In [1]:

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

In [38]:

```python
data=pd.read_csv("Titanic-Dataset.csv")
```

In [39]:

```python
data.head()
```

Out[39]:

	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 [40]:

```
data.tail()
```

Out[40]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	C
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	(
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	

In [41]:

```
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 [42]:

```python
data.describe()
```

Out[42]:

	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

Handling Null Values

In [43]:

```python
data.isnull().any()
```

Out[43]:

```
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 [44]:

```python
data.isnull().sum()
```

Out[44]:

```
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 [45]:

```python
mean=data["Age"].mean()
```

Filling the null values in Age column with Mean

In [46]:

```python
data["Age"]=data["Age"].fillna(mean)
```

In [47]:

```python
data["Age"].tail()
```

Out[47]:

```
886    27.000000
887    19.000000
888    29.699118
889    26.000000
890    32.000000
Name: Age, dtype: float64
```

In [48]:

```python
data["Age"].isnull().sum()
```

Out[48]:

```
0
```

Filling the Null values in Embarked with mode

In [49]:

```python
Em_mode=data["Embarked"].mode()
```

In [50]:

```python
data["Embarked"]=data["Embarked"].fillna(Em_mode[0])
```

In [51]:

```python
data["Embarked"].isnull().sum()
```

Out[51]:

```
0
```

In [ ]:

Filling the null values in Cabin with mode

In [52]:

```python
Cabin_mode=data["Cabin"].mode()
```

In [53]:

```python
data["Cabin"]
```

Out[53]:

```
0        NaN
1        C85
2        NaN
3       C123
4        NaN
        ...
886      NaN
887      B42
888      NaN
889     C148
890      NaN
Name: Cabin, Length: 891, dtype: object
```

In [54]:

```python
Cabin_mode
```

Out[54]:

```
0        B96 B98
1    C23 C25 C27
2             G6
Name: Cabin, dtype: object
```

In [55]:

```python
data["Cabin"]=data["Cabin"].fillna(Cabin_mode[2])
```

In [56]:

```python
data["Cabin"].isnull().sum()
```

Out[56]:

```
0
```

In [57]:

```python
data["Cabin"]
```

Out[57]:

```
0        G6
1       C85
2        G6
3      C123
4        G6
       ...
886      G6
887     B42
888      G6
889    C148
890      G6
Name: Cabin, Length: 891, dtype: object
```

In [58]:

```python
data.isnull().sum()
```

Out[58]:

```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
```

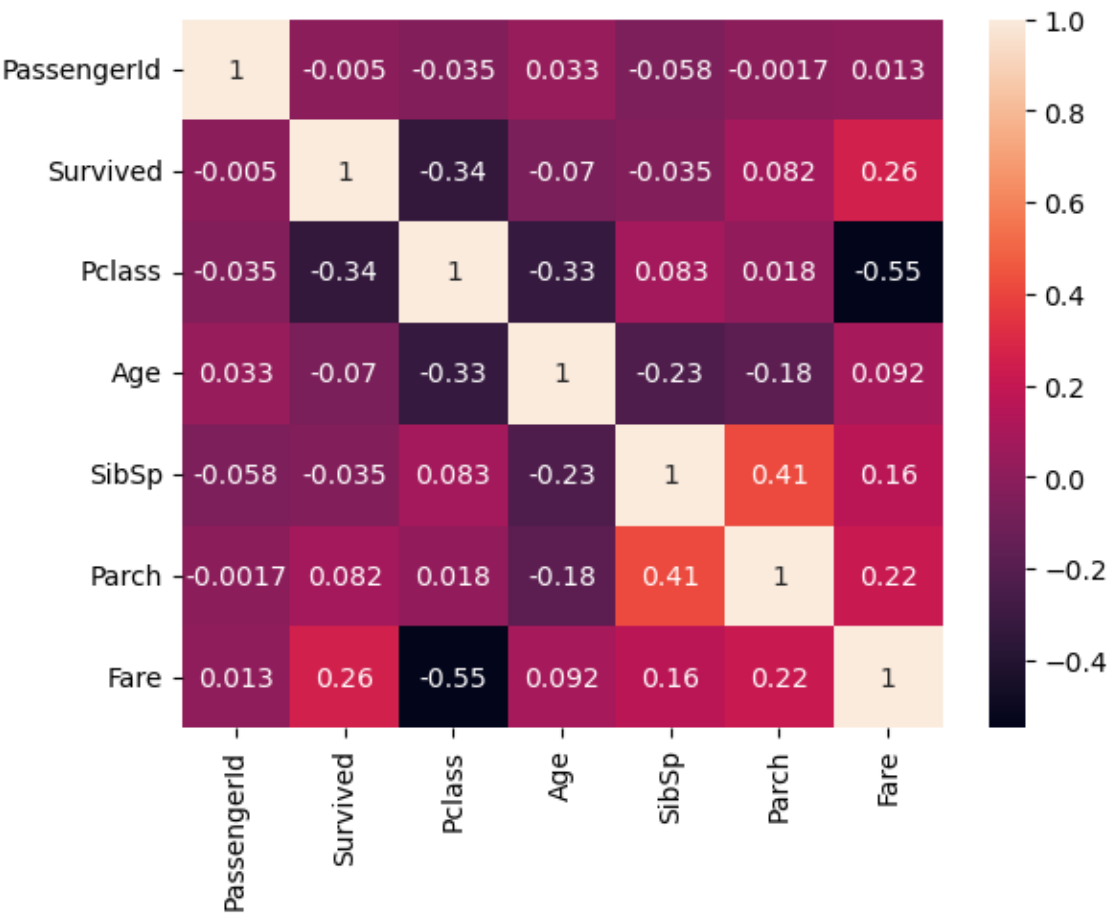# Data Visualisation

In [59]:

```python
cor=data.corr()
```

```
C:\Users\pichi\AppData\Local\Temp\ipykernel_25480\1426905697.py:1: FutureW
arning: The default value of numeric_only in DataFrame.corr is deprecated.
In a future version, it will default to False. Select only valid columns o
r specify the value of numeric_only to silence this warning.
  cor=data.corr()
```

In [60]:

```python
sns.heatmap(cor,annot=True)
```
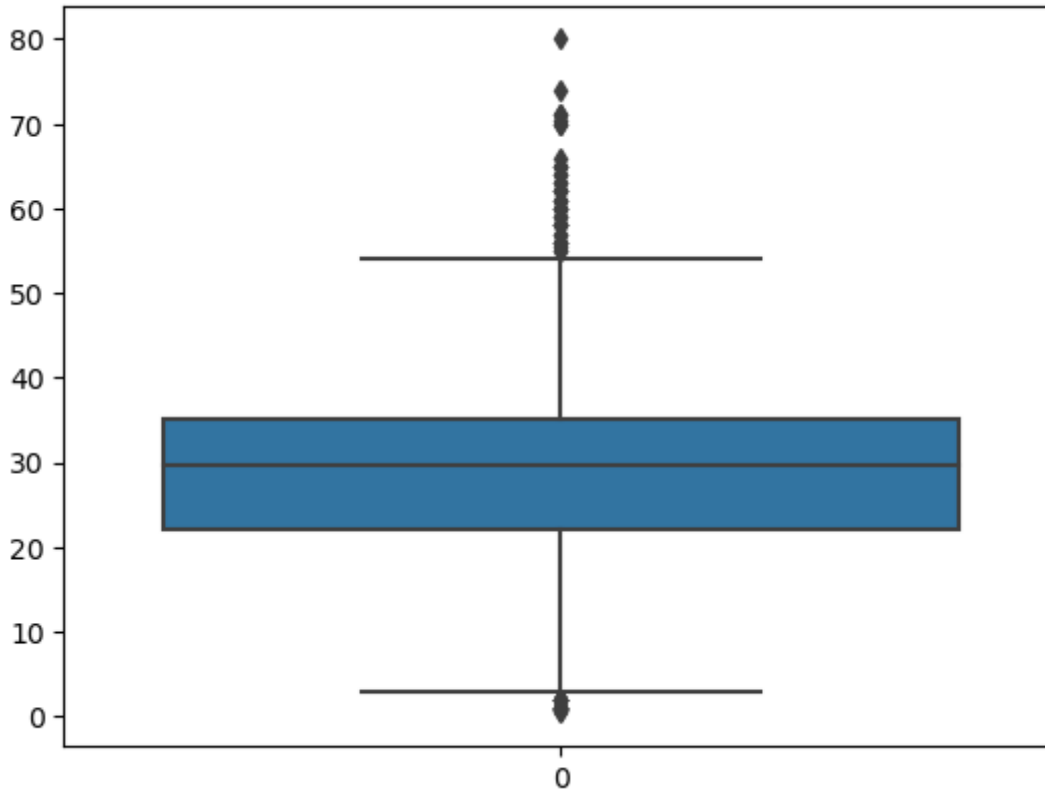
Out[60]:

```
<Axes: >
```

Handling the outliers

In [61]:

```
sns.boxplot(data["Age"])
```

Out[61]:

<Axes: >

Outliers

In [62]:

```
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 [63]:

```
IQR_Age=Age_q3-Age_q1
IQR_Age
```

Out[63]:

13.0

In [64]:

```python
upperlimit_Age=Age_q3+1.5*IQR_Age
upperlimit_Age
```

Out[64]:

54.5

In [65]:

```python
lower_limit_Age = Age_q1-1.5*IQR_Age
lower_limit_Age
```

Out[65]:

2.5

In [66]:

```python
median_Age=data["Age"].median()
median_Age
```

Out[66]:

29.69911764705882

In [67]:

```python
data["Age"]=np.where(data["Age"]>upperlimit_Age,median_Age,data["Age"])
```
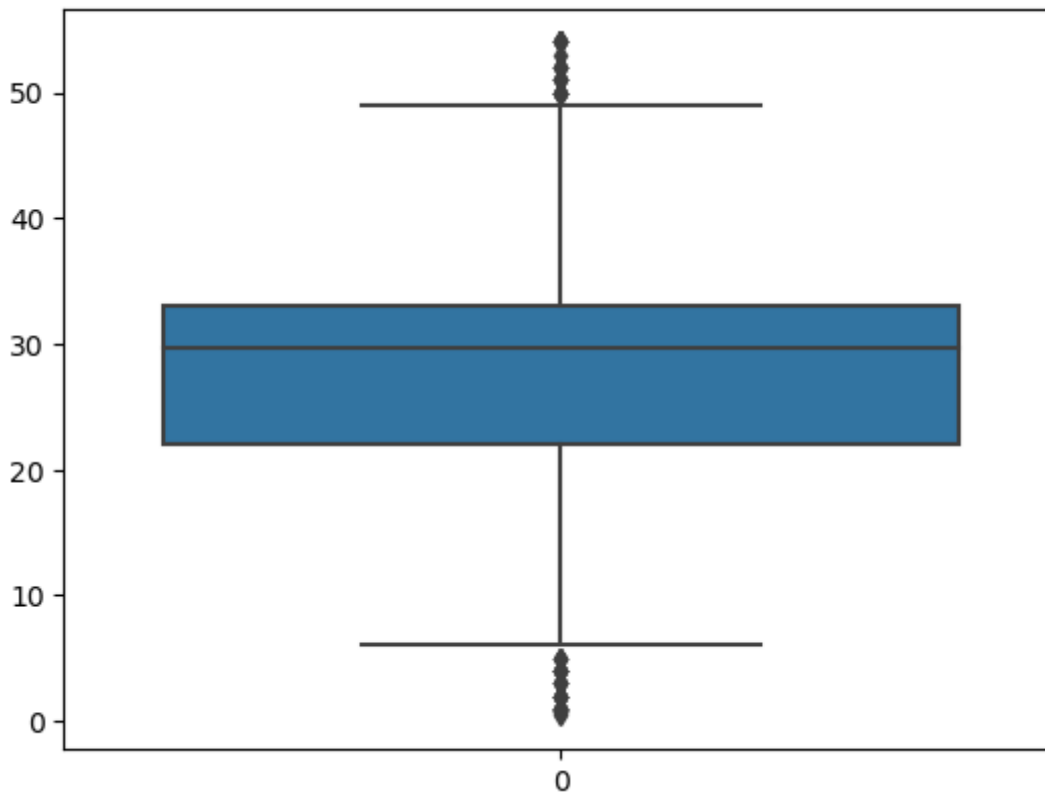
In [68]:

```python
(data["Age"]>54.5).sum()
```

Out[68]:

0

In [69]:
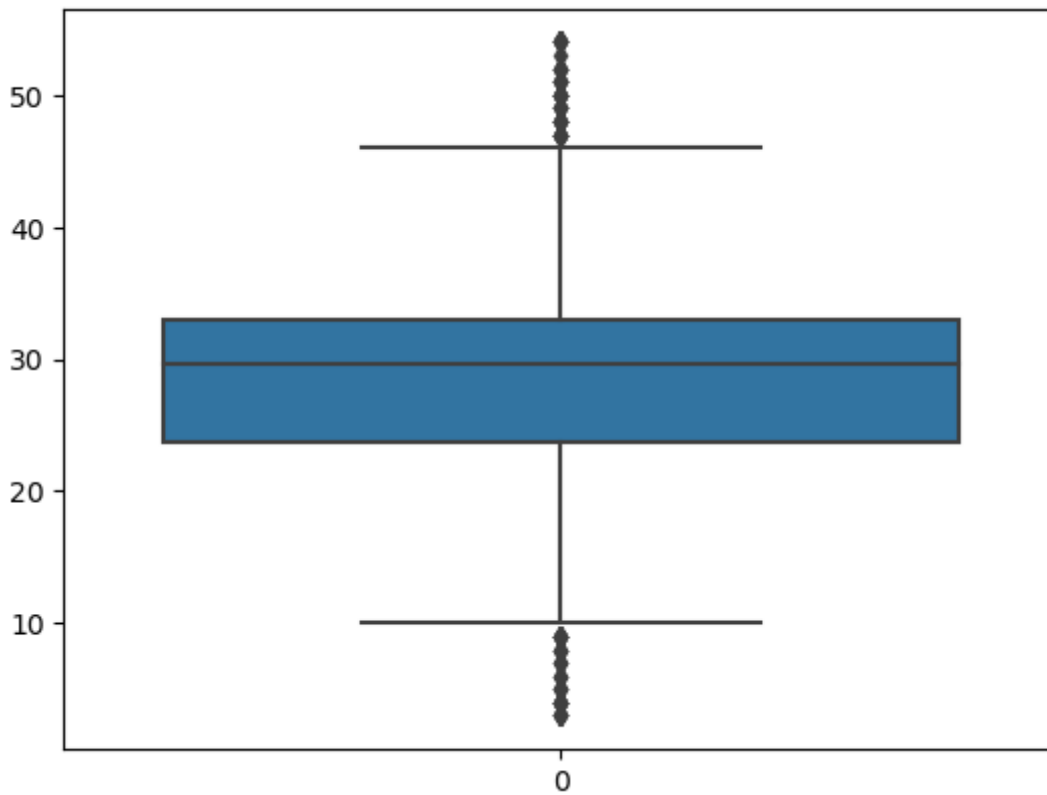
```python
sns.boxplot(data["Age"])
```

Out[69]:

```
<Axes: >
```


In [70]:

```python
data["Age"]=np.where(data["Age"]<lower_limit_Age,median_Age,data["Age"])
```

In [71]:

```python
sns.boxplot(data["Age"])
```

Out[71]:

```
<Axes: >
```
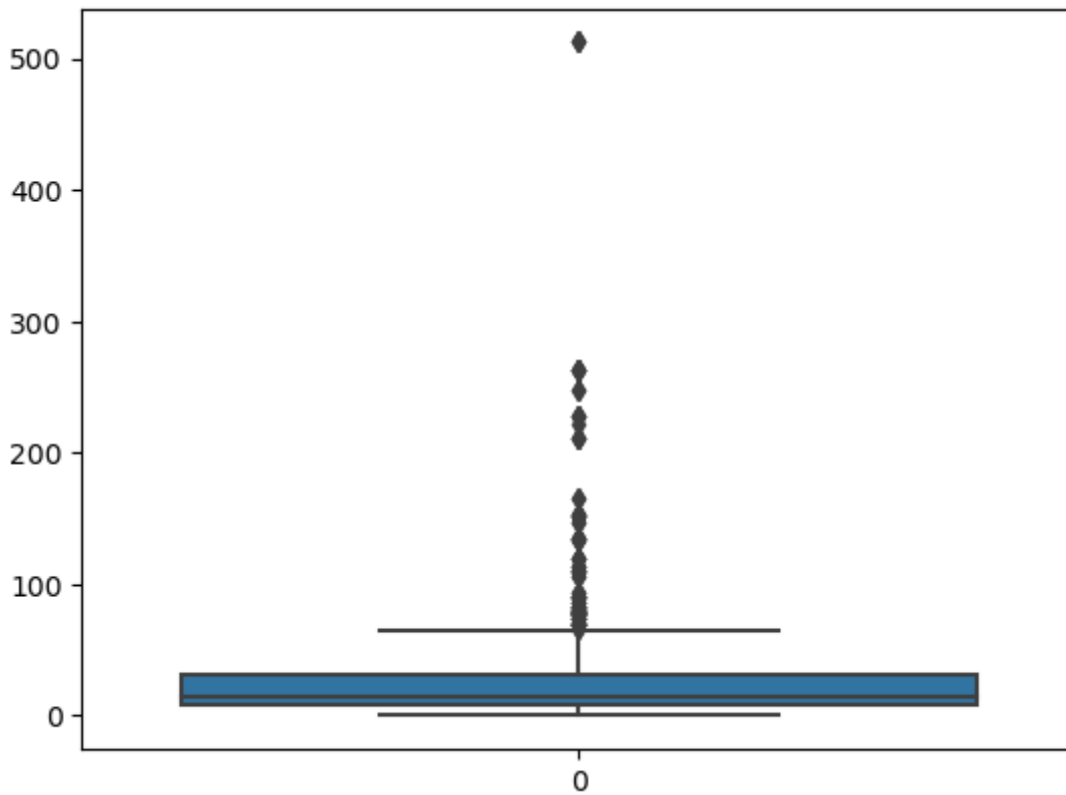
In [72]:

```
sns.boxplot(data["Fare"])
```

Out[72]:

```
<Axes: >
```


In [73]:

```
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 [74]:

```
IQR_Fare=Fare_q3-Fare_q1
IQR_Fare
```

Out[74]:

```
23.0896
```

In [75]:

```
upperlimit_Fare=Fare_q3+1.5*IQR_Fare
upperlimit_Fare
```

Out[75]:

```
65.6344
```

In [76]:

```python
lower_limit_Fare = Fare_q1-1.5*IQR_Fare
lower_limit_Fare
```

Out[76]:

-26.724

In [77]:

```python
median_Fare=data["Fare"].median()
median_Fare
```
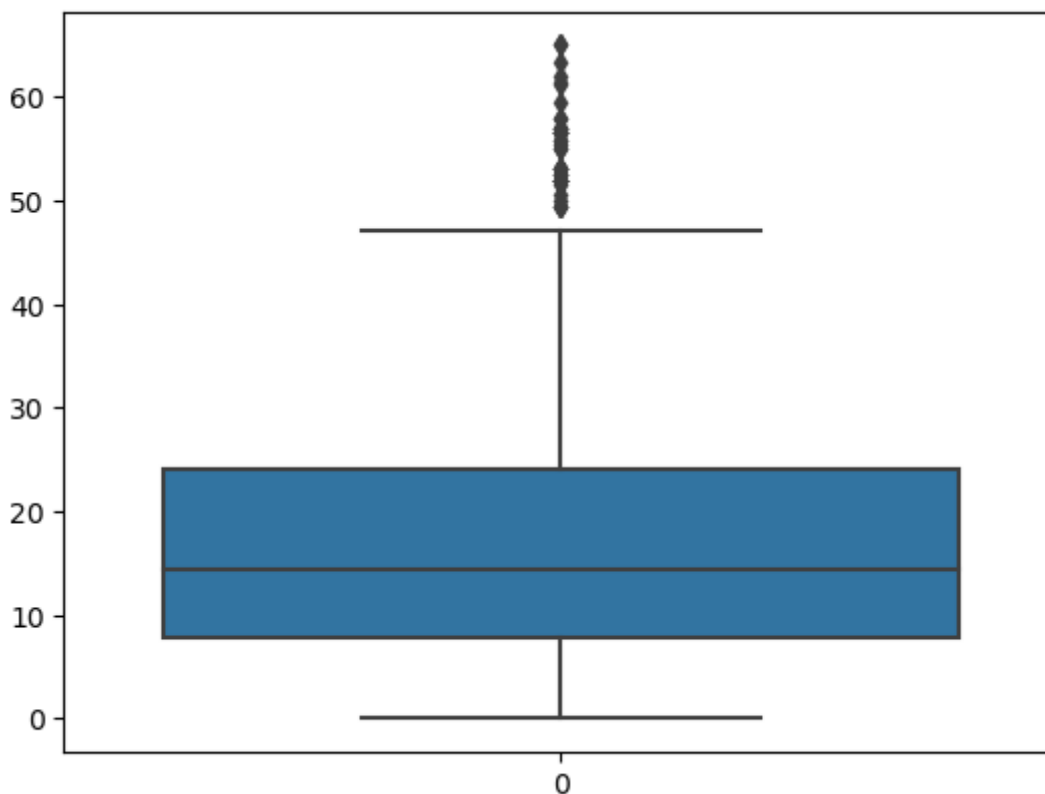
Out[77]:

14.4542

In [78]:

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

In [79]:

```python
sns.boxplot(data["Fare"])
```

Out[79]:

<Axes: >

In [80]:

```python
(data["Fare"]>65).sum()
```

Out[80]:

0

dropping the variables

In [84]:

```python
data.drop(['Name'],axis=1,inplace=True)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[84], line 1
----> 1 data.drop(['Name'],axis=1,inplace=True)
      2 data

File C:\ProgramData\anaconda3\lib\site-packages\pandas\util\_decorators.py:331, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    325 if len(args) > num_allow_args:
    326     warnings.warn(
    327         msg.format(arguments=_format_argument_list(allow_args)),
    328         FutureWarning,
    329         stacklevel=find_stack_level(),
    330     )
--> 331 return func(*args, **kwargs)

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\frame.py:5399, in DataFrame.drop(self, labels, axis, index, columns, level, inplace, errors)
    5251 @deprecate_nonkeyword_arguments(version=None, allowed_args=["self", "labels"])
    5252 def drop(   # type: ignore[override]
    5253     self,
    (...)
    5260     errors: IgnoreRaise = "raise",
    5261 ) -> DataFrame | None:
    5262     """
    5263     Drop specified labels from rows or columns.
    5264
    (...)
    5397             weight  1.0    0.8
    5398     """
-> 5399     return super().drop(
    5400         labels=labels,
    5401         axis=axis,
    5402         index=index,
    5403         columns=columns,
    5404         level=level,
    5405         inplace=inplace,
    5406         errors=errors,
    5407     )

File C:\ProgramData\anaconda3\lib\site-packages\pandas\util\_decorators.py:331, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    325 if len(args) > num_allow_args:
    326     warnings.warn(
    327         msg.format(arguments=_format_argument_list(allow_args)),
    328         FutureWarning,
    329         stacklevel=find_stack_level(),
    330     )
--> 331 return func(*args, **kwargs)

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\generic.py:4505, in NDFrame.drop(self, labels, axis, index, columns, level, inplace, errors)
    4503 for axis, labels in axes.items():
    4504     if labels is not None:
```

```
-> 4505          obj = obj._drop_axis(labels, axis, level=level, errors=err
ors)
   4507 if inplace:
   4508     self._update_inplace(obj)

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\generic.py:454
6, in NDFrame._drop_axis(self, labels, axis, level, errors, only_slice)
   4544          new_axis = axis.drop(labels, level=level, errors=errors)
   4545     else:
-> 4546          new_axis = axis.drop(labels, errors=errors)
   4547     indexer = axis.get_indexer(new_axis)
   4549 # Case for non-unique axis
   4550 else:

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\indexes\base.p
y:6934, in Index.drop(self, labels, errors)
   6932 if mask.any():
   6933     if errors != "ignore":
-> 6934          raise KeyError(f"{list(labels[mask])} not found in axis")
   6935     indexer = indexer[~mask]
   6936 return self.delete(indexer)

KeyError: "['Name'] not found in axis"
```

In [85]:

```
data
```

Out[85]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	male	22.000000	1	0	A/5 21171	7.2500	
1	2	1	1	female	38.000000	1	0	PC 17599	14.4542	
2	3	1	3	female	26.000000	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	female	35.000000	1	0	113803	53.1000	C
4	5	0	3	male	35.000000	0	0	373450	8.0500	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	male	27.000000	0	0	211536	13.0000	
887	888	1	1	female	19.000000	0	0	112053	30.0000	
888	889	0	3	female	29.699118	1	2	W./C. 6607	23.4500	
889	890	1	1	male	26.000000	0	0	111369	30.0000	C
890	891	0	3	male	32.000000	0	0	370376	7.7500	

891 rows × 11 columns

In [86]:

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

In [87]:

```python
data
```

Out[87]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Emba
0	1	0	3	male	22.000000	1	0	7.2500	G6	
1	2	1	1	female	38.000000	1	0	14.4542	C85	
2	3	1	3	female	26.000000	0	0	7.9250	G6	
3	4	1	1	female	35.000000	1	0	53.1000	C123	
4	5	0	3	male	35.000000	0	0	8.0500	G6	
...	...	...	...	...	...	...	...	...	...	
886	887	0	2	male	27.000000	0	0	13.0000	G6	
887	888	1	1	female	19.000000	0	0	30.0000	B42	
888	889	0	3	female	29.699118	1	2	23.4500	G6	
889	890	1	1	male	26.000000	0	0	30.0000	C148	
890	891	0	3	male	32.000000	0	0	7.7500	G6	

891 rows × 10 columns

In [88]:

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

In [89]:

```
data
```

Out[89]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	0	3	male	22.000000	1	0	7.2500	G6	S
1	1	1	female	38.000000	1	0	14.4542	C85	C
2	1	3	female	26.000000	0	0	7.9250	G6	S
3	1	1	female	35.000000	1	0	53.1000	C123	S
4	0	3	male	35.000000	0	0	8.0500	G6	S
...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.000000	0	0	13.0000	G6	S
887	1	1	female	19.000000	0	0	30.0000	B42	S
888	0	3	female	29.699118	1	2	23.4500	G6	S
889	1	1	male	26.000000	0	0	30.0000	C148	C
890	0	3	male	32.000000	0	0	7.7500	G6	Q

891 rows × 9 columns

In [90]:

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

In [91]:

```
data
```

Out[91]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
1	1	1	female	38.000000	1	0	14.4542	C
2	1	3	female	26.000000	0	0	7.9250	S
3	1	1	female	35.000000	1	0	53.1000	S
4	0	3	male	35.000000	0	0	8.0500	S
...	...	...	...	...	...	...	...	...
886	0	2	male	27.000000	0	0	13.0000	S
887	1	1	female	19.000000	0	0	30.0000	S
888	0	3	female	29.699118	1	2	23.4500	S
889	1	1	male	26.000000	0	0	30.0000	C
890	0	3	male	32.000000	0	0	7.7500	Q

891 rows × 8 columns

Splitting the data

In [92]:

```python
y=data["Survived"]
```

In [93]:

```python
y.head()
```

Out[93]:

```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [100]:

```python
data
```

Out[100]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	22.000000	1	0	7.2500	S
1	1	female	38.000000	1	0	14.4542	C
2	3	female	26.000000	0	0	7.9250	S
3	1	female	35.000000	1	0	53.1000	S
4	3	male	35.000000	0	0	8.0500	S
...	...	...	...	...	...	...	...
886	2	male	27.000000	0	0	13.0000	S
887	1	female	19.000000	0	0	30.0000	S
888	3	female	29.699118	1	2	23.4500	S
889	1	male	26.000000	0	0	30.0000	C
890	3	male	32.000000	0	0	7.7500	Q

891 rows × 7 columns

Encoding

In [101]:

```python
from sklearn.preprocessing import LabelEncoder
```

In [102]:

```python
le=LabelEncoder()
```

In [103]:

```python
data["Sex"]=le.fit_transform(data["Sex"])
```

In [104]:

```python
data["Sex"]
```

Out[104]:

```
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 [105]:

```python
data.head()
```

Out[105]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	S
1	1	0	38.0	1	0	14.4542	C
2	3	0	26.0	0	0	7.9250	S
3	1	0	35.0	1	0	53.1000	S
4	3	1	35.0	0	0	8.0500	S

In [106]:

```python
data["Embarked"]=le.fit_transform(data["Embarked"])
```

In [107]:

```python
data.head()
```

Out[107]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	2
1	1	0	38.0	1	0	14.4542	0
2	3	0	26.0	0	0	7.9250	2
3	1	0	35.0	1	0	53.1000	2
4	3	1	35.0	0	0	8.0500	2

In [108]:

```python
data["Pclass"].nunique()
```

Out[108]:

3

In [109]:

```python
data["Pclass"].unique()
```

Out[109]:

array([3, 1, 2], dtype=int64)

In [112]:

```python
data["Sex"].unique()
```

Out[112]:

array([1, 0])

In [113]:

```python
data["Embarked"].unique()
```

Out[113]:

array([2, 0, 1])

Spliting the train and test data

In [114]:

```python
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_state=0)
```

In [119]:

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[119]:

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

Feature Scaling

In [123]:

```python
from sklearn.preprocessing import StandardScaler
```

In [124]:

```python
sc=StandardScaler()
```

In [125]:

```python
x_train=sc.fit_transform(x_train)
```

In [126]:

```python
x_train
```

Out[126]:

```
array([[-1.5325562 ,  0.72592065,  2.25384386, ..., -0.47299765,
         0.67925137,  0.56710989],
       [-1.5325562 , -1.37756104,  2.04763778, ..., -0.47299765,
        -0.26059483, -2.03075381],
       [ 0.84844757,  0.72592065,  0.05765817, ...,  1.93253327,
         2.26045064,  0.56710989],
       ...,
       [ 0.84844757,  0.72592065,  0.05765817, ..., -0.47299765,
        -0.78281017, -0.73182196],
       [ 0.84844757, -1.37756104,  0.70729828, ..., -0.47299765,
        -0.03170555,  0.56710989],
       [-0.34205431,  0.72592065,  0.05765817, ...,  0.72976781,
         1.64661898,  0.56710989]])
```

In [127]:

```python
x_test=sc.fit_transform(x_test)
```

In [128]:

```
x_test
```

Out[128]:

```
array([[ 0.77963055,  0.76537495,  0.12227065, ..., -0.47809977,
        -0.15813988, -1.76531134],
       [ 0.77963055,  0.76537495,  0.12227065, ..., -0.47809977,
        -0.72165412,  0.63014911],
       [ 0.77963055,  0.76537495, -2.14803266, ...,  0.87064484,
         1.03823178, -0.56758111],
       ...,
       [ 0.77963055,  0.76537495, -0.14768767, ..., -0.47809977,
        -0.15847431, -1.76531134],
       [ 0.77963055, -1.30654916, -1.04784292, ..., -0.47809977,
        -0.72607524,  0.63014911],
       [-1.64991582,  0.76537495,  0.12227065, ..., -0.47809977,
         0.92369033, -1.76531134]])
```

In [ ]:

In [ ]: