

name -amardeep singh

IMPORTING THE LIBRAREIS

```
In [1]: #importing the necessary values
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

IMPORTING THE DATASET

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

```
In [3]: df.head(5)
```

```
Out[3]:
```

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

```
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.corr()
```

C:\Users\hp\AppData\Local\Temp\ipykernel_3472\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()
```

```
Out[5]:
```

	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 [6]: df.corr().Survived.sort_values(ascending = False)
```

C:\Users\hp\AppData\Local\Temp\ipykernel_3472\1287823212.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().Survived.sort_values(ascending = False)
```

```
Out[6]:
```

Survived	1.000000
Fare	0.257307
Parch	0.081629
PassengerId	-0.005007
SibSp	-0.035322
Age	-0.077221
Pclass	-0.338481

Name: Survived, dtype: float64

CHECKING FOR NULL VALUES

```
In [7]: df.isnull().sum()
```

```
Out[7]: 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 [8]: df.shape
```

```
Out[8]: (891, 12)
```

```
In [9]: #REMOVING THE CABIN COLUMN FROM THE DATASET BECAUSE IT HAS SO MANY NULL VALUES
df.drop(columns=['Cabin'], inplace=True)
```

```
In [ ]:
```

```
In [10]: #FILLING THE AGE COLUMN WHICH HAVE MISSING VALUES WITH MEDIAN
df['Age'].fillna(df['Age'].median(), inplace=True)
```

```
In [11]: #FILLING THE EMBARKED WITH MOST REPEATED AS IT IS A CATEGORICAL
most_rep=df['Embarked'].value_counts().idxmax()
df["Embarked"].replace(np.nan, most_rep, inplace=True)
```

```
In [12]: #CHECKING IF ANY MORE NULL VALUES IS PRESENT IN THE DATAFRAME
df.isnull().sum()
```

```
Out[12]: PassengerId      0
Survived      0
Pclass        0
Name          0
Sex           0
Age           0
SibSp         0
Parch         0
Ticket        0
Fare          0
Embarked      0
dtype: int64
```

DATA-VISUALISATION

```
In [13]: df.head(2)
```

```
Out[13]:
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...)	female	38.0	1	0	PC 17599	71.2833	

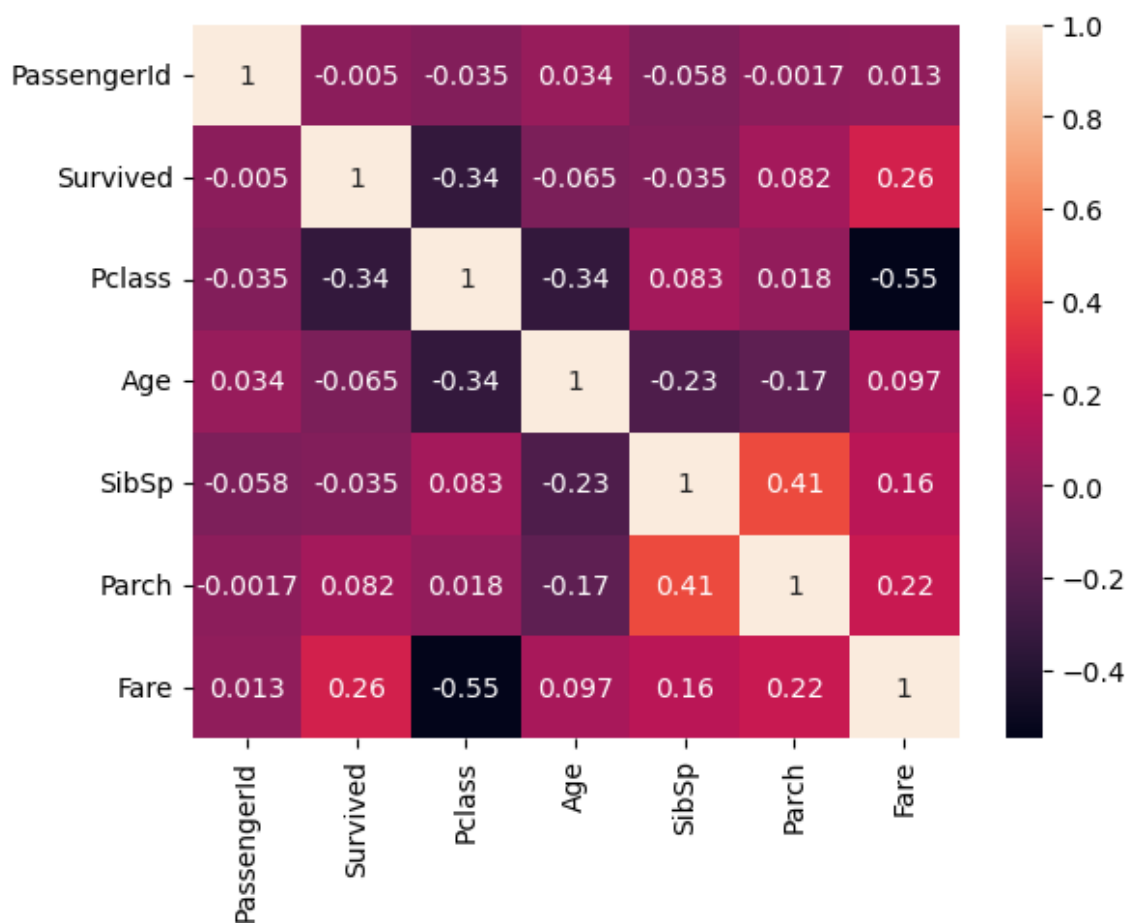
DATA VISUALISATION

