Name: Shubh Udaybhai Pandya
Reg. No.: 21BCE5770
Morning Batch(VIT Chennai)

Assignment 3: Data Preprocessing

1.Import the Libraries

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

2.Importing the dataset.

In [2]:
```
dataset = pd.read_csv("Titanic-Dataset.csv")
dataset.head(5)
```

Out[2]:

	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 [3]:
```
dataset.shape
```

Out[3]:
```
(891, 12)
```

3.Checking for Null Values.

In [4]:
```
dataset.isnull().any()
```

Out[4]:
```
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
```

Name: Shubh Udaybhai Pandya
Reg. No.: 21BCE5770
Morning Batch(VIT Chennai)

In [5]:
```
dataset.isnull().sum()
```

Out[5]:
```
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 [6]:
```
#filling the null values of age with its median value
dataset['Age'].fillna(dataset['Age'].median(), inplace=True)
```

In [7]:
```
#replacing the null rows with column embarked with its mode
dataset['Embarked']=dataset['Embarked'].fillna(dataset['Embarked'].mode()[0])
```

In [8]:
```
#as cabin has many null values ,but since it is of no use, we will drop it when splitting data
```

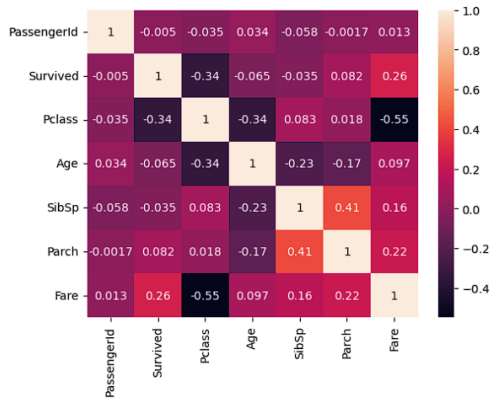
In [9]:
```
dataset.isnull().sum()
```

Out[9]:
```
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
```

4.Data Visualization.

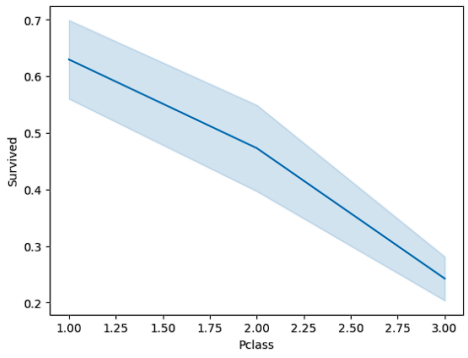
In [10]:
```
sns.heatmap(dataset.corr(),annot=True)
```

Out[10]:
```
<Axes: >
```



In [11]:
```
#It means that as the Plclass ticket is increasing, the survival rate is decreasing.
sns.lineplot(x="Pclass",y="Survived",data=dataset)
```

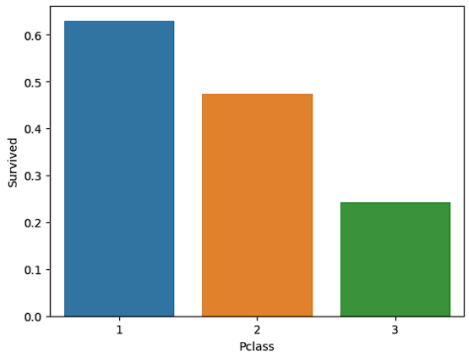
Out[11]:
```
<Axes: xlabel='Pclass', ylabel='Survived'>
```



Name: Shubh Udaybhai Pandya
Reg. No.: 21BCE5770
Morning Batch(VIT Chennai)

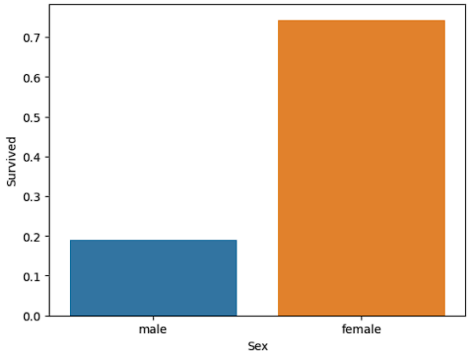
In [12]:
```
sns.barplot(y=dataset['Survived'],x=dataset['Pclass'],ci=0)
```

Out[12]:
```
<Axes: xlabel='Pclass', ylabel='Survived'>
```



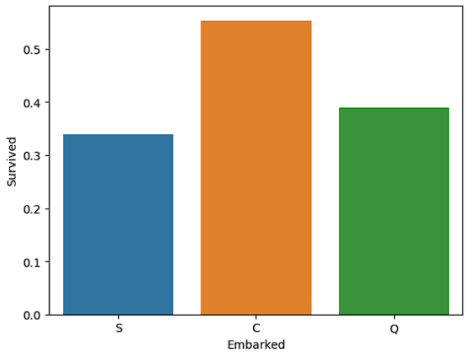
In [13]:
```
#mostly female have survived.
sns.barplot(y=dataset['Survived'],x=dataset['Sex'],ci=0)
```

Out[13]:
```
<Axes: xlabel='Sex', ylabel='Survived'>
```



In [14]:
```
#Mostly passengers with embarked C have mostly survived.
sns.barplot(y=dataset['Survived'],x=dataset['Embarked'],ci=0)
```

Out[14]:
```
<Axes: xlabel='Embarked', ylabel='Survived'>
```

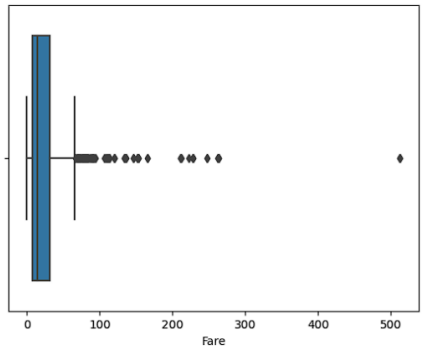


Name: Shubh Udaybhai Pandya
Reg. No.: 21BCE5770
Morning Batch(VIT Chennai)

5.Outlier Detection

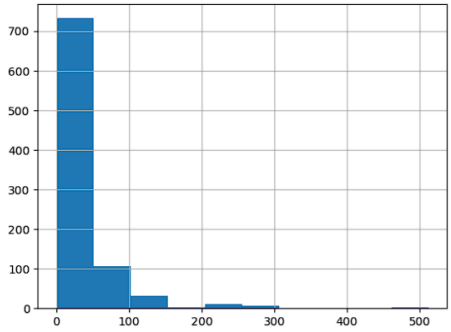
In [15]:
```
sns.boxplot(x=dataset["Fare"])
```

Out[15]:
```
<Axes: xlabel='Fare'>
```



In [16]:
```
dataset['Fare'].hist()
```

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



In [17]:
```
print('skewness value of Age: ',dataset['Age'].skew())
print('skewness value of Fare: ',dataset['Fare'].skew())
```

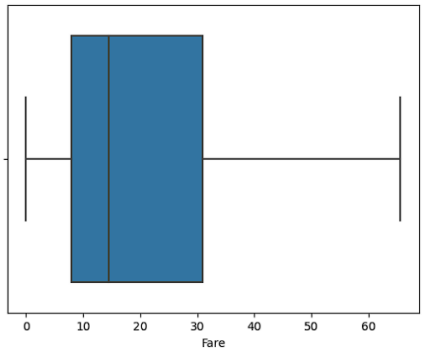
```
skewness value of Age:  0.5102446555756495
skewness value of Fare:  4.787316519674893
```

Inference : It shows that fare has outliers that are left skewed.

In [18]:
```
#So we will use flooring and capping for removing outliers
Q1 = dataset['Fare'].quantile(0.25)
Q3 = dataset['Fare'].quantile(0.75)
IQR = Q3 - Q1
whisker_width = 1.5
lower_whisker = Q1 -(whisker_width*IQR)
upper_whisker = Q3 + (whisker_width*IQR)
dataset['Fare']=np.where(dataset['Fare']>upper_whisker,upper_whisker,np.where(dataset['Fare']<lower_whisker,lower_whisker,dataset['Fare']))
```

In [19]:
```
sns.boxplot(x=dataset["Fare"])
```

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



Inference: Hence We have successfully removed outliers.

Name: Shubh Udaybhai Pandya
Reg. No.: 21BCE5770
Morning Batch(VIT Chennai)

6.Splitting Dependent and Independent variables

In [20]:
```
#dropping unecessary columns
dataset.drop(['PassengerId','Name','Ticket','Cabin'],axis=1,inplace=True)
```

In [21]:
```
x=dataset.drop(columns=['Survived'])
y=dataset.iloc[:,0:1]
```

In [22]:
```
y.shape
```

Out[22]:
```
(891, 1)
```

7.Perform Encoding

In [23]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()

x["Sex"]=le.fit_transform(x["Sex"])
sex_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("Mapping for 'Sex' column:", sex_mapping)
```

```
Mapping for 'Sex' column: {'female': 0, 'male': 1}
```

In [24]:
```
x["Embarked"]=le.fit_transform(x["Embarked"])

embarked_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("Mapping for 'Embarked' column:", embarked_mapping)
```

```
Mapping for 'Embarked' column: {'C': 0, 'Q': 1, 'S': 2}
```

8.Feature Scaling

In [25]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [26]:
```
x=sc.fit_transform(x)
```

In [27]:
```
x
```

Out[27]:
```
array([[ 0.82737724,  0.73769513, -0.56573646, ..., -0.47367361,
        -0.82055245,  0.58595414],
       [-1.56610693, -1.35557354,  0.66386103, ..., -0.47367361,
         2.03162322, -1.9423032 ],
       [ 0.82737724, -1.35557354, -0.25833709, ..., -0.47367361,
        -0.78757757,  0.58595414],
       ...,
       [ 0.82737724, -1.35557354, -0.1046374 , ...,  2.00893337,
        -0.02915533,  0.58595414],
       [-1.56610693,  0.73769513, -0.25833709, ..., -0.47367361,
         0.29082313, -1.9423032 ],
       [ 0.82737724,  0.73769513,  0.20276197, ..., -0.47367361,
        -0.79612661, -0.67817453]])
```

9.Splitting Data into Train and Test

In [28]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
```

In [29]:
```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(x_test.shape)
```

```
(623, 7)
(623, 1)
(268, 7)
(268, 7)
```

In [30]:
```
x.shape
```

Out[30]:
```
(891, 7)
```