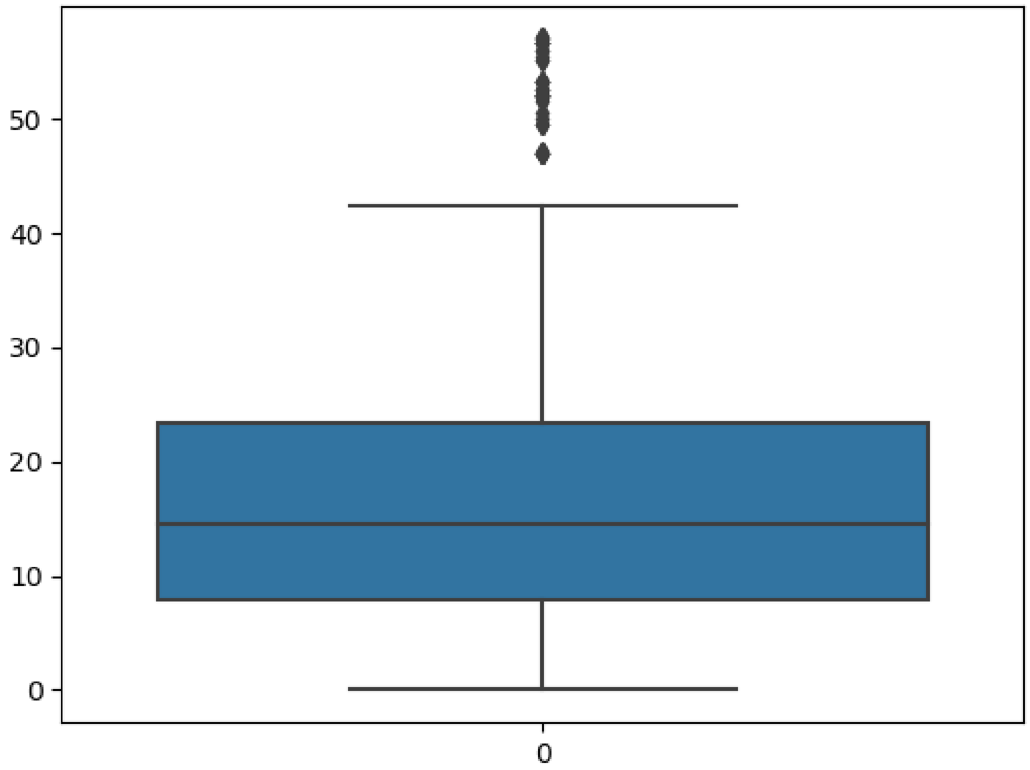


```
# Outlier removal by replacement with median
q1=df.Fare.quantile(0.25)
q3=df.Fare.quantile(0.75)

q1

7.9104

q3

31.0

IQR=q3-q1
IQR

23.0896

upper_limit=q3+1.5*IQR
upper_limit

65.6344

lower_limit=q1-1.5*IQR
lower_limit

-26.724

df['Fare']=np.where(df['Fare']>upper_limit,14.4542,df['Fare'])
```

```
# df['Fare']=np.where(df['Fare']>upper_limit,upper_limit,np.where(df['Fare']<lower_limit,lower_limit,df['Fare']))

sns.boxplot(df["Fare"])

<Axes: >
```



```
# Outlier removal by percentile method
p99 = df.Fare.quantile(0.99)
p99

57.09792000000002

df=df[df.Fare<=p99]

sns.boxplot(df.Fare)

<Axes: >
```



```
# Splitting dependent and independent variables
df.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
1.0
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0
1.0
2                             Heikkinen, Miss. Laina  female  26.0
0.0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0
1.0
4                           Allen, Mr. William Henry    male  35.0
0.0

   Parch            Ticket     Fare Embarked
0    0.0         A/5 21171   7.2500        S
1    0.0          PC 17599  14.4542        C
```

```
2    0.0  STON/O2. 3101282   7.9250        S
3    0.0            113803  53.1000        S
4    0.0            373450   8.0500        S
```

```
df=df.drop(['PassengerId','Name','Ticket','Embarked'],axis=1)
df
```

```
     Survived  Pclass     Sex   Age  SibSp  Parch     Fare
0           0       3    male  22.0    1.0    0.0   7.2500
1           1       1  female  38.0    1.0    0.0  14.4542
2           1       3  female  26.0    0.0    0.0   7.9250
3           1       1  female  35.0    1.0    0.0  53.1000
4           0       3    male  35.0    0.0    0.0   8.0500
..        ...     ...     ...   ...    ...    ...      ...
886         0       2    male  27.0    0.0    0.0  13.0000
887         1       1  female  19.0    0.0    0.0  30.0000
888         0       3  female  29.7    1.0    0.0  23.4500
889         1       1    male  26.0    0.0    0.0  30.0000
890         0       3    male  32.0    0.0    0.0   7.7500

[882 rows x 7 columns]
```

```
df.shape
```

```
(882, 7)
```

```
df.head()
```

```
   Survived  Pclass     Sex   Age  SibSp  Parch     Fare
0         0       3    male  22.0    1.0    0.0   7.2500
1         1       1  female  38.0    1.0    0.0  14.4542
2         1       3  female  26.0    0.0    0.0   7.9250
3         1       1  female  35.0    1.0    0.0  53.1000
4         0       3    male  35.0    0.0    0.0   8.0500
```

```
# Independent variables should be 2d array or dataframe
X=df.drop(columns=["Survived"],axis=1)
X.head()
```

```
   Pclass     Sex   Age  SibSp  Parch     Fare
0       3    male  22.0    1.0    0.0   7.2500
1       1  female  38.0    1.0    0.0  14.4542
2       3  female  26.0    0.0    0.0   7.9250
3       1  female  35.0    1.0    0.0  53.1000
4       3    male  35.0    0.0    0.0   8.0500
```

```
X.shape
```

```
(882, 6)
```

```
type(X)
```

```
pandas.core.frame.DataFrame
```

```
# Dependent variable should be 1d array or series
Y=df["Survived"]
Y.head()
```

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

```
# Encoding
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
X["Sex"]=le.fit_transform(X["Sex"])
X.head()
```

```
   Pclass  Sex   Age  SibSp  Parch     Fare
0       3    1  22.0    1.0    0.0   7.2500
1       1    0  38.0    1.0    0.0  14.4542
2       3    0  26.0    0.0    0.0   7.9250
3       1    0  35.0    1.0    0.0  53.1000
4       3    1  35.0    0.0    0.0   8.0500
```

```
print(le.classes_)
```

```
['female' 'male']
```

```
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

```
{'female': 0, 'male': 1}
```

```
# Feature scaling
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
```

```
X_Scaled=pd.DataFrame(ms.fit_transform(X),columns=X.columns)
```

```
X_Scaled.head()
```

```
   Pclass  Sex       Age  SibSp  Parch      Fare
0     1.0  1.0  0.048431    0.5    0.0  0.127193
1     0.0  0.0  0.084340    0.5    0.0  0.253582
2     1.0  0.0  0.057408    0.0    0.0  0.139035
3     0.0  0.0  0.077607    0.5    0.0  0.931579
4     1.0  1.0  0.077607    0.0    0.0  0.141228
```

```python
# Splitting Data into Train and Test
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X_Scaled,Y,test_size=0.2,random_state=0)

print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(705, 6) (177, 6) (705,) (177,)
```