PAMPANI SONU DURGA AVINASH

Assignment-3

21BCE9333

Importing the libraries

In [22]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Importing the dataset

In [57]:
```
data=pd.read_csv("Titanic-Dataset.csv")
```

In [5]:
```
data.head()
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN

In [6]:
```
1 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
```

In [7]:
```
1 data.describe()
```

Out[7]:

	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

Checking for Null values

In [3]:
```
1 data.isnull().any()
```

Out[3]:
```
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 [4]:
```
1 data.isnull().sum()
```

Out[4]:
```
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
```

Handling Null values

In [8]:
```
1 data["Age"]=data["Age"].fillna(data["Age"].mean())
```

In [11]:
```
1 data["Age"].isnull().sum()
```

Out[11]: 0

In [15]:
```
1 data["Cabin"]=data["Cabin"].fillna(data["Cabin"].mode()[0])
```

In [16]:
```
1 data["Cabin"].isnull().sum()
```

Out[16]: 0

In [18]:
```
1 data["Embarked"]=data["Embarked"].fillna(data["Embarked"].mode()[0])
```

In [19]:
```
1 data["Embarked"].isnull().sum()
```

Out[19]: 0

In [20]:
```
1 data.tail()
```

Out[20]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
886	887	0	2	Montvila, Rev. Juozas	male	27.000000	0	0	211536	13.00	B9(B9
887	888	1	1	Graham, Miss. Margaret Edith	female	19.000000	0	0	112053	30.00	B4
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	29.699118	1	2	W./C. 6607	23.45	B9(B9
889	890	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	111369	30.00	C14
890	891	0	3	Dooley, Mr. Patrick	male	32.000000	0	0	370376	7.75	B9(B9

Data Visualization

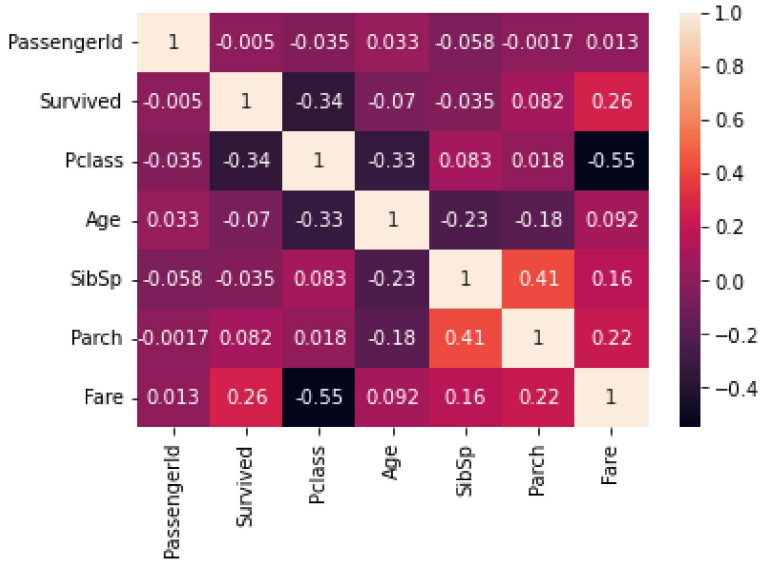
In [21]:
```
1 cor=data.corr()
2 cor
```

Out[21]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.033207	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.069809	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.331339	0.083081	0.018443	-0.549500
Age	0.033207	-0.069809	-0.331339	1.000000	-0.232625	-0.179191	0.091566
SibSp	-0.057527	-0.035322	0.083081	-0.232625	1.000000	0.414838	0.159651
Parch	-0.001652	0.081629	0.018443	-0.179191	0.414838	1.000000	0.216225
Fare	0.012658	0.257307	-0.549500	0.091566	0.159651	0.216225	1.000000

In [24]:
```
1 sns.heatmap(cor,annot=True)
```

Out[24]: <AxesSubplot:>

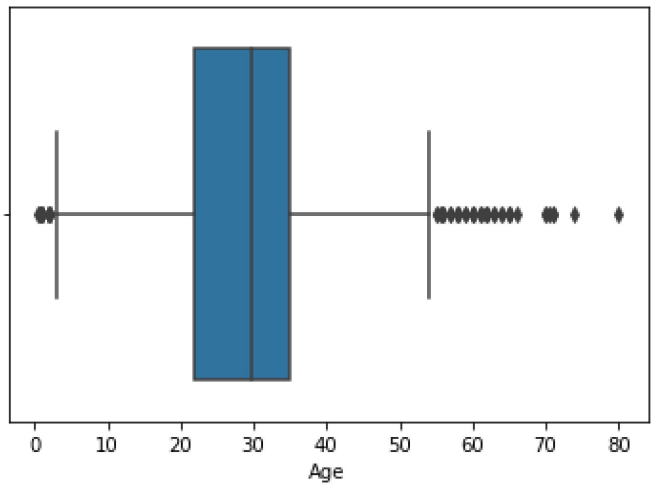


Outliers Detection

In [25]:
```
1 sns.boxplot(data["Age"])
```

```
C:\Users\Avinash\anaconda\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments without
an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[25]: <AxesSubplot:xlabel='Age'>



In [29]:
```
age_q1=data.Age.quantile(0.25)
age_q3=data.Age.quantile(0.75)
print(age_q1)
print(age_q3)
```

```
22.0
35.0
```

In [30]:
```
IQR_Age=age_q3-age_q1
IQR_Age
```

Out[30]: 13.0

In [31]:
```
ul_Age=age_q3+1.5*IQR_Age
ul_Age
```

Out[31]: 54.5

In [32]:
```
ll_Age=age_q1-1.5*IQR_Age
ll_Age
```

Out[32]: 2.5

In [33]:
```
median_Age=data["Age"].median()
median_Age
```

Out[33]: 29.69911764705882

In [34]:
```
data["Age"]=np.where(data["Age"]>ul_Age,median_Age,data["Age"])
```

In [37]:
```
data["Age"]=np.where(data["Age"]<ll_Age,median_Age,data["Age"])
```

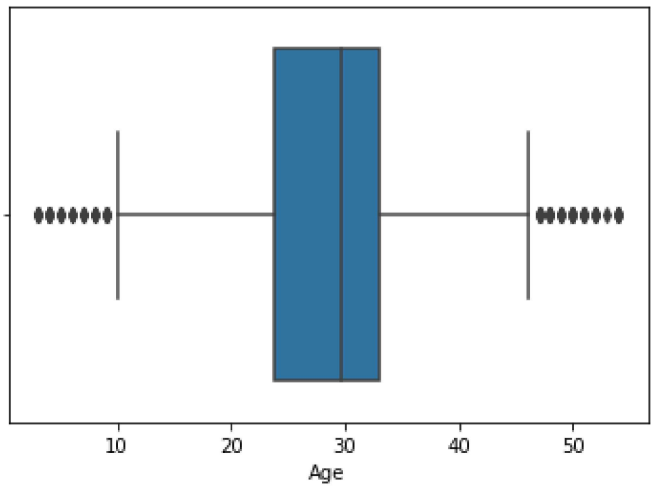
In [38]:
```
(data["Age"]>54.5).sum()
```

Out[38]: 0

In [40]:
```
1 sns.boxplot(data["Age"])
```

```
C:\Users\Avinash\anaconda\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments without
an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

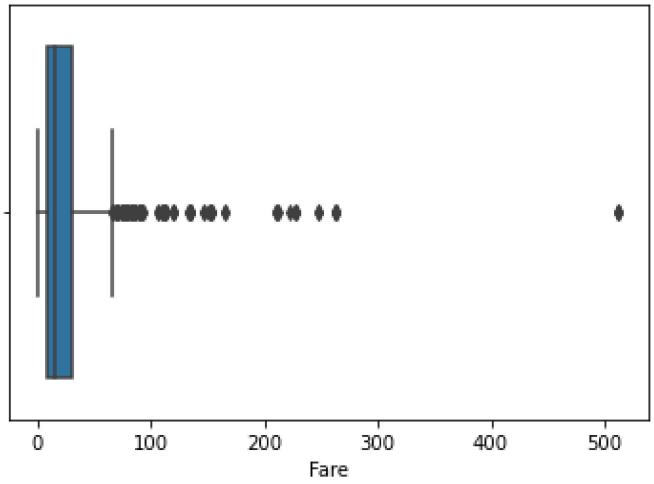
Out[40]: <AxesSubplot:xlabel='Age'>



In [41]:
```
1 sns.boxplot(data["Fare"])
```

```
C:\Users\Avinash\anaconda\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments without
an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[41]: <AxesSubplot:xlabel='Fare'>



In [42]:
```
fare_q1=data.Fare.quantile(0.25)
fare_q3=data.Fare.quantile(0.75)
print(fare_q1)
print(fare_q3)
```

```
7.9104
31.0
```

In [43]:
```
IQR_Fare=fare_q3-fare_q1
IQR_Fare
```

Out[43]: 23.0896

In [44]:
```
upperlimit_Fare=fare_q3+1.5*IQR_Fare
upperlimit_Fare
```

Out[44]: 65.6344

In [45]:
```
lower_limit_Fare=fare_q1-1.5*IQR_Fare
lower_limit_Fare
```

Out[45]: -26.724

In [46]:
```
median_Fare=data["Fare"].median()
median_Fare
```

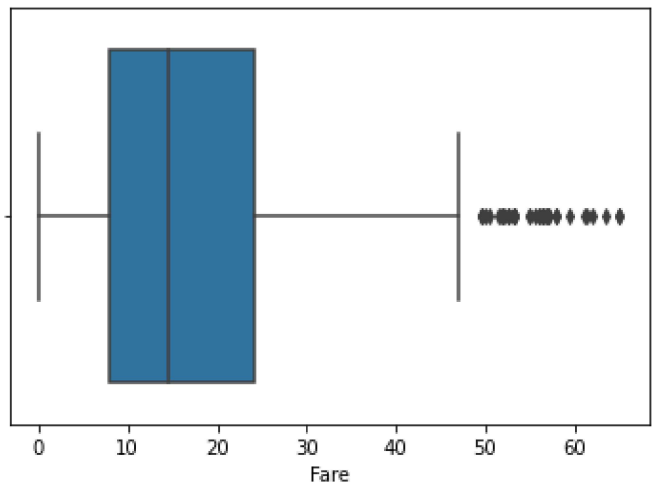
Out[46]: 14.4542

In [47]:
```
data['Fare']=np.where((data['Fare']>upperlimit_Fare),median_Fare,data['Fare']
```

In [48]:
```
sns.boxplot(data["Fare"])
```

```
C:\Users\Avinash\anaconda\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments without
an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[48]: <AxesSubplot:xlabel='Fare'>



In [49]:
```
1 (data["Fare"]>65).sum()
```

Out[49]: 0

Dropping unwanted Columns

In [58]:
```
1 data.drop(["Name"],axis=1,inplace=True)
```

In [59]:
```
1 data
```

Out[59]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarke
0	1	0	3	male	22.0	1	0	A/5 21171	7.2500	NaN	
1	2	1	1	female	38.0	1	0	PC 17599	71.2833	C85	
2	3	1	3	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	
3	4	1	1	female	35.0	1	0	113803	53.1000	C123	
4	5	0	3	male	35.0	0	0	373450	8.0500	NaN	
...	...	...	...	...	...	...	...	...	...	...	.
886	887	0	2	male	27.0	0	0	211536	13.0000	NaN	
887	888	1	1	female	19.0	0	0	112053	30.0000	B42	
888	889	0	3	female	NaN	1	2	W./C. 6607	23.4500	NaN	
889	890	1	1	male	26.0	0	0	111369	30.0000	C148	
890	891	0	3	male	32.0	0	0	370376	7.7500	NaN	

891 rows × 11 columns

In [60]:
```
1 data.drop(['Ticket'],axis=1,inplace=True)
```

In [61]:
```
1 data
```

Out[61]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	1	0	3	male	22.0	1	0	7.2500	NaN	S
1	2	1	1	female	38.0	1	0	71.2833	C85	C
2	3	1	3	female	26.0	0	0	7.9250	NaN	S
3	4	1	1	female	35.0	1	0	53.1000	C123	S
4	5	0	3	male	35.0	0	0	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	male	27.0	0	0	13.0000	NaN	S
887	888	1	1	female	19.0	0	0	30.0000	B42	S
888	889	0	3	female	NaN	1	2	23.4500	NaN	S
889	890	1	1	male	26.0	0	0	30.0000	C148	C
890	891	0	3	male	32.0	0	0	7.7500	NaN	Q

891 rows × 10 columns

In [62]:
```
1 data.drop(["PassengerId"],axis=1,inplace=True)
```

In [64]:
```
1 data.drop(["Cabin"],axis=1,inplace=True)
```

In [65]:
```
1 data
```

Out[65]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S
1	1	1	female	38.0	1	0	71.2833	C
2	1	3	female	26.0	0	0	7.9250	S
3	1	1	female	35.0	1	0	53.1000	S
4	0	3	male	35.0	0	0	8.0500	S
...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S
887	1	1	female	19.0	0	0	30.0000	S
888	0	3	female	NaN	1	2	23.4500	S
889	1	1	male	26.0	0	0	30.0000	C
890	0	3	male	32.0	0	0	7.7500	Q

891 rows × 8 columns

Splitting the data into dependent and independent

In [66]:
```
y=data["Survived"]
```

In [67]:
```
y.head()
```

Out[67]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [68]:
```
data
```

Out[68]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S
1	1	1	female	38.0	1	0	71.2833	C
2	1	3	female	26.0	0	0	7.9250	S
3	1	1	female	35.0	1	0	53.1000	S
4	0	3	male	35.0	0	0	8.0500	S
...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S
887	1	1	female	19.0	0	0	30.0000	S
888	0	3	female	NaN	1	2	23.4500	S
889	1	1	male	26.0	0	0	30.0000	C
890	0	3	male	32.0	0	0	7.7500	Q

891 rows × 8 columns

Encoding

In [69]:
```
from sklearn.preprocessing import LabelEncoder
```

In [70]:
```
le=LabelEncoder()
```

In [71]:
```
data["Sex"]=le.fit_transform(data["Sex"])
```

In [72]:
```
1 data["Sex"]
```

Out[72]:
```
0      1
1      0
2      0
3      0
4      1
      ..
886    1
887    0
888    0
889    1
890    1
Name: Sex, Length: 891, dtype: int32
```

In [73]:
```
1 data.head()
```

Out[73]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	1	22.0	1	0	7.2500	S
1	1	1	0	38.0	1	0	71.2833	C
2	1	3	0	26.0	0	0	7.9250	S
3	1	1	0	35.0	1	0	53.1000	S
4	0	3	1	35.0	0	0	8.0500	S

In [74]:
```
1 data["Embarked"]=le.fit_transform(data["Embarked"])
```

In [75]:
```
1 data["Embarked"].head()
```

Out[75]:
```
0    2
1    0
2    2
3    2
4    2
Name: Embarked, dtype: int32
```

In [76]:
```
1 data["Pclass"].nunique()
```

Out[76]:
```
3
```

In [77]:
```
1 data["Pclass"].unique()
```

Out[77]:
```
array([3, 1, 2], dtype=int64)
```

In [78]:
```
1 data["Sex"].unique()
```

Out[78]:
```
array([1, 0])
```

In [79]:
```
1 data["Embarked"].unique()
```

Out[79]:
```
array([2, 0, 1, 3])
```

Feature Scaling

In [82]:
```
from sklearn.preprocessing import StandardScaler
```

In [83]:
```
sc=StandardScaler()
```

In [84]:
```
x_train=sc.fit_transform(x_train)
```

In [85]:
```
x_train
```

Out[85]:
```
array([[ 1.25474307, -1.5325562 ,  0.72592065, ..., -0.47299765,
        -0.12253019,  0.56011053],
       [ 1.25474307, -1.5325562 , -1.37756104, ..., -0.47299765,
         0.91812372, -2.02469583],
       [-0.79697591,  0.84844757,  0.72592065, ...,  1.93253327,
         0.29950338,  0.56011053],
       ...,
       [-0.79697591,  0.84844757,  0.72592065, ..., -0.47299765,
        -0.51276504, -0.73229265],
       [ 1.25474307,  0.84844757, -1.37756104, ..., -0.47299765,
        -0.31228976,  0.56011053],
       [-0.79697591, -0.34205431,  0.72592065, ...,  0.72976781,
         0.13566725,  0.56011053]])
```

In [86]:
```
x_test=sc.fit_transform(x_test)
```

In [87]:
```
x_test
```

Out[87]:
```
array([[-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.324475  , -1.76531134],
       [-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.45513843,  0.63014911],
       [-0.77151675,  0.77963055,  0.76537495, ...,  0.87064484,
        -0.04706937, -0.56758111],
       ...,
       [-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.32455255, -1.76531134],
       [ 1.29614814,  0.77963055, -1.30654916, ..., -0.47809977,
        -0.45616356,  0.63014911],
       [-0.77151675, -1.64991582,  0.76537495, ..., -0.47809977,
        -0.07362838, -1.76531134]])
```

Spliting the train and test data

In [88]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(data,y,test_size=0.3,random_st
```

In [89]:
```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[89]:
```
((623, 8), (268, 8), (623,), (268,))
```