assignment-15sept

September 21, 2023

1 Assignment 15 sep

2 Perform Data preprocessing on Titanic dataset Data Collection. Please download the dataset from https://www.kaggle.com/datasets/yasserh/titanic-dataset Data Preprocessing o Import the Libraries. o Importing the dataset. o Checking for Null Values. o Data Visualization. o Outlier Detection o Splitting Dependent and Independent variables o Perform Encoding o Feature Scaling. o Splitting Data into Train and Test

3 1. Import the libraries

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

4 2. Importing the dataset

[14]:
```
df=pd.read_csv("Titanic-Dataset.csv")
```

[15]:
```
df.head()
```

[15]:
```
   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
```

[16]:
```
df.describe()
```

[16]:
```
       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
```

[17]:
```
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
```

[18]:
```
df.corr()
```

```
<ipython-input-18-2f6f6606aa2c>: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()
```

[18]:
```
             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
```

[19]:
```
df.corr().Fare.sort_values(ascending=False)
```

```
<ipython-input-19-f51f352aac84>: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)
```

[19]:
```
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
```

5 3. Checking the null values

[20]:
```
df.isnull().any()
```

[20]:
```
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
```

[21]:
```
df.isnull().sum()
```

[21]:
```
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
```

[22]:
```
df.Parch.nunique()
```

[22]:
```
7
```

[23]:
```
df.Parch.unique()
```

[23]:
```
array([0, 1, 2, 5, 3, 4, 6])
```

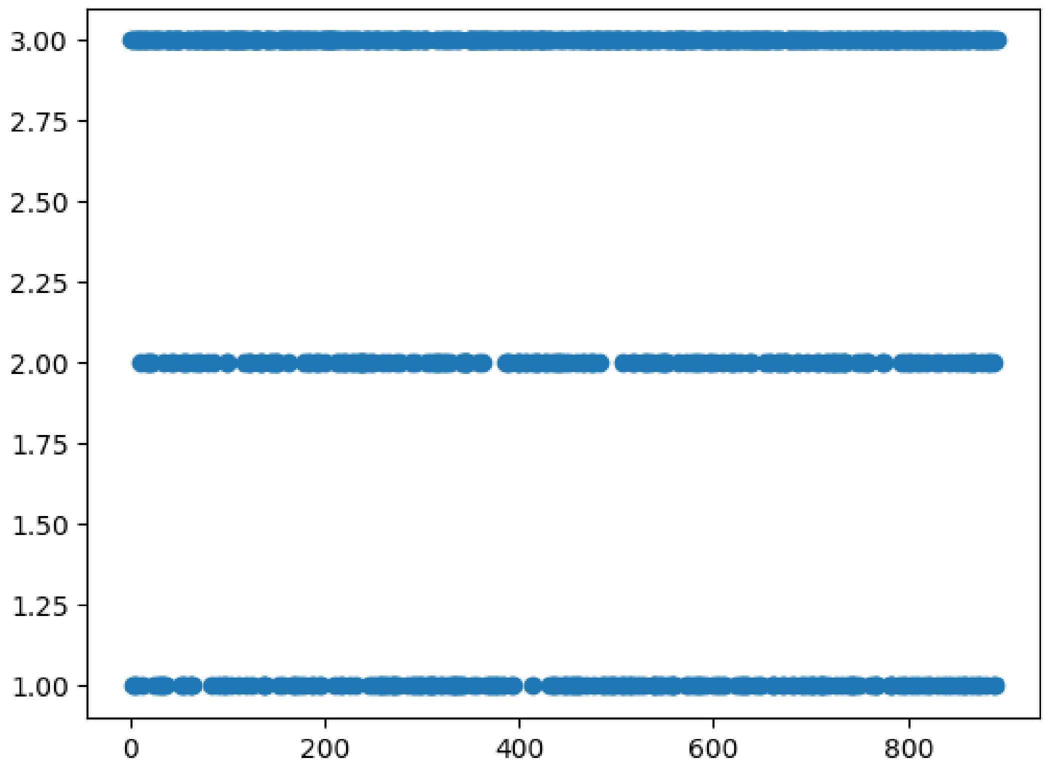
[24]:
```
df.Parch.value_counts()
```

```
[24]: 0    678
      1    118
      2     80
      5      5
      3      5
      4      4
      6      1
      Name: Parch, dtype: int64
```

6 4. Data Visualization

```
[25]: plt.scatter(df["PassengerId"],df["Pclass"])
```

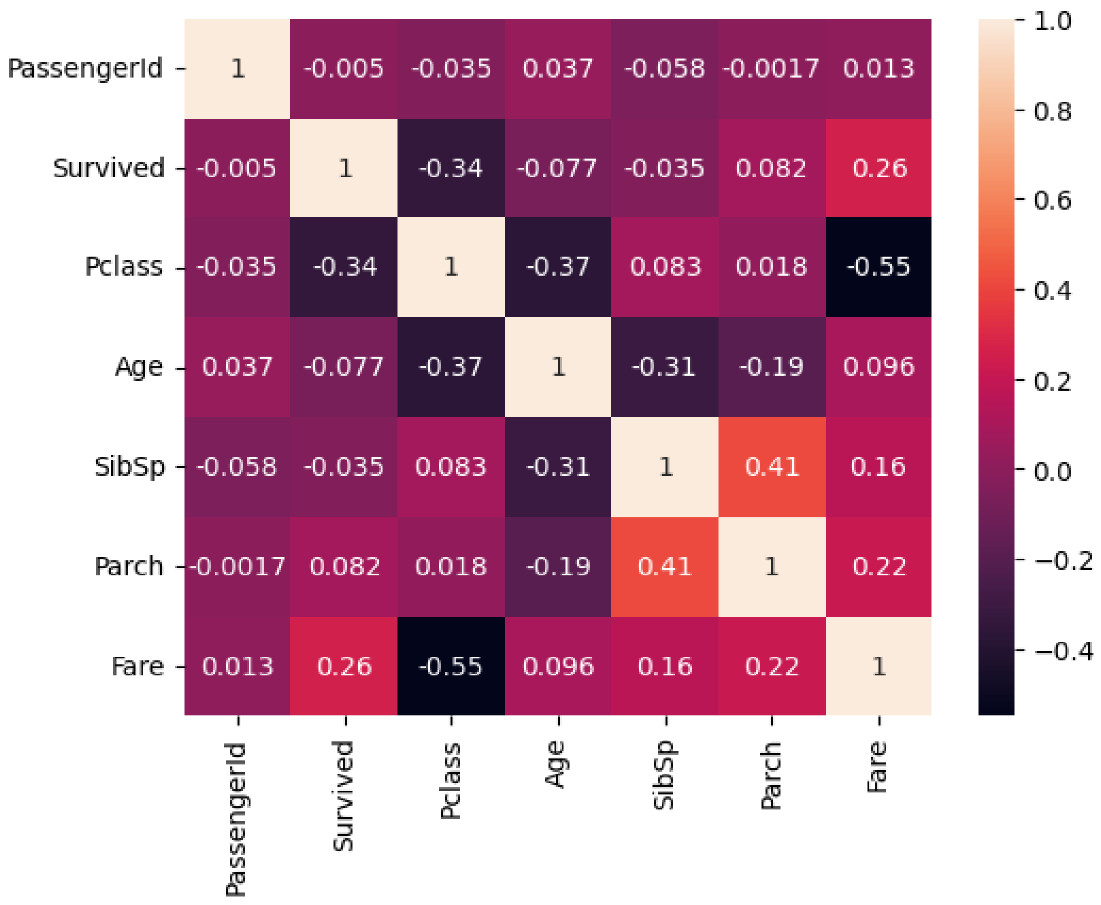
```
[25]: <matplotlib.collections.PathCollection at 0x7ca4c583b5e0>
```



```
[26]: sns.heatmap(df.corr(),annot=True)
```

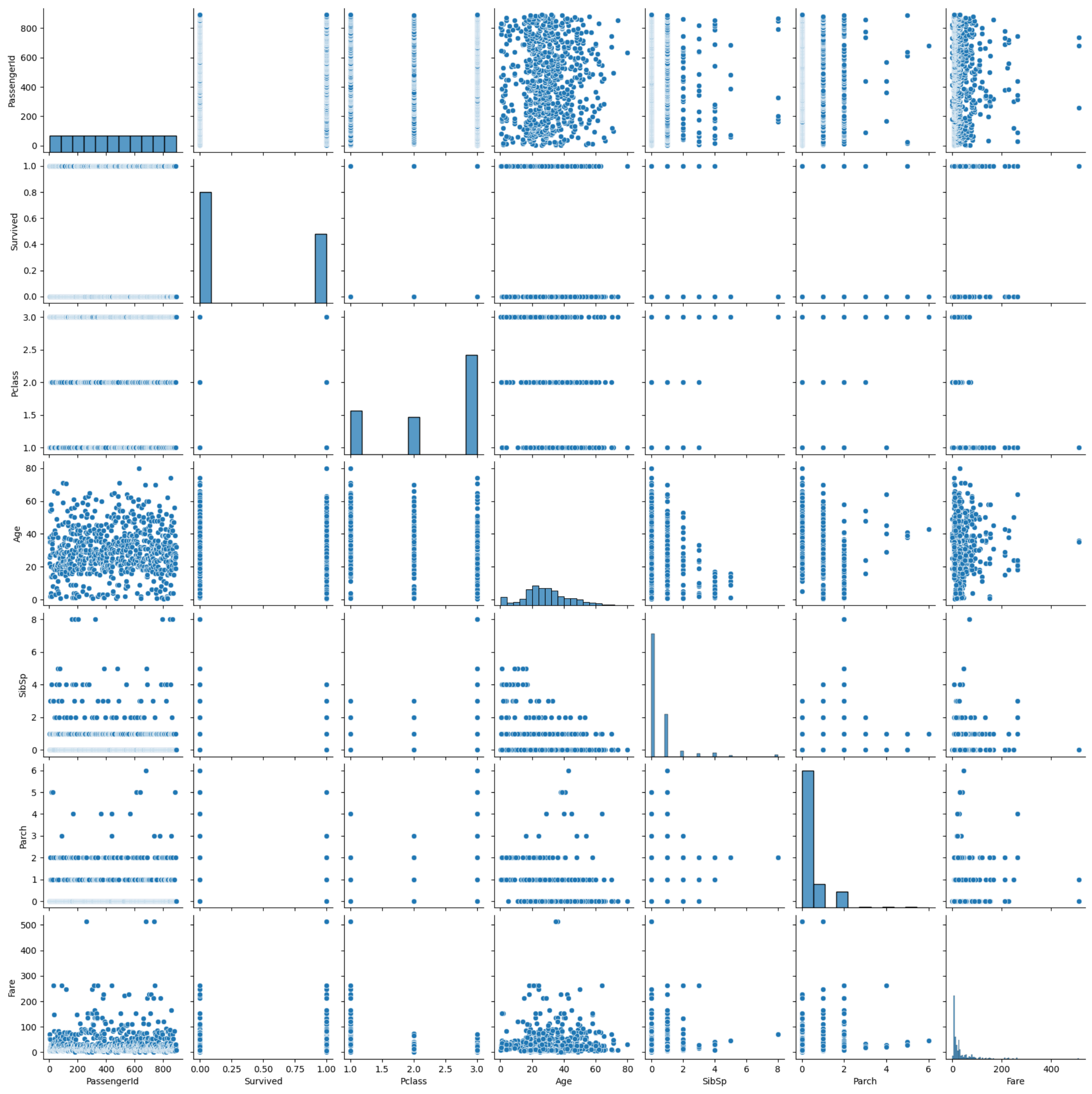
```
<ipython-input-26-8df7bcac526d>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  sns.heatmap(df.corr(),annot=True)
```

[26]: <Axes: >



[27]:
```
sns.pairplot(df)
```

[27]:
```
<seaborn.axisgrid.PairGrid at 0x7ca4c26f1150>
```



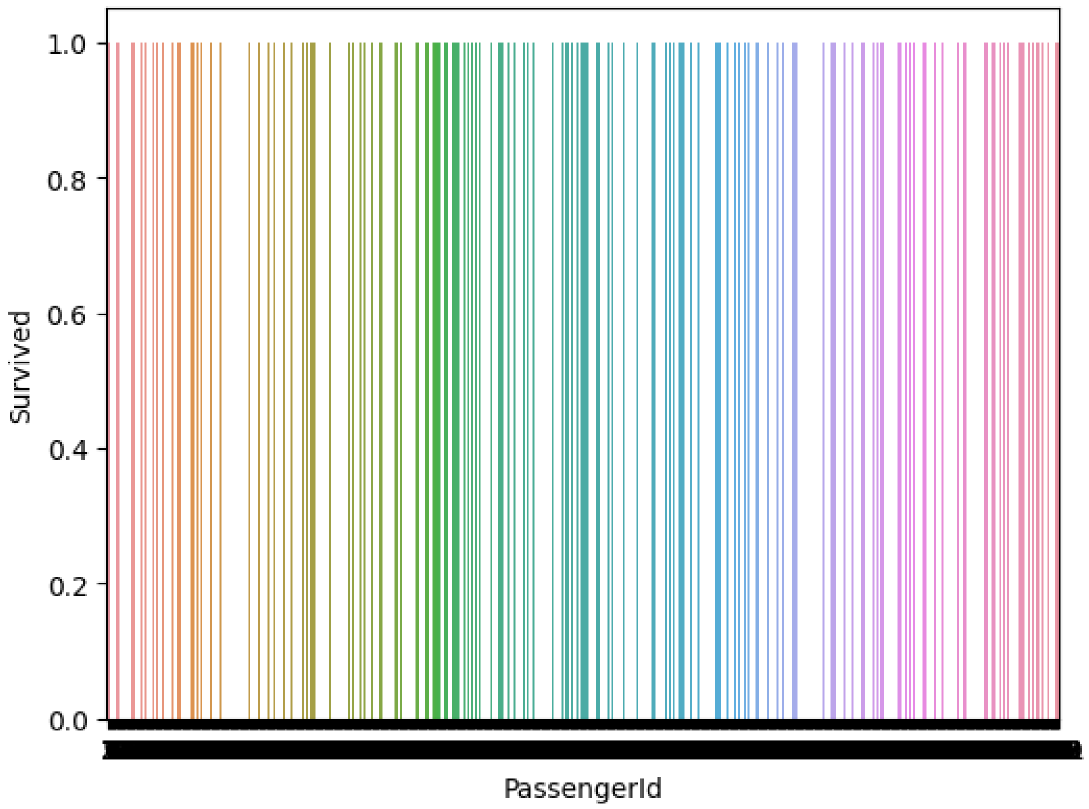
[28]:
```
sns.barplot(x=df["PassengerId"],y=df["Survived"],ci=0)
```

```
<ipython-input-28-0a8bf95a1f9c>:1: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=('ci', 0)` for the same effect.

  sns.barplot(x=df["PassengerId"],y=df["Survived"],ci=0)
```

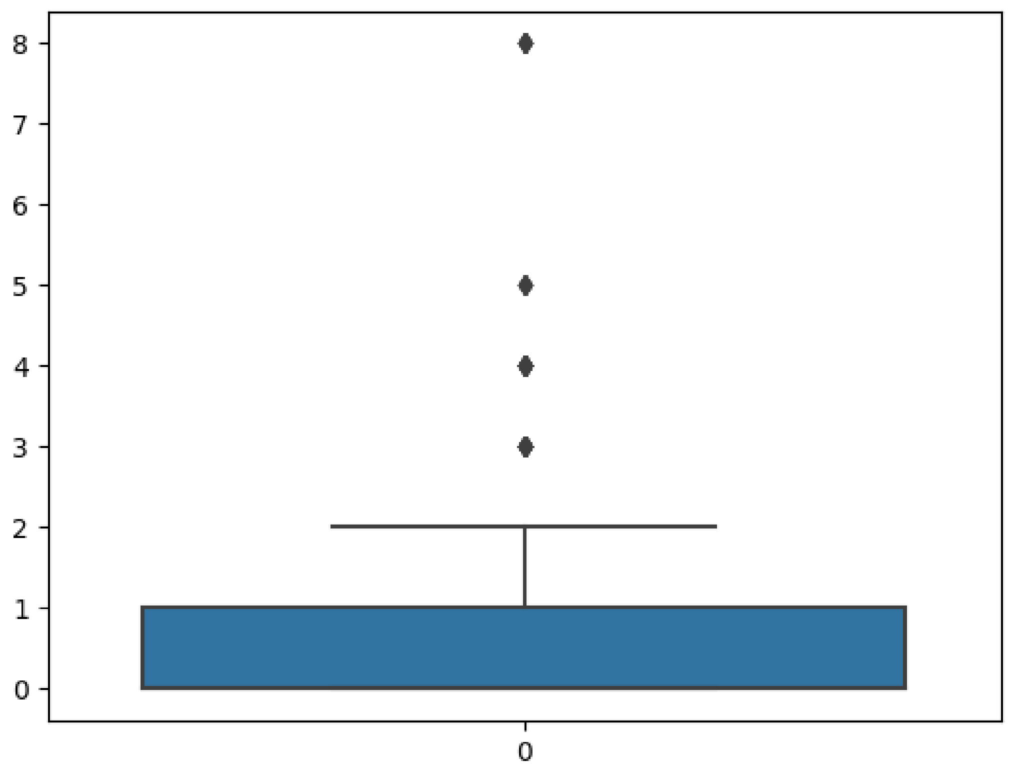
[28]:
```
<Axes: xlabel='PassengerId', ylabel='Survived'>
```



7 5. Outlier Detection

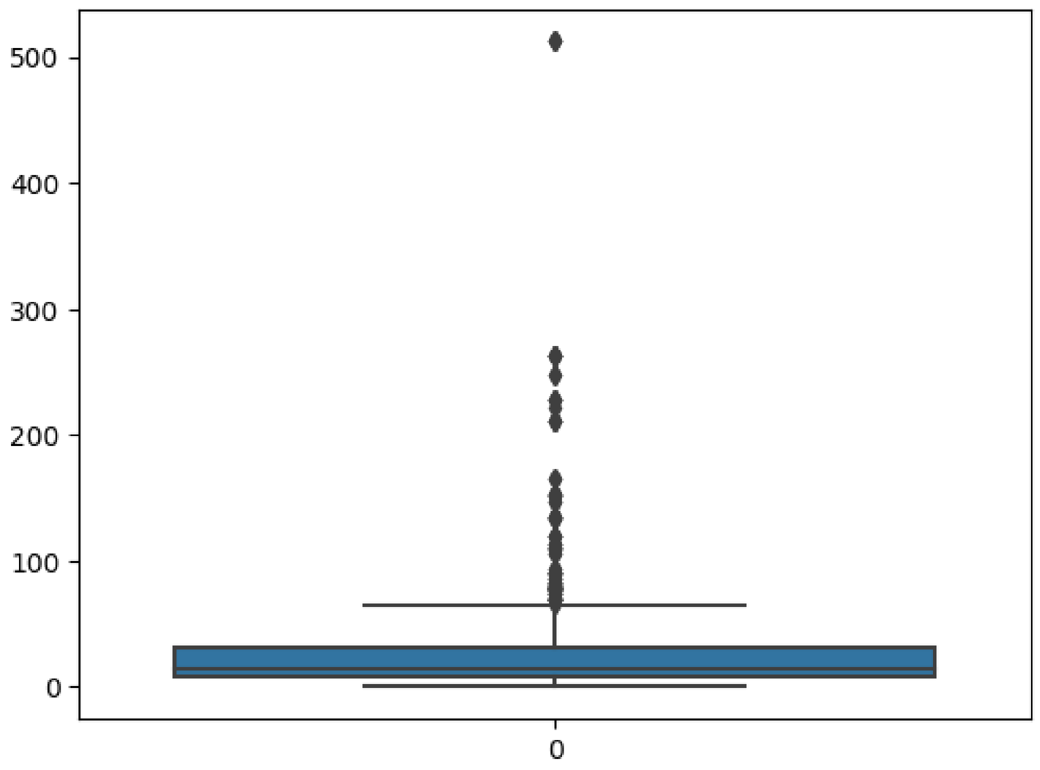
```
[29]: sns.boxplot(df["SibSp"])
```

```
[29]: <Axes: >
```



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

```
[30]: <Axes: >
```



8 6. Splitting Dependent and Independent Variables

[31]:
```
df.head()
```

[31]:
```
   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
```

[32]:
```
X=df.drop(columns=["SibSp"],axis=1)
X.head()
```

[32]:
```
   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  Parch  \
0                            Braund, Mr. Owen Harris    male  22.0      0
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      0
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      0
4                           Allen, Mr. William Henry    male  35.0      0

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

[33]:
```
X.shape
```

[33]:
```
(891, 11)
```

[34]:
```
type(X)
```

[34]:
```
pandas.core.frame.DataFrame
```

[35]:
```
y=df["SibSp"]
```

[36]:
```
y.head()
```

[36]:
```
0    1
1    1
2    0
3    1
4    0
Name: SibSp, dtype: int64
```

9 7. Encoding

[37]:
```
X.head()
```

[37]:
```
   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  Parch  \
0                            Braund, Mr. Owen Harris    male  22.0      0
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      0
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      0
4                           Allen, Mr. William Henry    male  35.0      0

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

[38]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
X["Name"]=le.fit_transform(X["Name"])
X["Sex"]=le.fit_transform(X["Sex"])
X["Ticket"]=le.fit_transform(X["Ticket"])
X["Cabin"]=le.fit_transform(X["Cabin"])
X["Embarked"]=le.fit_transform(X["Embarked"])
X.head()
```

[38]:
```
   PassengerId  Survived  Pclass  Name  Sex   Age  Parch  Ticket     Fare  \
0            1         0       3   108    1  22.0      0     523   7.2500
1            2         1       1   190    0  38.0      0     596  71.2833
2            3         1       3   353    0  26.0      0     669   7.9250
3            4         1       1   272    0  35.0      0      49  53.1000
4            5         0       3    15    1  35.0      0     472   8.0500

   Cabin  Embarked
0    147         2
1     81         0
2    147         2
3     55         2
4    147         2
```

[39]:
```
print(le.classes_)
```

```
['C' 'Q' 'S' nan]
```

[40]:
```
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

[40]:
```
{'C': 0, 'Q': 1, 'S': 2, nan: 3}
```

10 8. Feature Scaling

[41]:
```
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
X_Scaled=pd.DataFrame(ms.fit_transform(X),columns=X.columns)
X_Scaled.head()
```

[41]:
```
   PassengerId  Survived  Pclass      Name  Sex       Age  Parch    Ticket  \
0     0.000000       0.0     1.0  0.121348  1.0  0.271174    0.0  0.769118
1     0.001124       1.0     0.0  0.213483  0.0  0.472229    0.0  0.876471
2     0.002247       1.0     1.0  0.396629  0.0  0.321438    0.0  0.983824
3     0.003371       1.0     0.0  0.305618  0.0  0.434531    0.0  0.072059
4     0.004494       0.0     1.0  0.016854  1.0  0.434531    0.0  0.694118

       Fare    Cabin  Embarked
0  0.014151  1.00000  0.666667
1  0.139136  0.55102  0.000000
2  0.015469  1.00000  0.666667
3  0.103644  0.37415  0.666667
4  0.015713  1.00000  0.666667
```

11 9. Train Test Split

[42]:
```
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_test.shape,y_test.shape)
```

```
(712, 11) (179, 11) (179,) (179,)
```