1.Import the Libraries

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

2.Import the Dataset

```
In [2]: df = pd.read_csv("Titanic-Dataset.csv")
```

```
In [3]: df.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
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May 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

```
In [4]: df.describe()
```

Out[4]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
In [5]: 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
```

```
In [6]: df.corr()
```

```
  df.corr()
```

Out[6]:

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

```
In [7]: df.corr().Fare.sort_values(ascending=False)
```

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

```
Out[7]: 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
```

3.Checking for Null Values

```
In [8]: df.isnull().any()
```

```
Out[8]: 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
```

```
In [9]: df.isnull().sum()
```

```
Out[9]: 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
```

```
In [12]: df.Pclass.nunique()
```

```
Out[12]: 3
```

```
In [13]: df.Pclass.unique()
```

```
Out[13]: array([3, 1, 2], dtype=int64)
```

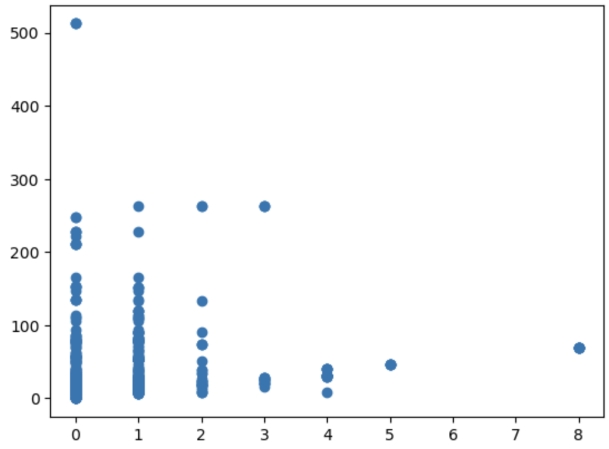
```
In [14]: df.Pclass.value_counts()
```

```
Out[14]: 3    491
         1    216
         2    184
         Name: Pclass, dtype: int64
```

4.Data Visualization

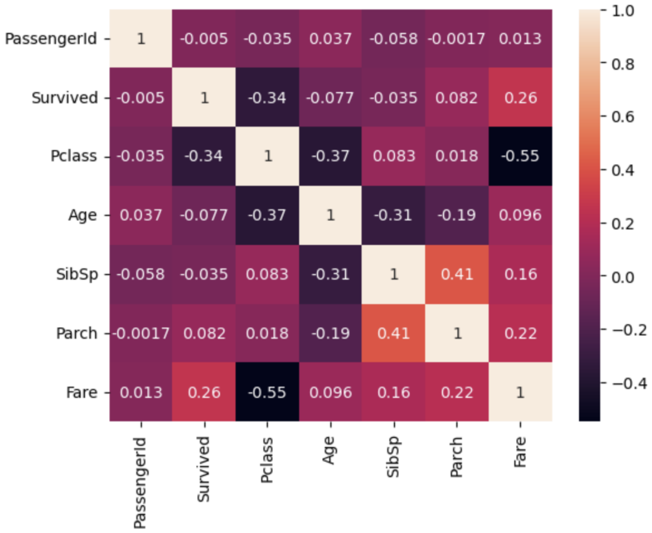
```
In [15]: plt.scatter(df["SibSp"],df["Fare"])
```

```
Out[15]: <matplotlib.collections.PathCollection at 0x1c701a94190>
```



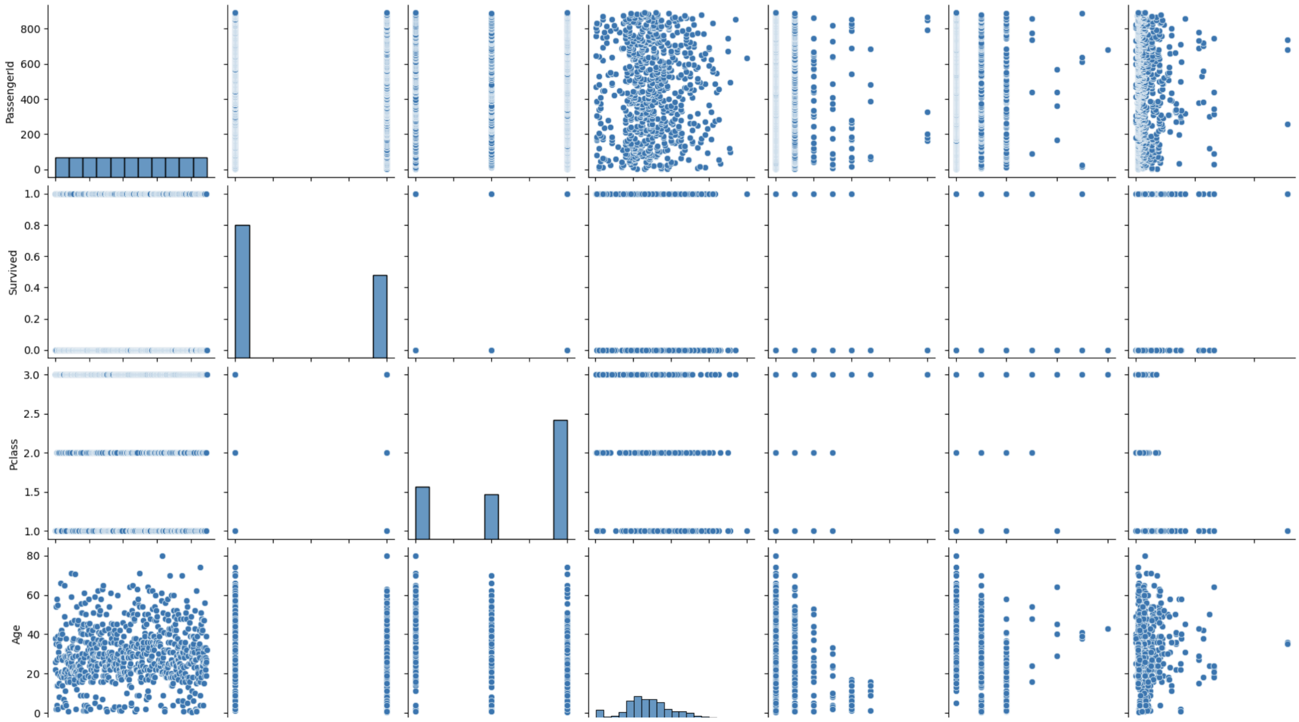
In [16]:
```
sns.heatmap(df.corr(numeric_only = True), annot = True)
```

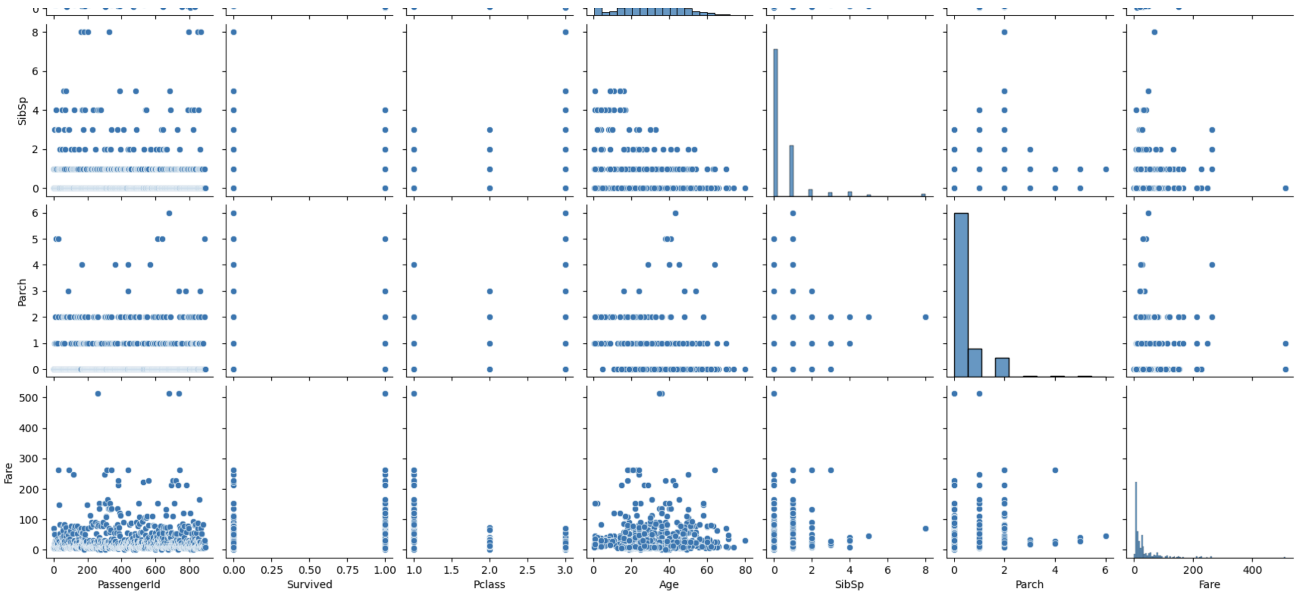
Out[16]:
```
<Axes: >
```



```
In [17]: sns.pairplot(df)
```

```
Out[17]: <seaborn.axisgrid.PairGrid at 0x1c702c9d6d0>
```




SibSp
Parch
Fare
0
2
4
6
8
0
1
2
3
4
5
6
0
100
200
300
400
500
0
200
400
600
800
PassengerId
0.00
0.25
0.50
0.75
1.00
Survived
1.0
1.5
2.0
2.5
3.0
Pclass
0
20
40
60
80
Age
0
2
4
6
8
SibSp
0
2
4
6
Parch
0
200
400
Fare

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare

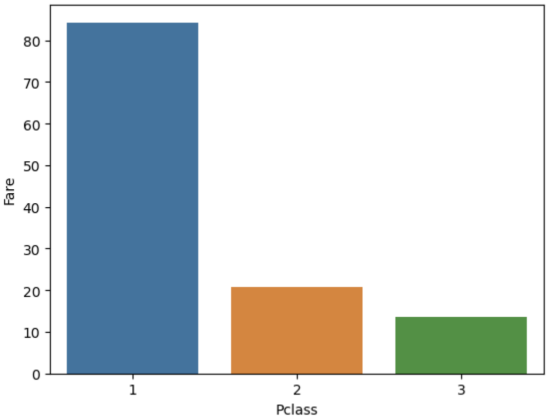
In [18]:
```
sns.barplot(x = df["Pclass"] , y = df["Fare"] , ci = 0)
```

```
C:\Users\alwin\AppData\Local\Temp\ipykernel_13932\1349153731.py:1: FutureWarning:

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

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

Out[18]:
```
<Axes: xlabel='Pclass', ylabel='Fare'>
```



5.Outlier Detection

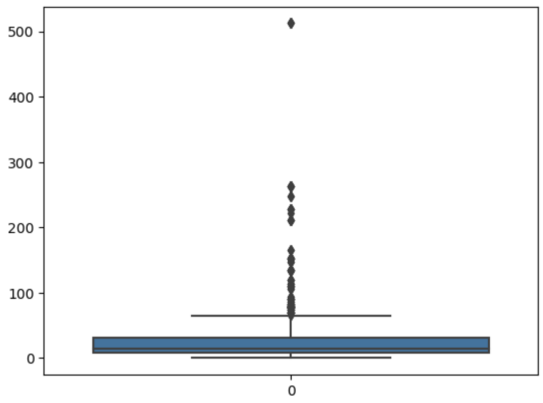
In [19]:
```
df.head()
```

Out[19]:

	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
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May 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

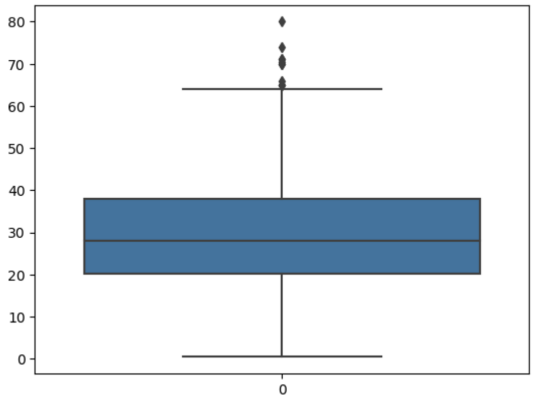
In [20]:
```
sns.boxplot(df["Fare"])
```

Out[20]:
```
<Axes: >
```



```
In [21]: sns.boxplot(df["Age"])
```

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



6.Splitting Dependent and Independent Variables

```
In [58]: df.head()
```

Out[58]:

	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
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May 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

In [70]:
```
#Independent Variables should be 2D Array or dataframe
x = df.drop(columns = ["Fare" , "Name" , "Ticket"] , axis = 1)
x.head()
```

Out[70]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Cabin	Embarked
0	1	0	3	male	22.0	1	0	NaN	S
1	2	1	1	female	38.0	1	0	C85	C
2	3	1	3	female	26.0	0	0	NaN	S
3	4	1	1	female	35.0	1	0	C123	S
4	5	0	3	male	35.0	0	0	NaN	S

In [71]:
```
x.shape
```

Out[71]:
```
(891, 9)
```

In [72]:
```
type(x)
```

Out[72]:
```
pandas.core.frame.DataFrame
```

In [73]:
```
y = df["Fare"]
y.head()
```

Out[73]:
```
0     7.2500
1    71.2833
2     7.9250
3    53.1000
4     8.0500
Name: Fare, dtype: float64
```

7.Encoding

In [74]:
```
x.head()
```

Out[74]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Cabin	Embarked
0	1	0	3	male	22.0	1	0	NaN	S
1	2	1	1	female	38.0	1	0	C85	C
2	3	1	3	female	26.0	0	0	NaN	S
3	4	1	1	female	35.0	1	0	C123	S
4	5	0	3	male	35.0	0	0	NaN	S

In [75]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [76]:
```
x["Sex"] = le.fit_transform(x["Sex"])
```

In [77]:
```
x["Cabin"] = le.fit_transform(x["Cabin"])
```

In [83]:
```
x["Embarked"] = le.fit_transform(x["Embarked"])
```

In [84]:
```
x.head()
```

Out[84]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Cabin	Embarked
0	1	0	3	1	22.0	1	0	147	2
1	2	1	1	0	38.0	1	0	81	0
2	3	1	3	0	26.0	0	0	147	2
3	4	1	1	0	35.0	1	0	55	2
4	5	0	3	1	35.0	0	0	147	2

In [85]:
```
print(le.classes_)
```

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

In [86]:
```
mapping = dict(zip(le.classes_ , range(len(le.classes_))))
mapping
```

Out[86]:
```
{'C': 0, 'Q': 1, 'S': 2, nan: 3}
```

8.Feature Scaling

In [87]:
```python
from sklearn.preprocessing import MinMaxScaler
ms = MinMaxScaler()
```

In [88]:
```python
x_Scaled = ms.fit_transform(x)
```

In [89]:
```python
x_Scaled = pd.DataFrame(ms.fit_transform(x) , columns = x.columns)
```

In [90]:
```python
x_Scaled.head()
```

Out[90]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Cabin	Embarked
0	0.000000	0.0	1.0	1.0	0.271174	0.125	0.0	1.00000	0.666667
1	0.001124	1.0	0.0	0.0	0.472229	0.125	0.0	0.55102	0.000000
2	0.002247	1.0	1.0	0.0	0.321438	0.000	0.0	1.00000	0.666667
3	0.003371	1.0	0.0	0.0	0.434531	0.125	0.0	0.37415	0.666667
4	0.004494	0.0	1.0	1.0	0.434531	0.000	0.0	1.00000	0.666667

9.Splitting Data into Train and Test

In [91]:
```python
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)
```

In [92]:
```python
print(x_train.shape, x_test.shape ,y_train.shape , y_test.shape)
```

```
(712, 9) (179, 9) (712,) (179,)
```

In [ ]: