

Name : G Aniesh Reddy

Reg no : 21BCE7058

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df = pd.read_csv('Titanic-Dataset.csv')
df
```

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	E
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	
...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	

891 rows × 12 columns

In [3]:

```
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 [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.isnull().any()
```

```
Out[5]:
```

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 [6]:
```

```
df.corr()
```

C:\Users\hp\AppData\Local\Temp\ipykernel_4088\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[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]:
```

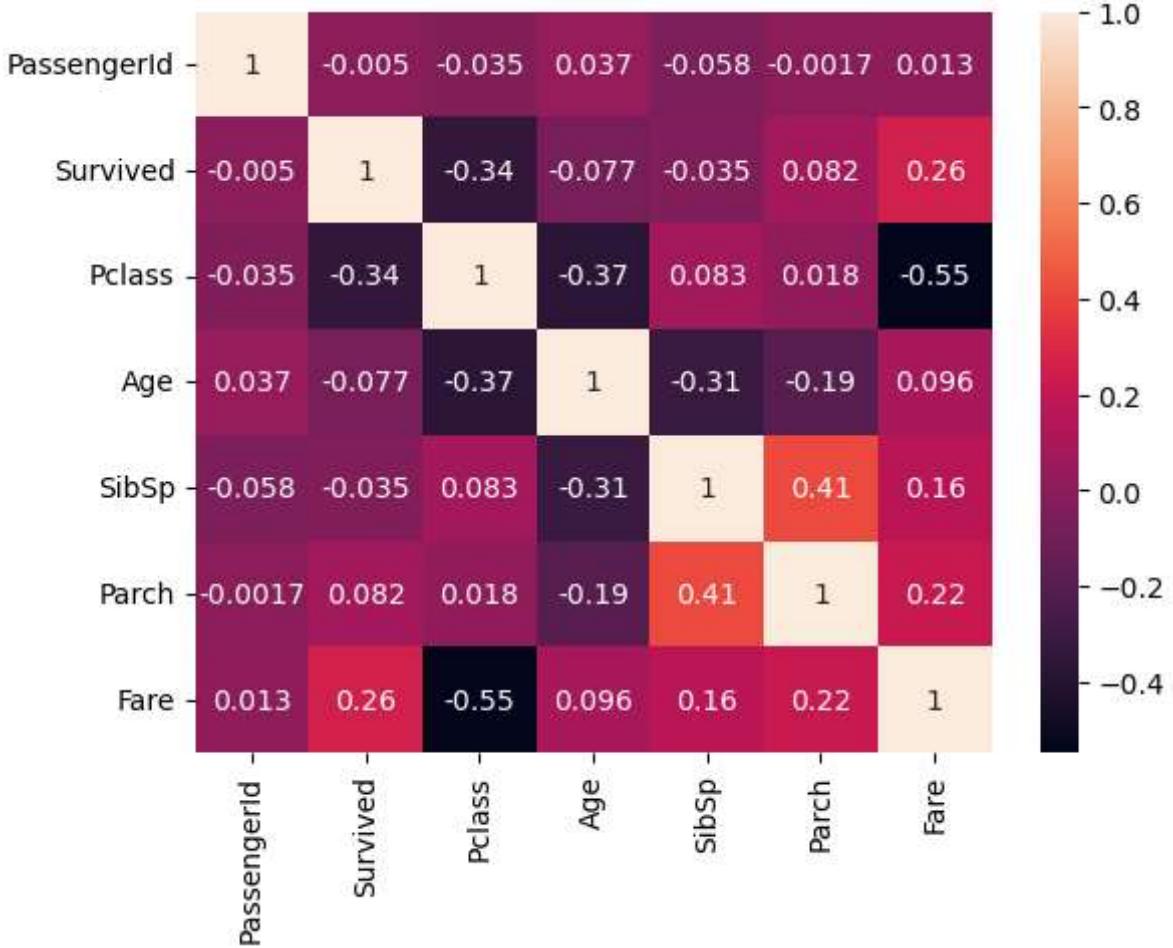
```
sns.heatmap(df.corr(), annot=True)
```

C:\Users\hp\AppData\Local\Temp\ipykernel_4088\4277794465.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[7]:
```

```
<Axes: >
```



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

```
Out[8]: 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 [9]: `df['Age'].mean()`

```
Out[9]: 29.69911764705882
```

In [10]: `df["Age"] = df["Age"].fillna(df["Age"].mean())`

In [11]: `df`

Out[11]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	Na
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...)	female	38.000000	1	0	PC 17599	71.2833	C8
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	Na
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	C12
4	5	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500	Na
...
886	887	0	2	Montvila, Rev. Juozas	male	27.000000	0	0	211536	13.0000	Na
887	888	1	1	Graham, Miss. Margaret Edith	female	19.000000	0	0	112053	30.0000	B4
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	29.699118	1	2	W./C. 6607	23.4500	Na
889	890	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	111369	30.0000	C14
890	891	0	3	Dooley, Mr. Patrick	male	32.000000	0	0	370376	7.7500	Na

891 rows × 12 columns



In [12]:

```
df["Cabin"] = df["Cabin"].fillna(df["Cabin"].mode()[0])
```

In [13]:

df

Out[13]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	BS B5
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...)	female	38.000000	1	0	PC 17599	71.2833	C8
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	BS B5
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	C12
4	5	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500	BS B5
...
886	887	0	2	Montvila, Rev. Juozas	male	27.000000	0	0	211536	13.0000	BS B5
887	888	1	1	Graham, Miss. Margaret Edith	female	19.000000	0	0	112053	30.0000	B4
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	29.699118	1	2	W./C. 6607	23.4500	BS B5
889	890	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	111369	30.0000	C14
890	891	0	3	Dooley, Mr. Patrick	male	32.000000	0	0	370376	7.7500	BS B5

891 rows × 12 columns



In [14]:

```
df["Embarked"] = df["Embarked"].fillna(df["Embarked"].mode()[0])
```

In [15]:

```
df
```

Out[15]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	BS B5
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...)	female	38.000000	1	0	PC 17599	71.2833	C8
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	BS B5
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	C12
4	5	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500	BS B5
...
886	887	0	2	Montvila, Rev. Juozas	male	27.000000	0	0	211536	13.0000	BS B5
887	888	1	1	Graham, Miss. Margaret Edith	female	19.000000	0	0	112053	30.0000	B4
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	29.699118	1	2	W./C. 6607	23.4500	BS B5
889	890	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	111369	30.0000	C14
890	891	0	3	Dooley, Mr. Patrick	male	32.000000	0	0	370376	7.7500	BS B5

891 rows × 12 columns

In [16]: `df.isnull().any()`

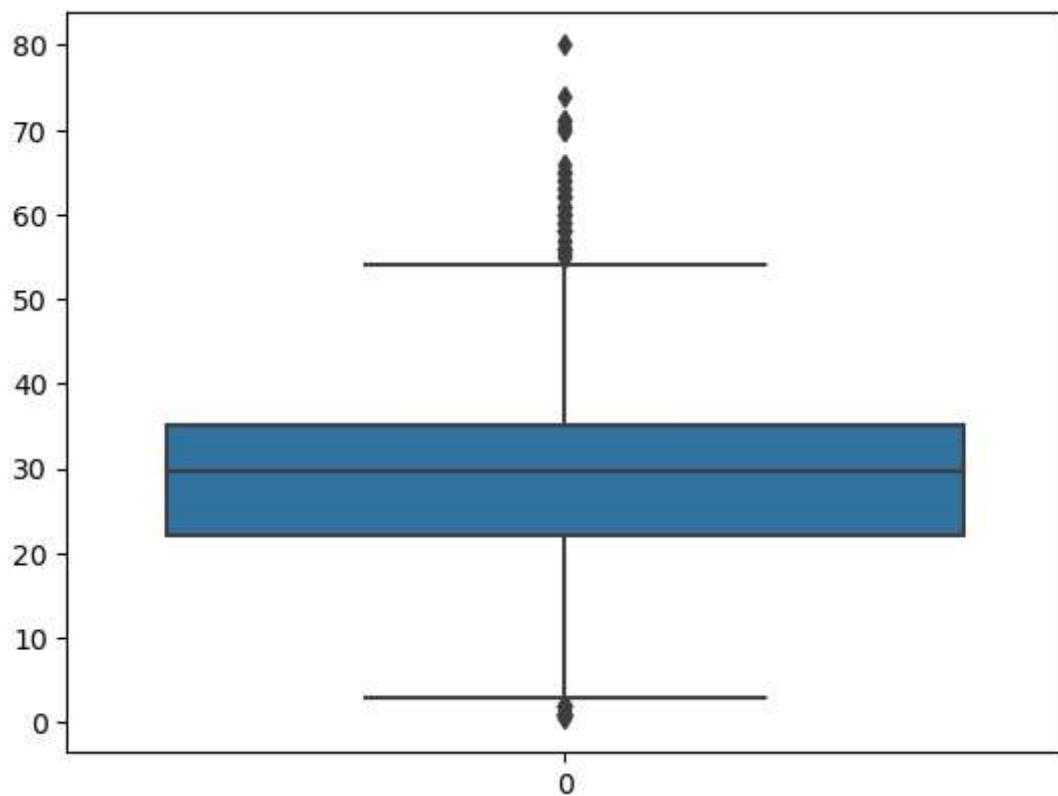
Out[16]:

PassengerId	False
Survived	False
Pclass	False
Name	False
Sex	False
Age	False
SibSp	False
Parch	False
Ticket	False
Fare	False
Cabin	False
Embarked	False

dtype: bool

In [17]: `sns.boxplot(df.Age)`

Out[17]: <Axes: >



In [18]: `q1 = df.Age.quantile(0.25)`

In [19]: `q3 = df.Age.quantile(0.75)`

In [20]: `print(q1)`

```
22.0
```

```
In [21]: print(q3)
```

```
35.0
```

```
In [22]: IQR = q3-q1  
IQR
```

```
Out[22]: 13.0
```

```
In [23]: upper_limit = q3+1.5*IQR  
upper_limit
```

```
Out[23]: 54.5
```

```
In [24]: lower_limit = q1-1.5*IQR  
lower_limit
```

```
Out[24]: 2.5
```

```
In [25]: df.median()
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_4088\530051474.py:1: FutureWarning: The default value of numeric_only in DataFrame.median is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=None' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
```

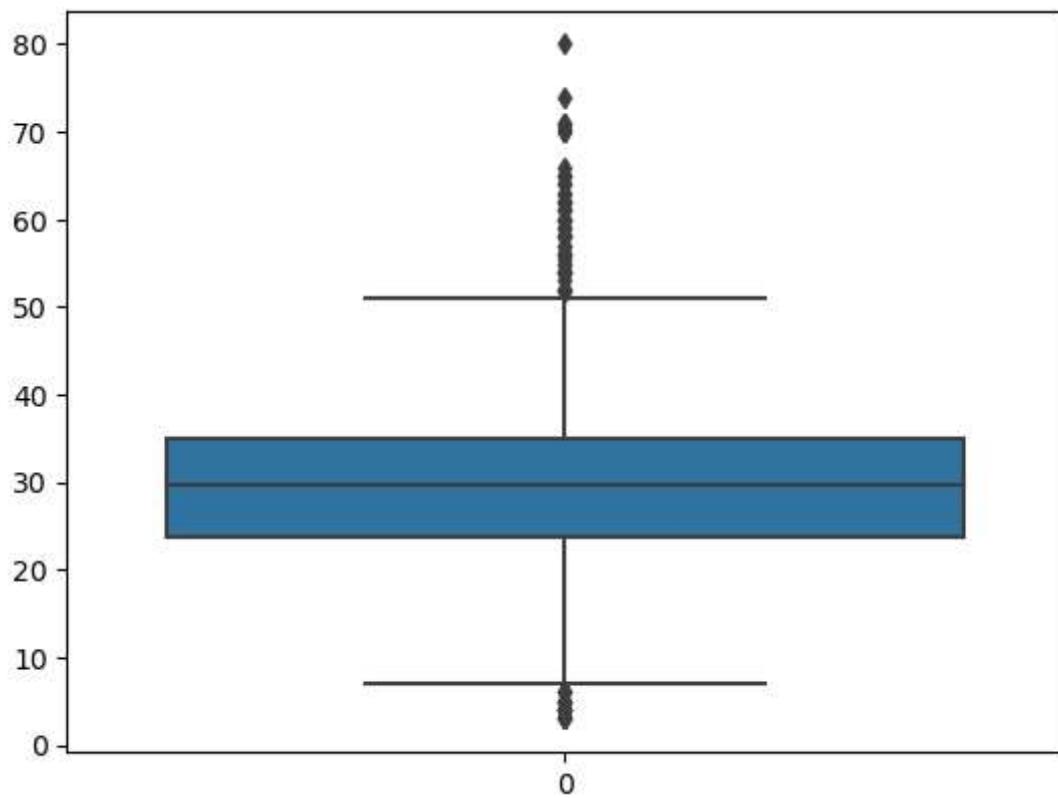
```
df.median()
```

```
Out[25]: PassengerId    446.000000  
Survived      0.000000  
Pclass        3.000000  
Age          29.699118  
SibSp        0.000000  
Parch        0.000000  
Fare         14.454200  
dtype: float64
```

```
In [26]: df['Age'] = np.where(df['Age']<lower_limit,29.69,df['Age'])
```

```
In [27]: sns.boxplot(df.Age)
```

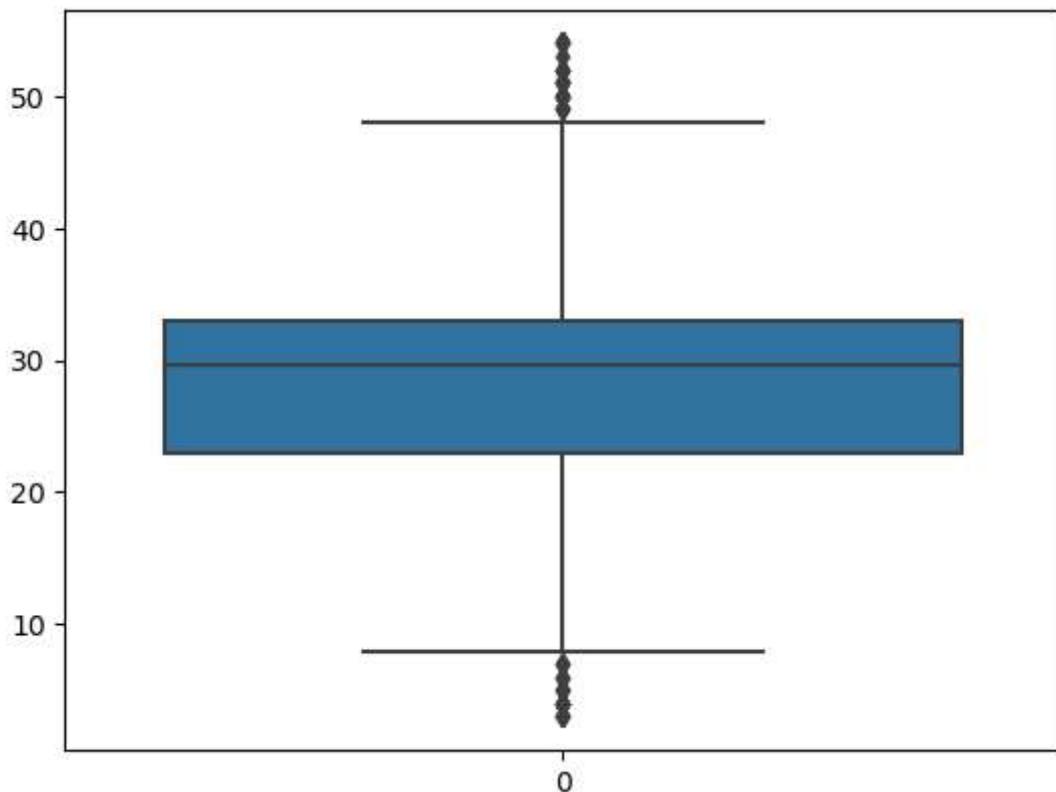
```
Out[27]: <Axes: >
```



```
In [28]: df = df[df.Age<upper_limit]
```

```
In [29]: sns.boxplot(df.Age)
```

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



```
In [64]:
```

```
from scipy import stats
z_scores = np.abs(stats.zscore(df['Age']))
max_threshold=3
outliers = df['Age'][z_scores > max_threshold]

# Print and visualize the outliers
print("Outliers detected using Z-Score:")
print(outliers)
```

```
Outliers detected using Z-Score:
Series([], Name: Age, dtype: float64)
```

```
In [65]:
```

```
from scipy import stats
z_scores = np.abs(stats.zscore(df['Fare']))
max_threshold=3
outliers = df['Fare'][z_scores > max_threshold]

# Print and visualize the outliers
print("Outliers detected using Z-Score:")
print(outliers)
```

```
Outliers detected using Z-Score:
27    263.0000
88    263.0000
118   247.5208
258   512.3292
299   247.5208
311   262.3750
341   263.0000
377   211.5000
```

```

380    227.5250
527    221.7792
557    227.5250
679    512.3292
689    211.3375
700    227.5250
716    227.5250
730    211.3375
737    512.3292
742    262.3750
779    211.3375
Name: Fare, dtype: float64

```

```
In [78]: column_name = 'Fare'

# Calculate the first quartile (Q1) and third quartile (Q3)
Q1 = df[column_name].quantile(0.25)
Q3 = df[column_name].quantile(0.75)

# Calculate the IQR
IQR = Q3 - Q1

# Define the Lower and upper bounds for outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Filter rows with values outside the IQR bounds
df_cleaned = df[(df[column_name] > lower_bound) & (df[column_name] < upper_bound)]

# Display the original and cleaned DataFrame sizes
print(f"Original DataFrame size: {df.shape}")
print(f"Cleaned DataFrame size: {df_cleaned.shape}")
df_cleaned
```

Original DataFrame size: (849, 12)

Cleaned DataFrame size: (741, 12)

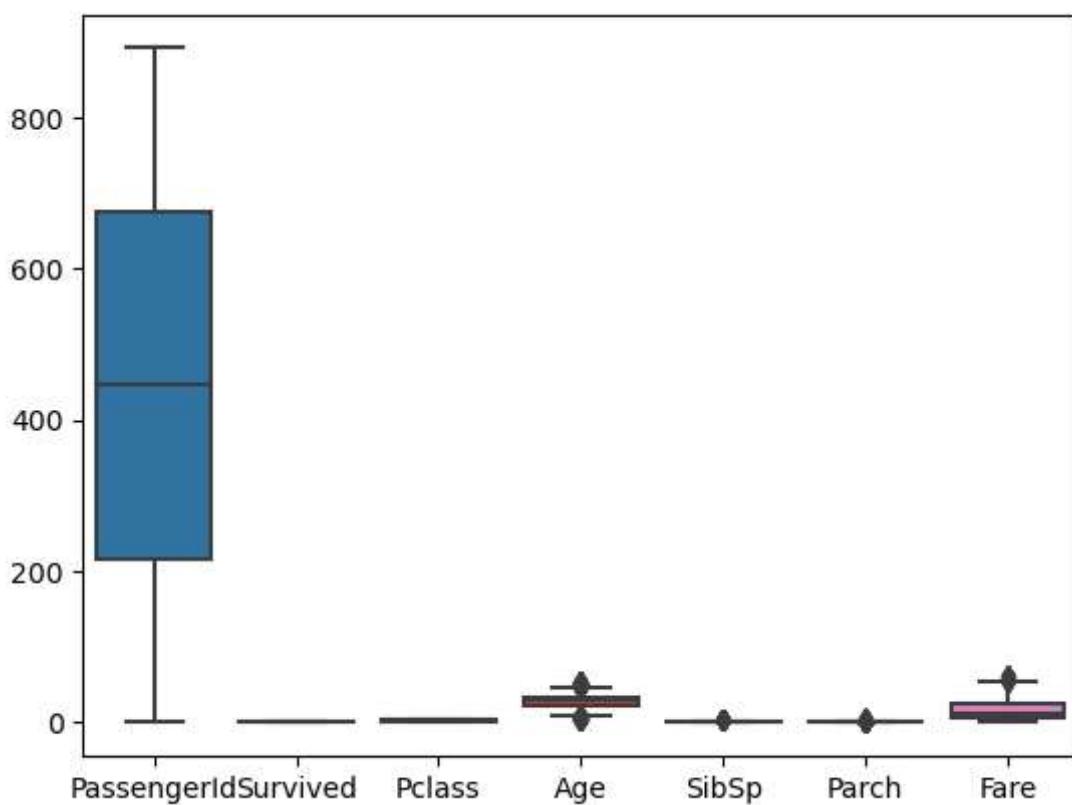
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	BS B5
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	BS B5
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	C12
4	5	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500	BS B5
5	6	0	3	Moran, Mr. James	male	29.699118	0	0	330877	8.4583	BS B5

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
...
886	887	0	2	Montvila, Rev. Juozas	male	27.000000	0	0	211536	13.0000	B5 B5
887	888	1	1	Graham, Miss. Margaret Edith	female	19.000000	0	0	112053	30.0000	B4
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	29.699118	1	2	W./C. 6607	23.4500	B5 B5
889	890	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	111369	30.0000	C14
890	891	0	3	Dooley, Mr. Patrick	male	32.000000	0	0	370376	7.7500	B5 B5

741 rows × 12 columns

In [79]: `sns.boxplot(df_cleaned)`

Out[79]: <Axes: >



Separating dependent and independent variables

```
In [30]:  
x = df.iloc[:, [2, 4, 5, 6, 7, 9, 11]]  
y = df.iloc[:, 1]
```

```
In [31]:  
x.head()
```

```
Out[31]:  


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


```

```
In [32]:  
y.head()
```

```
Out[32]:  
0    0  
1    1  
2    1  
3    1  
4    0  
Name: Survived, dtype: int64
```

```
In [33]:  
df.shape
```

```
Out[33]: (849, 12)
```

```
In [34]:  
x.shape
```

```
Out[34]: (849, 7)
```

```
In [35]:  
y.shape
```

```
Out[35]: (849,)
```

One hot encoding on Embarked column

```
In [36]:  
x.Embarked.value_counts()
```

```
Out[36]: S    618  
C    157  
Q    74  
Name: Embarked, dtype: int64
```

```
In [37]: embarked=pd.get_dummies(x["Embarked"],drop_first=True)
```

```
In [38]: embarked
```

```
Out[38]:
```

	Q	S
0	0	1
1	0	0
2	0	1
3	0	1
4	0	1
...
886	0	1
887	0	1
888	0	1
889	0	0
890	1	0

849 rows × 2 columns

```
In [39]: x=pd.concat([x,embarked],axis=1)
```

```
In [40]: x.head()
```

```
Out[40]:
```

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

```
In [41]: x.drop(["Embarked"],axis=1,inplace=True)
```

```
In [42]: x.head(10)
```

```
Out[42]:
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Q	S
0	3	male	22.000000	1	0	7.2500	0	1

Pclass	Sex	Age	SibSp	Parch	Fare	Q	S
1	female	38.000000	1	0	71.2833	0	0
2	female	26.000000	0	0	7.9250	0	1
3	female	35.000000	1	0	53.1000	0	1
4	male	35.000000	0	0	8.0500	0	1
5	male	29.699118	0	0	8.4583	1	0
6	male	54.000000	0	0	51.8625	0	1
7	male	29.690000	3	1	21.0750	0	1
8	female	27.000000	0	2	11.1333	0	1
9	female	14.000000	1	0	30.0708	0	0

In [43]: `x.shape`

Out[43]: `(849, 8)`

Label Encoding on sex column

In [44]: `from sklearn.preprocessing import LabelEncoder`

In [45]: `le=LabelEncoder()`

In [46]: `x["Sex"] = le.fit_transform(x["Sex"])`

In [47]: `x["Sex"]`

Out[47]:

0	1
1	0
2	0
3	0
4	1
	..
886	1
887	0
888	0
889	1
890	1

Name: Sex, Length: 849, dtype: int32

In [48]: `x["Sex"].value_counts()`

Out[48]:

1	545
0	304

Name: Sex, dtype: int64

```
In [49]: x["Sex"].nunique()
```

```
Out[49]: 2
```

```
In [50]: x.head()
```

```
Out[50]:
```

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

```
In [51]: x.Sex.value_counts()
```

```
Out[51]:
```

1	545
0	304

Name: Sex, dtype: int64

Splitting into training and testing set

```
In [52]: 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 [53]: x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
Out[53]: ((594, 8), (255, 8), (594,), (255,))
```

Feature Scaling

```
In [54]: from sklearn.preprocessing import StandardScaler  
sc=StandardScaler()
```

```
In [55]: x_train=sc.fit_transform(x_train)  
x_test=sc.fit_transform(x_test)
```

```
In [56]: x_train
```

```
Out[56]: array([[-0.41834557, -1.31330524,  1.17453394, ..., -0.24582194,  
                 -0.32262861,  0.62017367],  
                [ 0.79383622, -1.31330524,  0.0443624 , ..., -0.49093682,  
                 -0.32262861, -1.61245155],  
                [ 0.79383622,  0.76143761,  0.0443624 , ..., -0.48045329,
```

```
 3.09953914, -1.61245155],  
...,  
[ 0.79383622,  0.76143761,  0.67449708, ..., -0.48552931,  
-0.32262861,  0.62017367],  
[ 0.79383622,  0.76143761, -0.72560612, ..., -0.4744627 ,  
-0.32262861,  0.62017367],  
[ 0.79383622,  0.76143761, -1.22564298, ..., -0.49434746,  
-0.32262861,  0.62017367]])
```