```
In [14]: sns.heatmap(df.corr() , annot = True)
```

C:\Users\hp\AppData\Local\Temp\ipykernel_3472\2802813450.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.

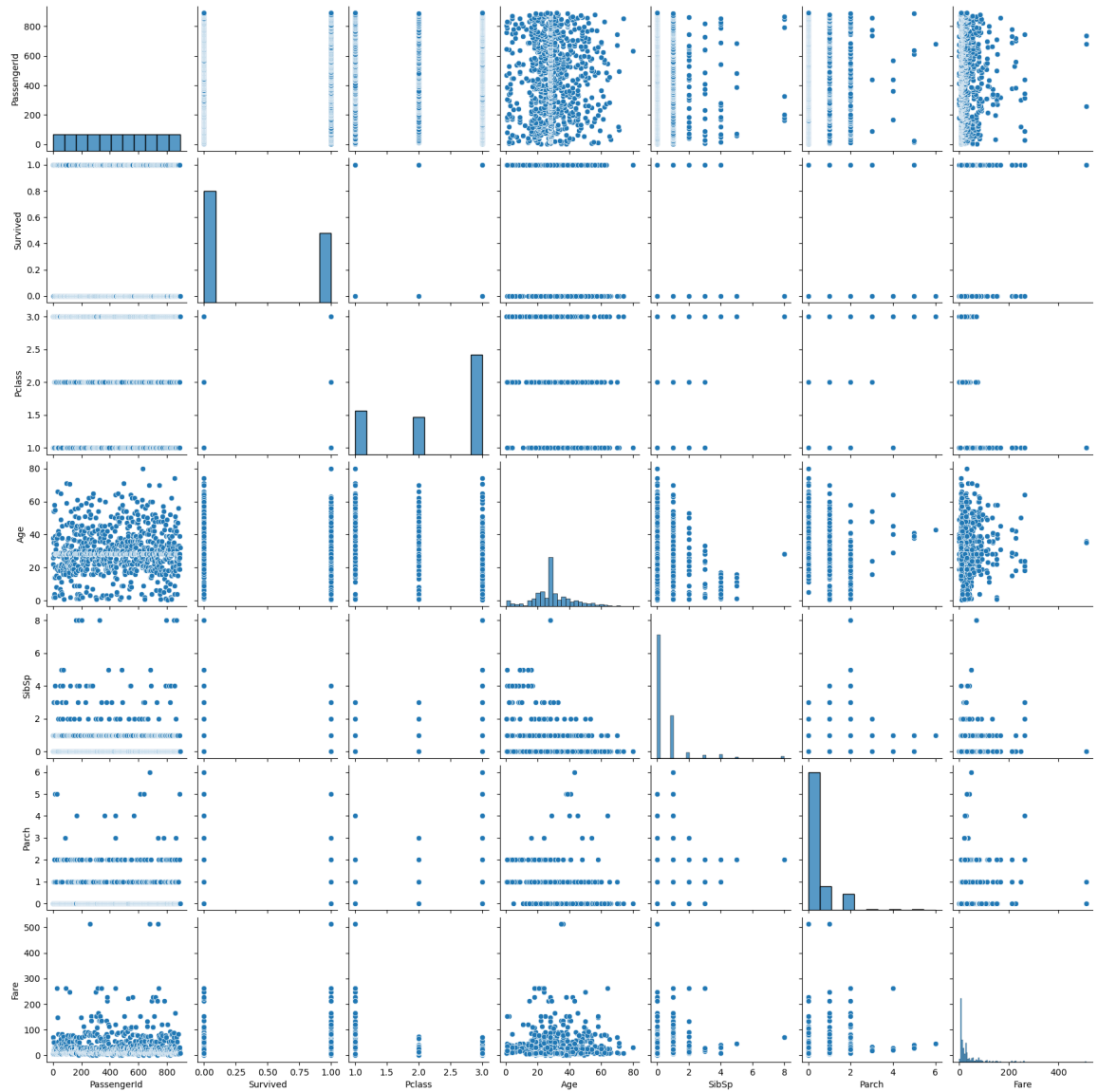
```
sns.heatmap(df.corr() , annot = True)
```

```
Out[14]: <Axes: >
```



```
In [15]: sns.pairplot(df)
```

```
Out[15]: <seaborn.axisgrid.PairGrid at 0x25287623810>
```



OULIER DETECTION

```
In [16]: df.head(20)
```

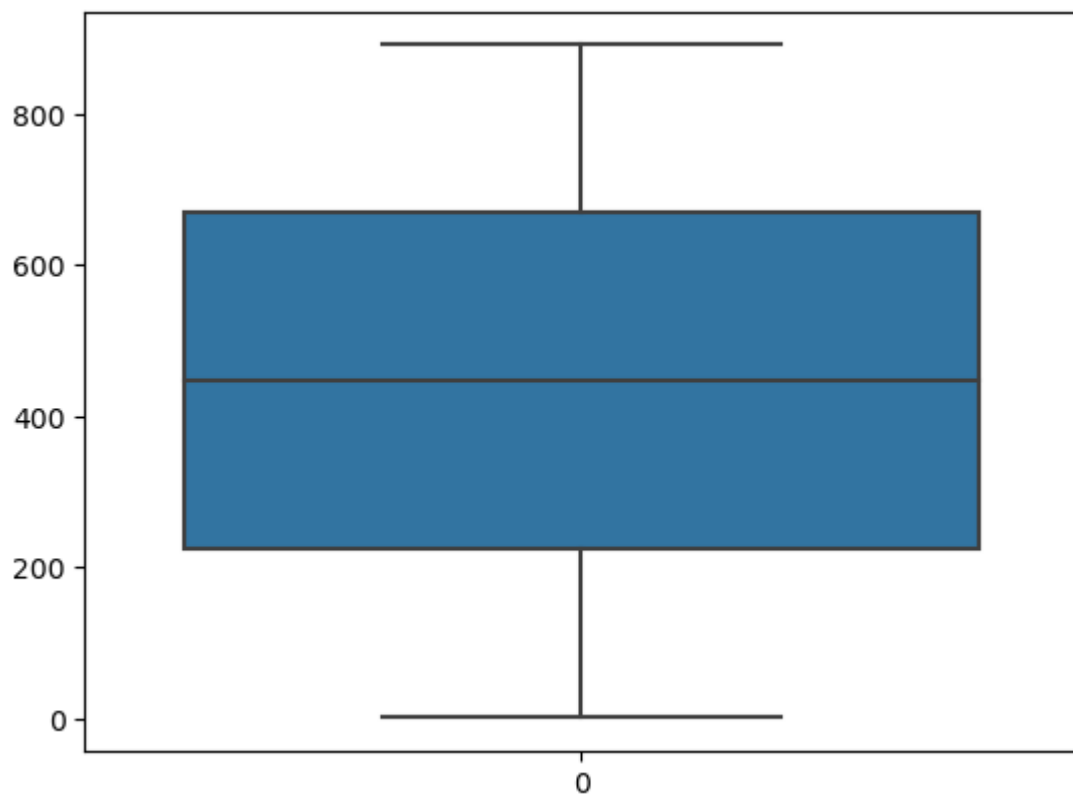
Out[16]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.25
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...)	female	38.0	1	0	PC 17599	71.28
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.92
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.10
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.05
5	6	0	3	Moran, Mr. James	male	28.0	0	0	330877	8.45
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.86
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.07
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.13
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.07
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.70
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.55
12	13	0	3	Saunderscock, Mr. William Henry	male	20.0	0	0	A/5. 2151	8.05
13	14	0	3	Andersson, Mr. Anders Johan	male	39.0	1	5	347082	31.27
14	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14.0	0	0	350406	7.85
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55.0	0	0	248706	16.00
16	17	0	3	Rice, Master. Eugene	male	2.0	4	1	382652	29.12

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
17	18	1	2	Williams, Mr. Charles Eugene	male	28.0	0	0	244373	13.00
18	19	0	3	Vander Planke, Mrs. Julius (Emelia Maria Vande...	female	31.0	1	0	345763	18.00
19	20	1	3	Masselmani, Mrs. Fatima	female	28.0	0	0	2649	7.22

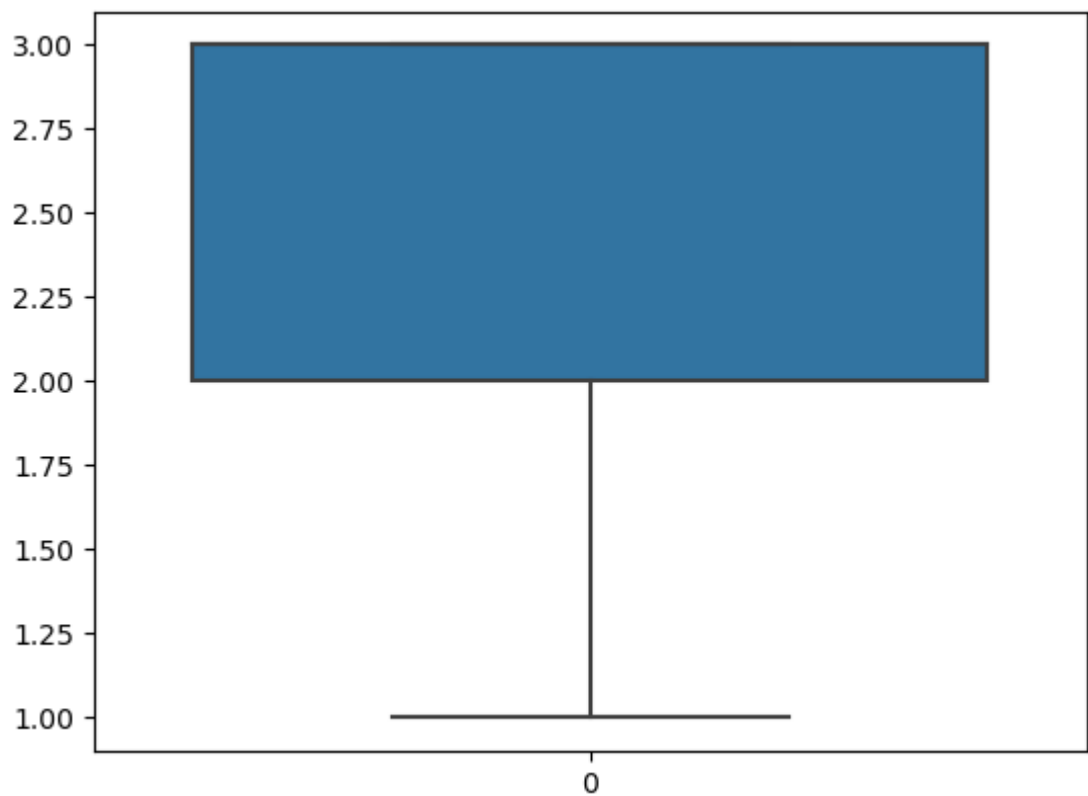
```
In [17]: #Checking if outliers are present in PassengerId
sns.boxplot(df.PassengerId)
```

```
Out[17]: <Axes: >
```



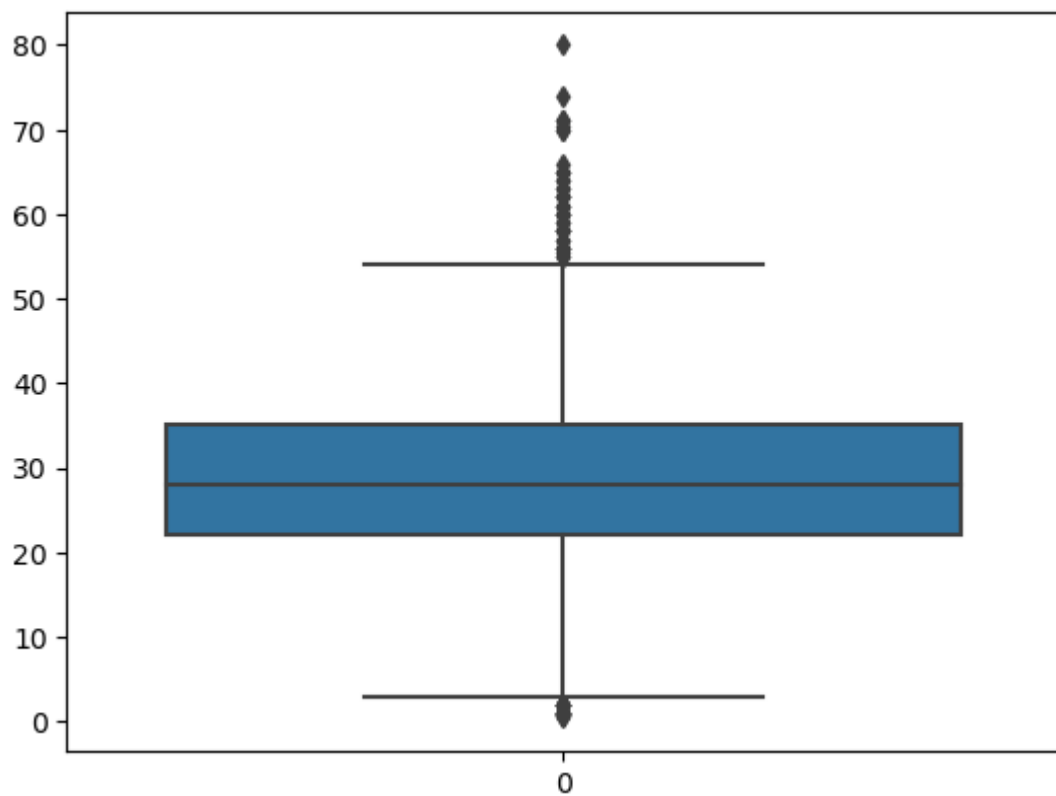

```
In [18]: #checking if any outliers are present in PClass  
sns.boxplot(df.Pclass)
```

Out[18]: <Axes: >



```
In [19]: #Checking if any outliers present in age column  
sns.boxplot(df.Age)
```

Out[19]: <Axes: >



In [20]: *#yes there are outliers in age-column let's replace them*
outlier removal by replacement with median

```
q1 = df.Age.quantile(0.25) #for finding the q1
q3 = df.Age.quantile(0.75) #for finding the q3
print(q1)
print(q3)
```

```
22.0
35.0
```

In [21]: `IQR = q3-q1`
`print(IQR)`

```
13.0
```

In [22]: *#upper_limit*
`upper_limit = q3+1.5*IQR`
#Lower_limit
`lower_limit = q1-1.5*IQR`

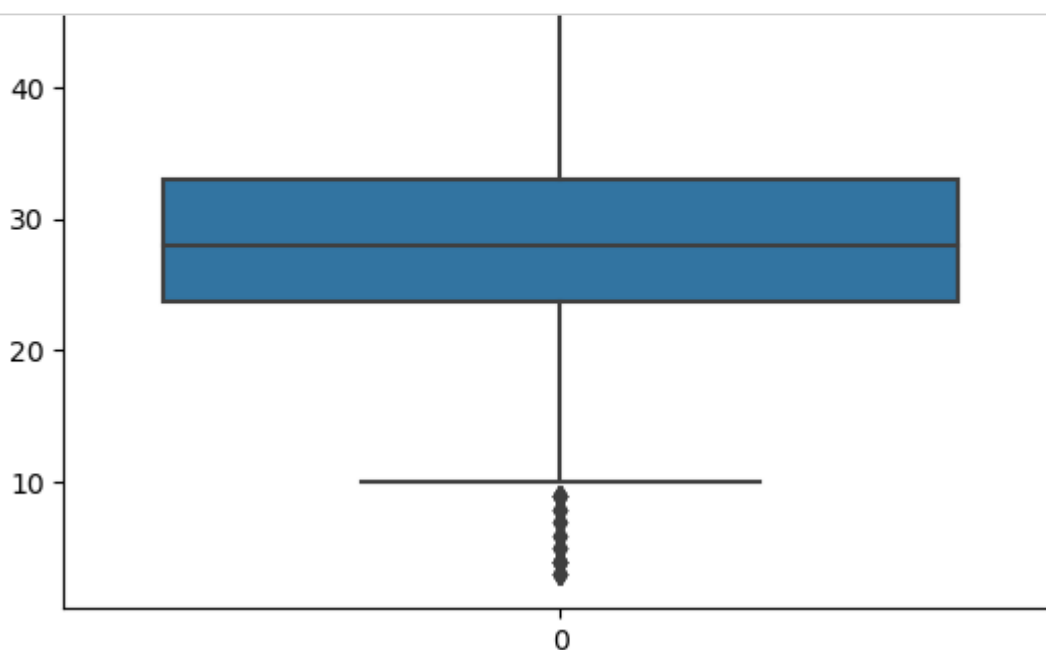
`print(upper_limit)`
`print(lower_limit)`

```
54.5
2.5
```

In [23]: *#finding the median for replacing the outliers with median*
`med = df['Age'].median()`

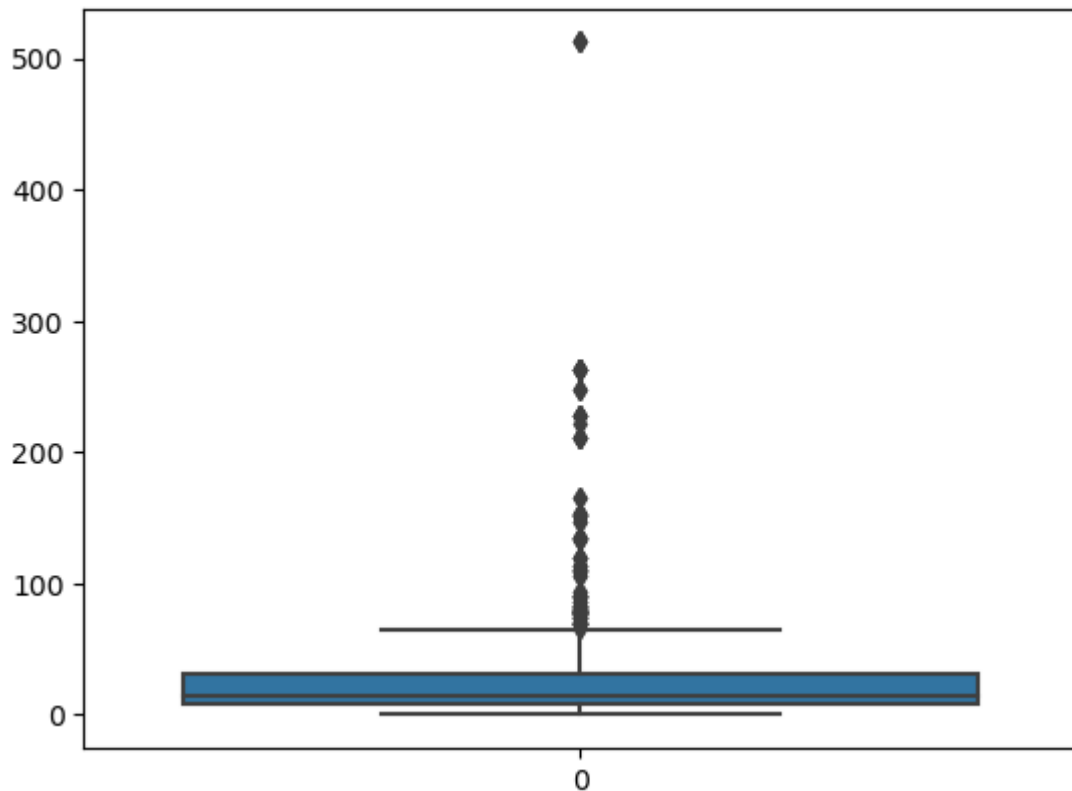
In [24]: *# replacing the outliers with the median*
`df['Age'] = np.where((df['Age'] > upper_limit) | (df['Age'] < lower_limit),`

In [25]: `sns.boxplot(df.Age)`



```
In [26]: #Checking if any outliers present in farecolumn  
sns.boxplot(df.Fare)
```

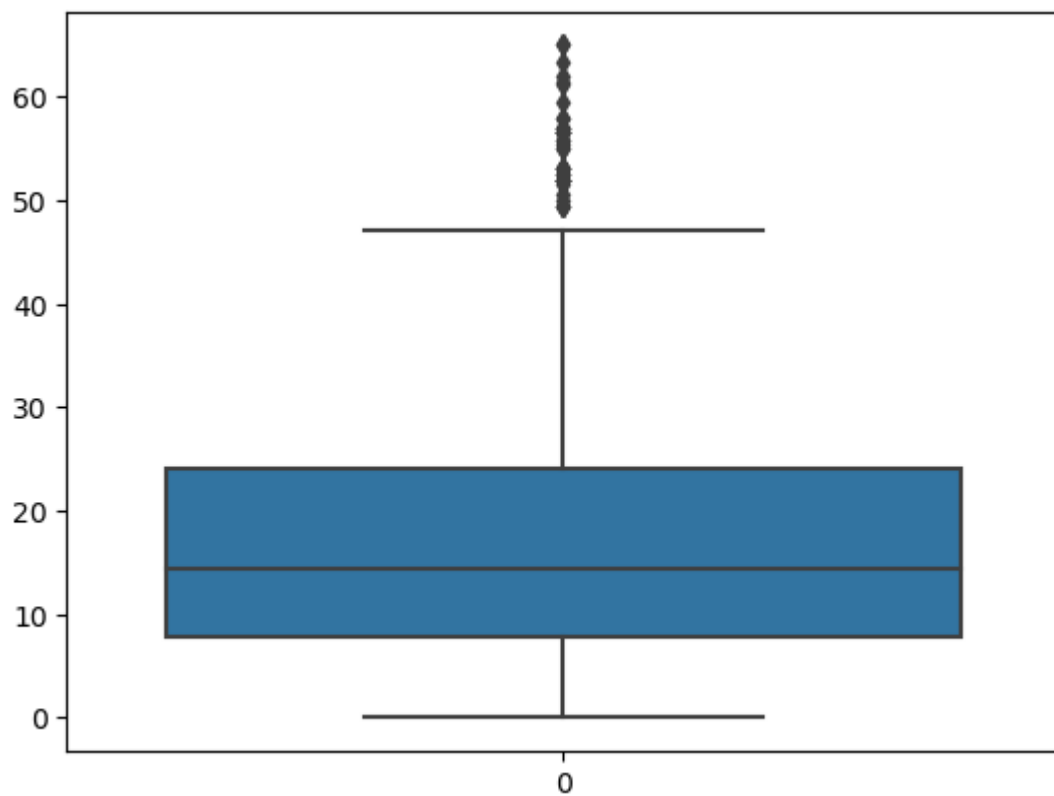
Out[26]: <Axes: >



```
In [27]: #yes outliers are there in the Fare column so let's replace again  
q1 = df.Fare.quantile(0.25) #for finding the q1  
q3 = df.Fare.quantile(0.75)  
IQR = q3-q1  
#upper_limit  
upper_limit = q3+1.5*IQR  
#Lower_limit  
lower_limit = q1-1.5*IQR  
med = df['Fare'].median()  
df['Fare'] = np.where((df['Fare'] > upper_limit) | (df['Fare'] < lower_limit), med, df['Fare'])
```

```
In [28]: #Checking if any outliers present in farecolumn  
sns.boxplot(df.Fare)
```

Out[28]: <Axes: >



SPLITTING DEPENDENT AND INDEPENDENT COLUMNS

```
In [31]: df.head(10)
```

```
Out[31]:
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	14.4542
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500
5	6	0	3	Moran, Mr. James	male	28.0	0	0	330877	8.4583
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625
7	8	0	3	Palsson, Master. Gosta Leonard	male	28.0	3	1	349909	21.0750
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708

```
In [39]: # the features (X) and the target (y)
#independent
X = df[['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked']]
print(type(X))
#dependent
y = df['Survived']
```

```
<class 'pandas.core.frame.DataFrame'>
```

ENCODING

```
In [40]: #here sex and Embarked has categorial values so we have to encode
from sklearn.preprocessing import LabelEncoder

# Initialize the LabelEncoder
label_encoder = LabelEncoder()

# Apply label encoding to 'Sex' as there only two categories
X.loc[:, 'Sex'] = label_encoder.fit_transform(X['Sex'])
# Apply one-hot encoding to 'Embarked' column as there are more categorical
X = pd.get_dummies(X, columns=['Embarked'], prefix=['Embarked'])
```

C:\Users\hp\AppData\Local\Temp\ipykernel_3472\3595101015.py:8: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
X.loc[:, 'Sex'] = label_encoder.fit_transform(X['Sex'])
C:\Users\hp\AppData\Local\Temp\ipykernel_3472\3595101015.py:8: Deprecation
Warning: In a future version, `df.iloc[:, i] = newvals` will attempt to se
t the values inplace instead of always setting a new array. To retain the
old behavior, use either `df[df.columns[i]] = newvals` or, if columns are
non-unique, `df.isetitem(i, newvals)`
X.loc[:, 'Sex'] = label_encoder.fit_transform(X['Sex'])
```

```
In [43]: X.head(3)
```

```
Out[43]:
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_C	Embarked_Q	Embarked_S
0	3	1	22.0	1	0	7.2500	0	0	1
1	1	0	38.0	1	0	14.4542	1	0	0
2	3	0	26.0	0	0	7.9250	0	0	1

FEATURE SCALING

```
In [46]: from sklearn.preprocessing import StandardScaler


# Initialize the StandardScaler
scaler = StandardScaler()

# Apply standardization to your features (X)
X_scaled = pd.DataFrame(scaler.fit_transform(X))
```

In [47]: X_scaled.head()

Out[47]:

	0	1	2	3	4	5	6	7	
0	0.827377	0.737695	-0.661724	0.432793	-0.473674	-0.797554	-0.482043	-0.307562	0.611
1	-1.566107	-1.355574	0.972921	0.432793	-0.473674	-0.230556	2.074505	-0.307562	-1.62
2	0.827377	-1.355574	-0.253063	-0.474545	-0.473674	-0.744429	-0.482043	-0.307562	0.611
3	-1.566107	-1.355574	0.666425	0.432793	-0.473674	2.811012	-0.482043	-0.307562	0.611
4	0.827377	0.737695	0.666425	-0.474545	-0.473674	-0.734591	-0.482043	-0.307562	0.611



SPLITTING DATA IN TO TRAIN AND TEST SET

In [50]: *#TRAIN TEST AND SPLIT*

```
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,r
```

In [51]: `print(x_train.shape , x_test.shape , y_train.shape , y_test.shape)`

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

In []: