Assignment 15 sep

Perform Data preprocessing on Titanic dataset 1.Data Collection. Please download the dataset from https://www.kaggle.com/datasets/yasserh/titanic-dataset (https://www.kaggle.com/datasets/yasserh/titanic-dataset)

2.Data Preprocessing o Import the Libraries. o Importing the dataset. o Checking for Null Values. o Data Visualization. o Outlier Detection o Splitting Dependent and Independent variables o Perform Encoding o Feature Scaling. o Splitting Data into Train and Test

Manikanta Seepana 21BCE7217

o Import the Libraries.

In [1]:

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

o Importing the dataset.

In [2]:

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

In [3]:
```
df
```

Out[3]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

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.head()
```

Out[5]:

	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 [6]:
```
df.shape
```

Out[6]:
```
(891, 12)
```

In [7]:
```
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 [8]:

```
df.corr()
```

```
C:\Users\seepana manikanta\AppData\Local\Temp\ipykernel_22820\1134722465.py:1: FutureWarning: The default value of n
umeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid column
s or specify the value of numeric_only to silence this warning.
  df.corr()
```

Out[8]:

	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

o Checking for Null Values

In [9]:

```
df.isnull().any()
```

Out[9]:

```
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 [10]:

```
df.isnull().sum()
```

Out[10]:

```
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 [11]:

```
type(df["Age"])
```

Out[11]:

```
pandas.core.series.Series
```

In [12]:

```
df["Age"].fillna(df["Age"].mean(),inplace=True)
```

In [13]:

```
df.isnull().any()
```

Out[13]:

```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age            False
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin           True
Embarked        True
dtype: bool
```

In [14]:

```
df.Embarked.nunique()
```

Out[14]:

```
3
```

In [15]:

```
df.Embarked.unique()
```

Out[15]:

```
array(['S', 'C', 'Q', nan], dtype=object)
```

In [16]:

```
df.Embarked.value_counts()
```

Out[16]:

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

In [17]:

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

In [18]:

```
df.Embarked.value_counts()
```

Out[18]:

```
S    646
C    168
Q     77
Name: Embarked, dtype: int64
```

In [19]:

```
df.isnull().any()
```

Out[19]:

```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age            False
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin           True
Embarked       False
dtype: bool
```

In [20]:

```
df=df.drop(columns=["Cabin"],axis=1)
```

In [21]:

```
df.isnull().any()
```

Out[21]:

```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age            False
SibSp          False
Parch          False
Ticket         False
Fare           False
Embarked       False
dtype: bool
```

o Data Visualization.

In [22]:

```
df.head()
```

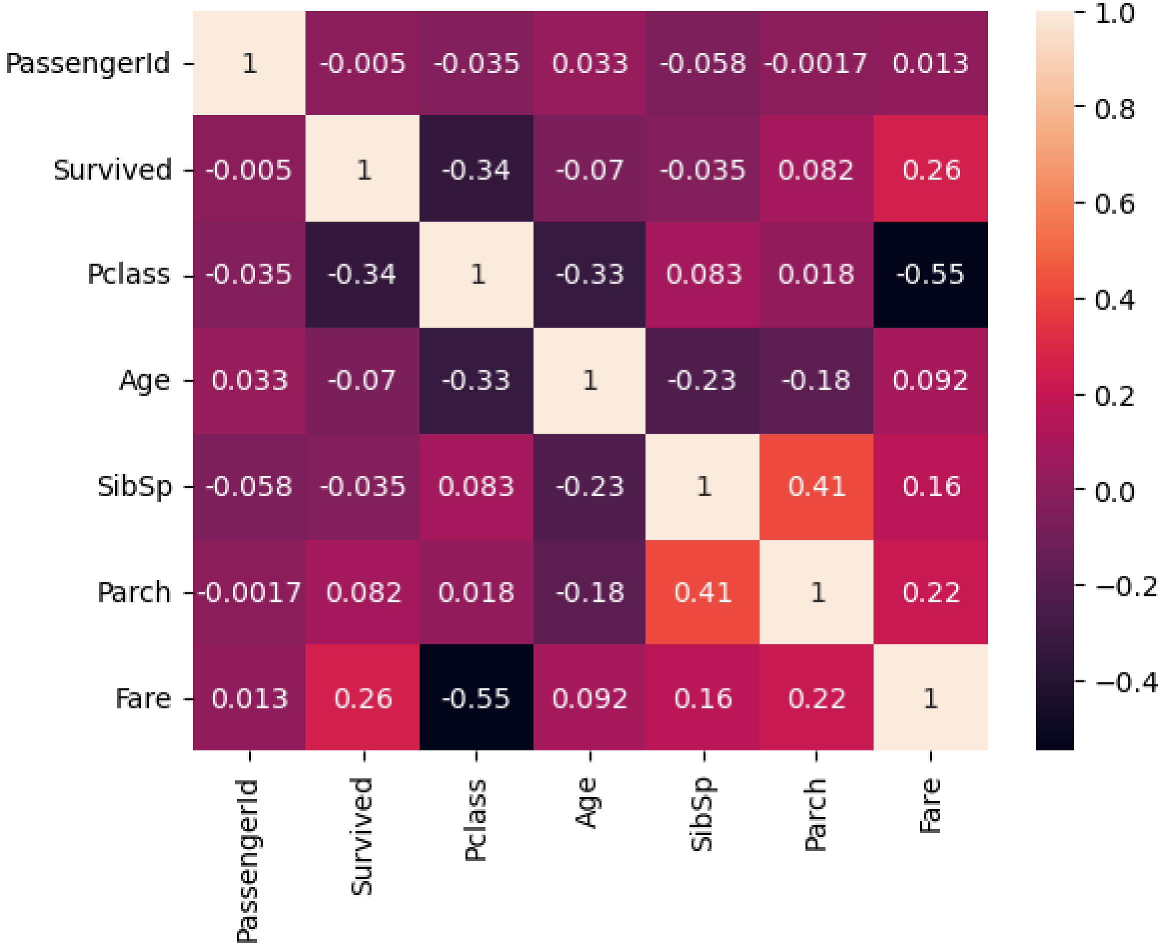
Out[22]:

	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	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S

In [23]:
```
sns.heatmap(df.corr(),annot=True)
```

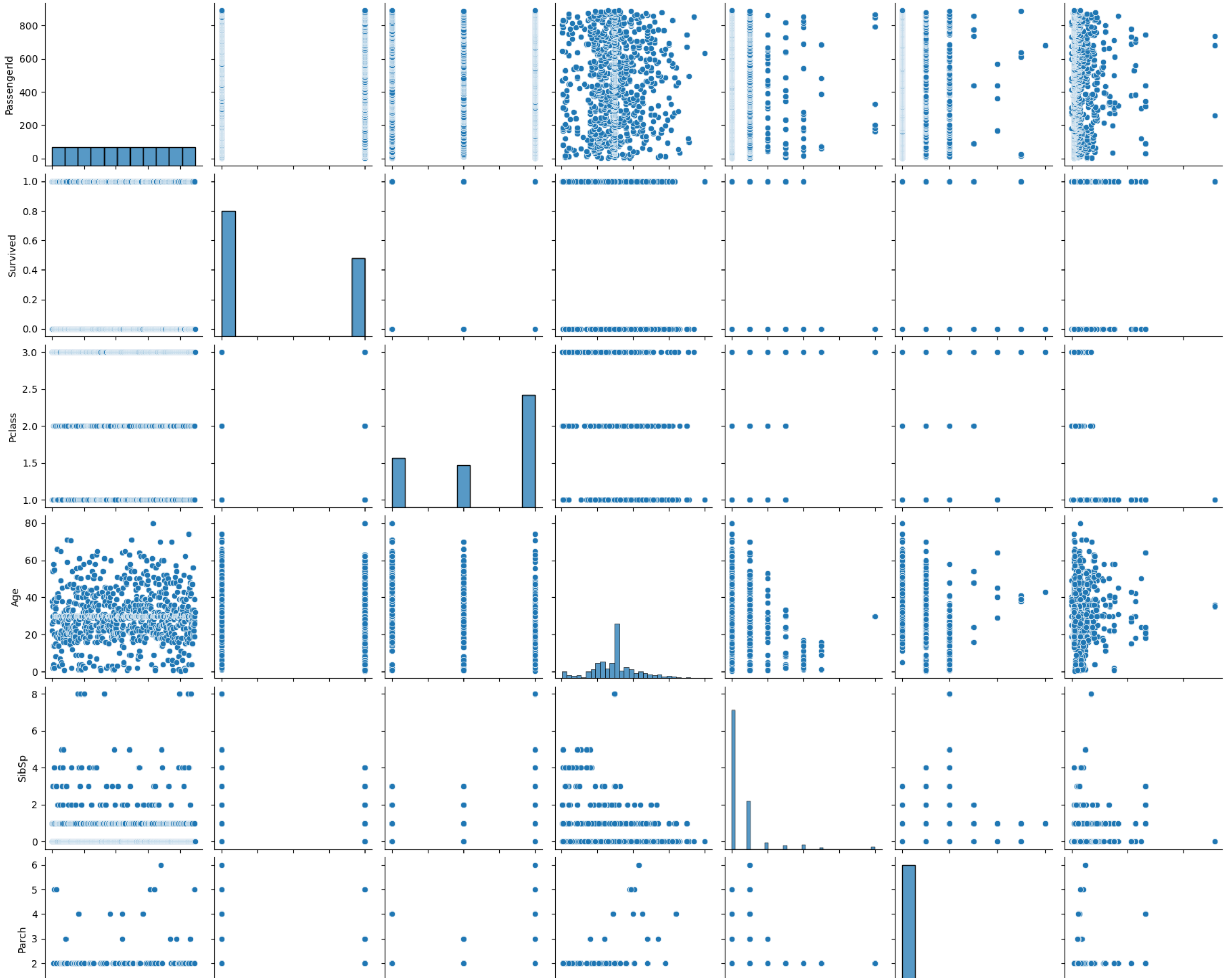
```
C:\Users\seepana manikanta\AppData\Local\Temp\ipykernel_22820\4277794465.py:1: FutureWarning: The default value of n
umeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid column
s or specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr(),annot=True)
```

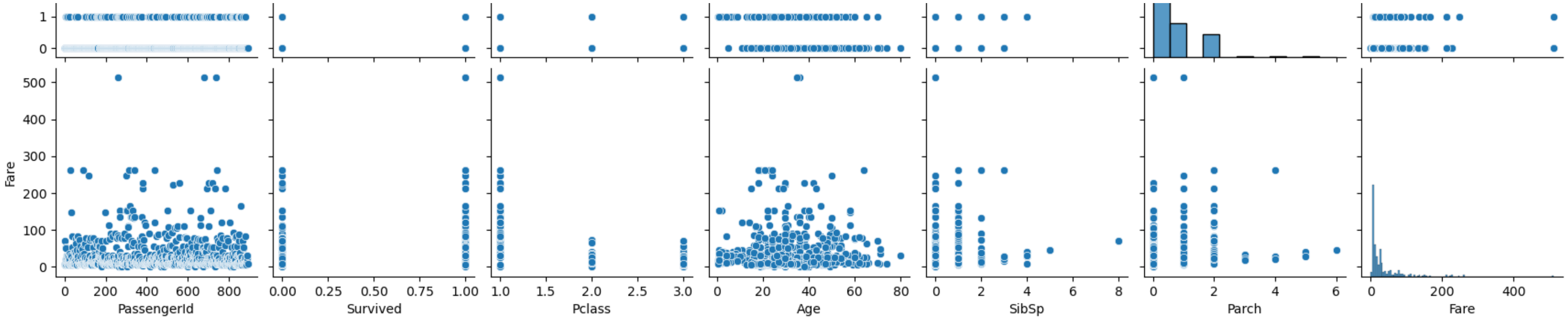
Out[23]:
```
<Axes: >
```



In [24]:
```
sns.pairplot(df)
```

Out[24]:
```
<seaborn.axisgrid.PairGrid at 0x1caac7d9c50>
```


PassengerId
800
600
400
200
0
Survived
1.0
0.8
0.6
0.4
0.2
0.0
Pclass
3.0
2.5
2.0
1.5
1.0
Age
80
60
40
20
0
SibSp
8
6
4
2
0
Parch
6
5
4
3
2

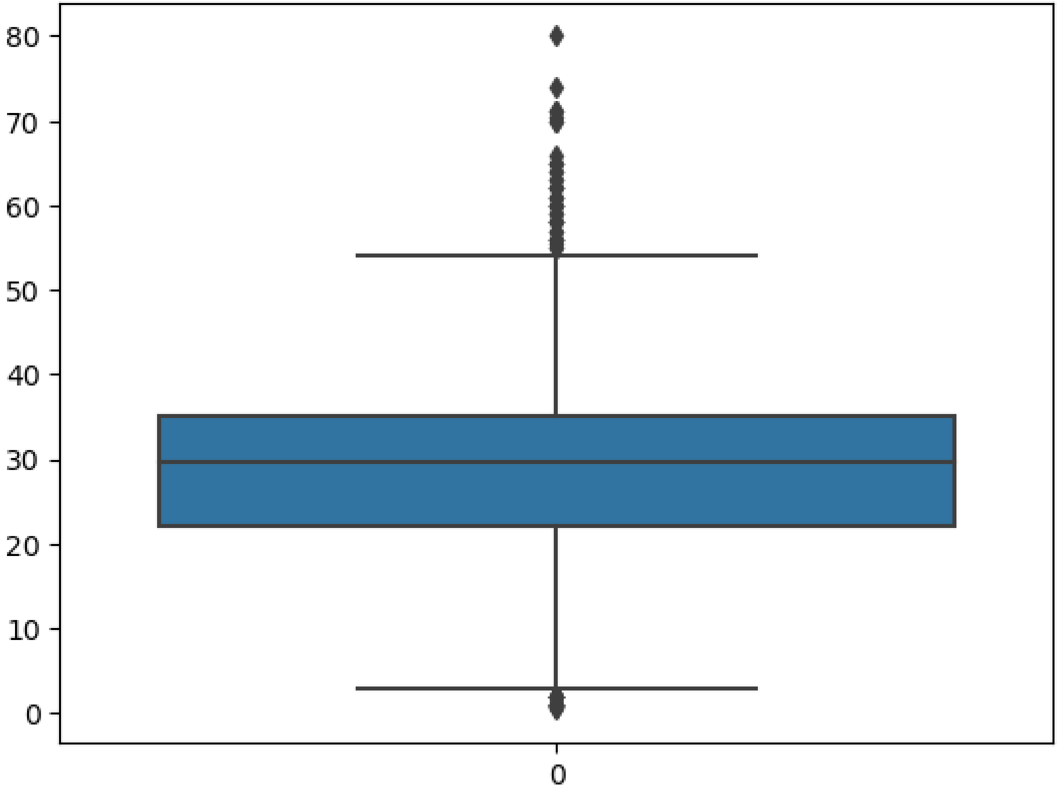


In [25]:

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

Out[25]:

```
<Axes: >
```



In [26]:

```
df.head()
```

Out[26]:

	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	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S

In [27]:

```
df.describe()
```

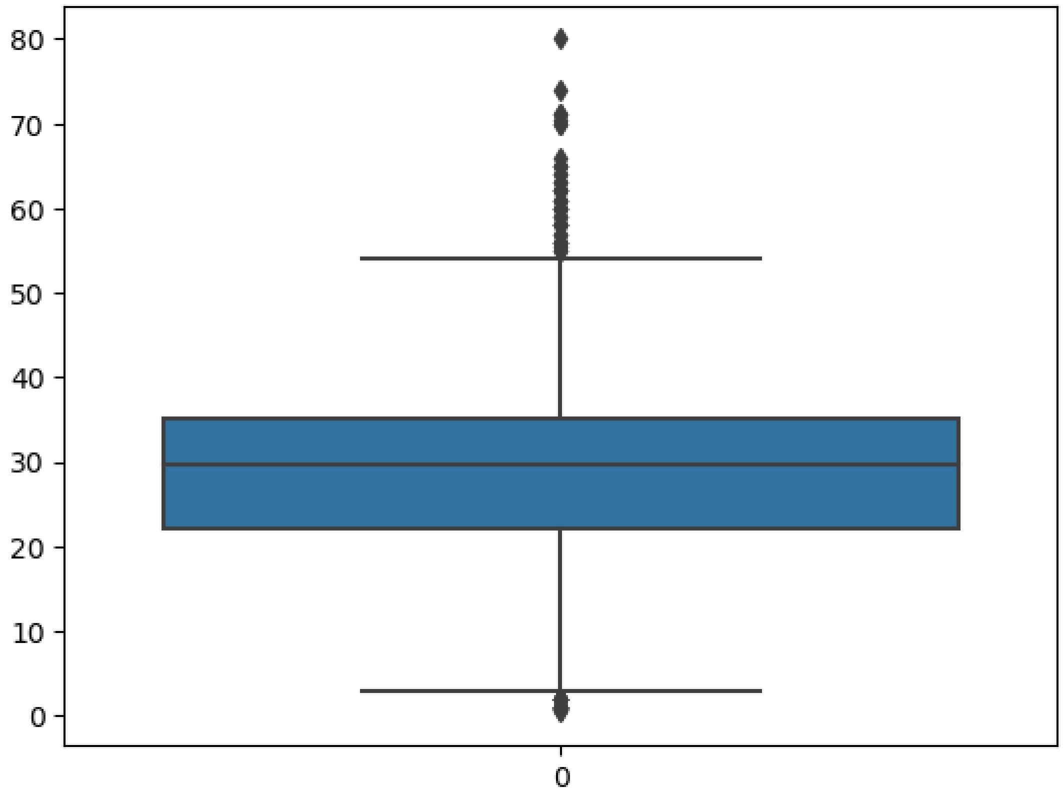
Out[27]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	891.000000	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	13.002015	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	22.000000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	29.699118	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	35.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

o Outlier Detection

```
In [28]: sns.boxplot(df.Age)
```

```
Out[28]: <Axes: >
```



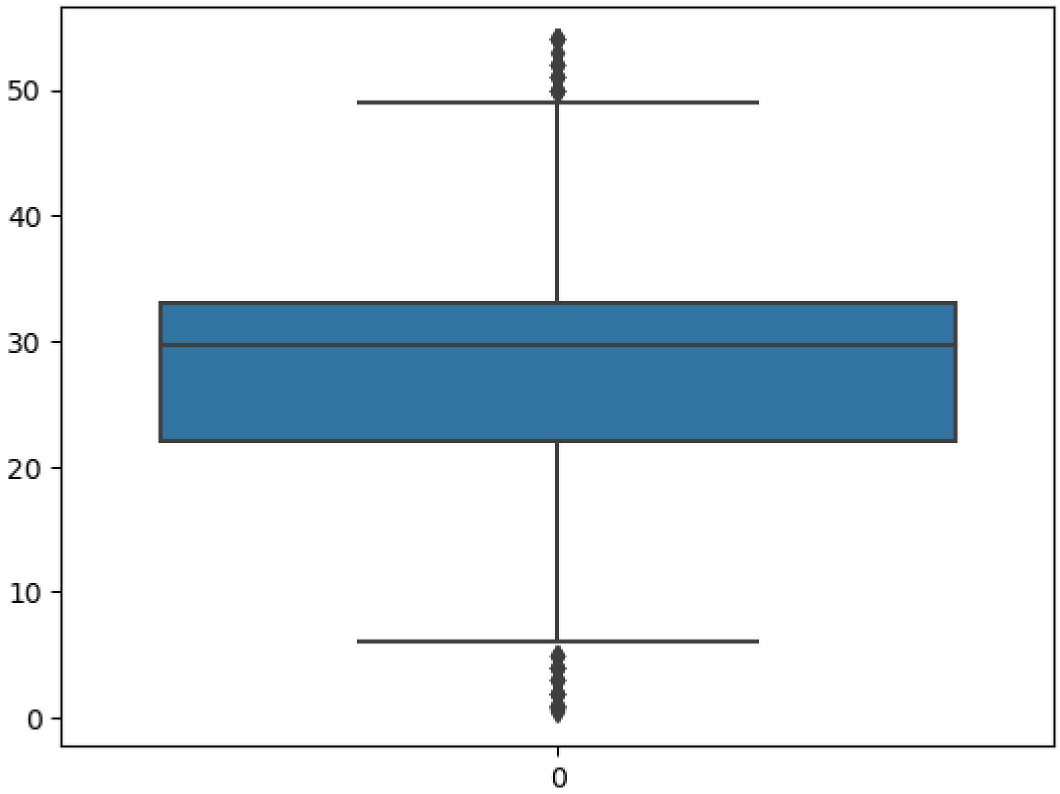
```
In [29]: df.shape
```

```
Out[29]: (891, 11)
```

In [30]:
```
q1 = df.Age.quantile(0.25)
q3 = df.Age.quantile(0.75)
IQR = q3-q1
upper_limit = q3+1.5*IQR
df = df[df.Age<upper_limit]
```

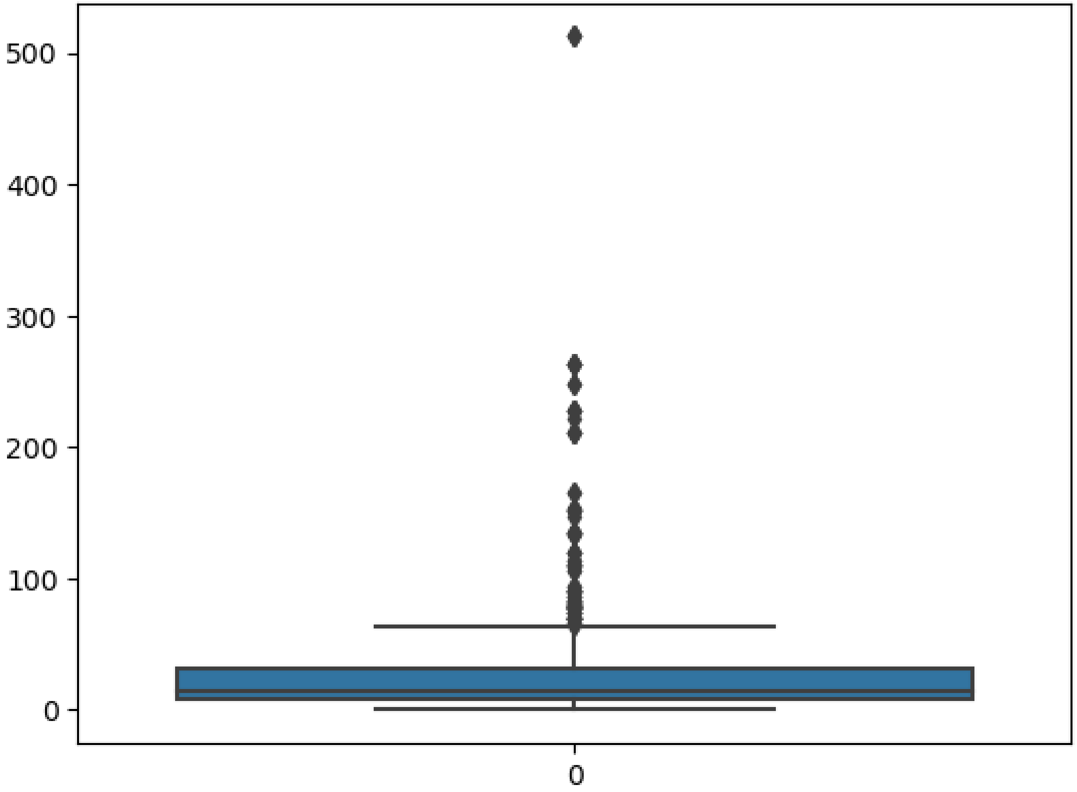
In [31]:
```
sns.boxplot(df.Age)
```

Out[31]:
```
<Axes: >
```



In [32]:
```
sns.boxplot(df.Fare)
```

Out[32]:
```
<Axes: >
```



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

Out[33]:
```
(849, 11)
```

In [34]:

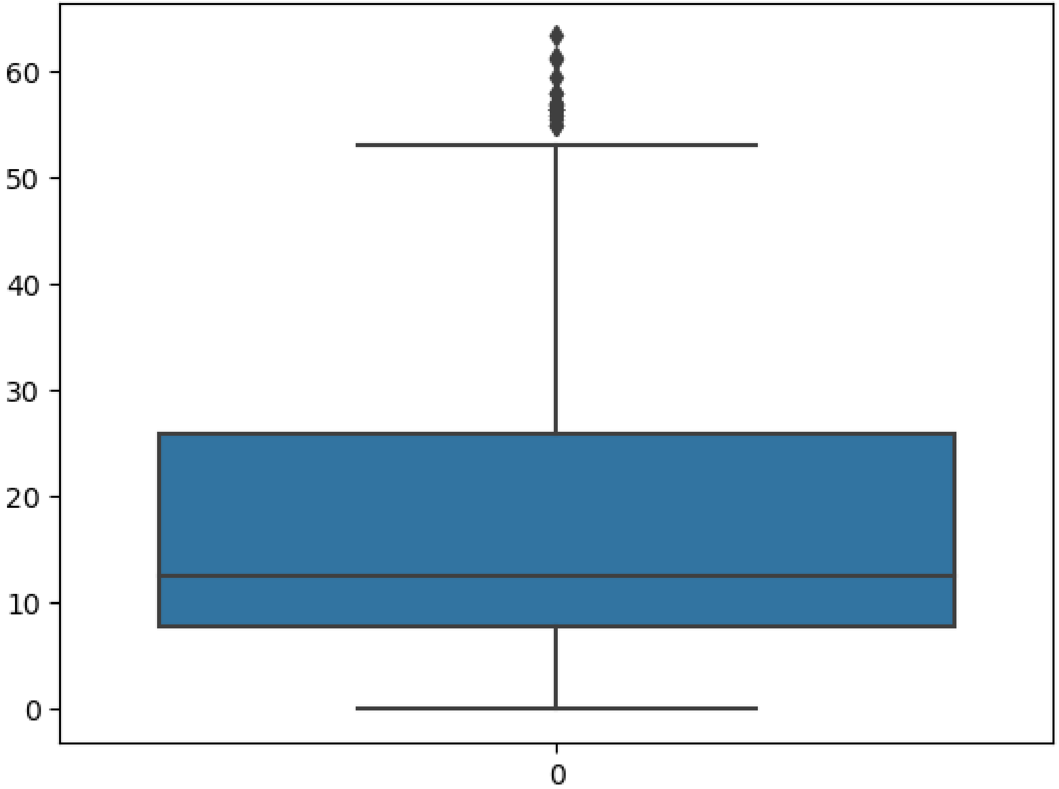
```
q1 = df.Fare.quantile(0.25)
q3 = df.Fare.quantile(0.75)
IQR = q3-q1
upper_limit = q3+1.5*IQR
df = df[df.Fare<upper_limit]
```

In [35]:

```
sns.boxplot(df.Fare)
```

Out[35]:

```
<Axes: >
```



o Splitting Dependent and Independent variables

In [36]:
```
df.head()
```

Out[36]:

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

In [37]:
```
x=df.drop(columns=["Name","Ticket","Embarked"],axis=1)
x.head()
```

Out[37]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare
0	1	0	3	male	22.000000	1	0	7.2500
2	3	1	3	female	26.000000	0	0	7.9250
3	4	1	1	female	35.000000	1	0	53.1000
4	5	0	3	male	35.000000	0	0	8.0500
5	6	0	3	male	29.699118	0	0	8.4583

In [38]:
```
x.shape
```

Out[38]:
```
(741, 8)
```

In [39]:
```
type(x)
```

Out[39]:
```
pandas.core.frame.DataFrame
```

In [40]:
```
y=df["Embarked"]
y
```

Out[40]:
```
0      S
2      S
3      S
4      S
5      Q
      ..
886    S
887    S
888    S
889    C
890    Q
Name: Embarked, Length: 741, dtype: object
```

o Perform Encoding

In [41]:
```
from sklearn.preprocessing import LabelEncoder
le =LabelEncoder()
```

In [42]:
```
x["Sex"]=le.fit_transform(x["Sex"])
```

In [43]:
```
print(le.classes_)
```

```
['female' 'male']
```

In [44]:
```
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

Out[44]:
```
{'female': 0, 'male': 1}
```

o Feature Scaling.

In [45]:
```
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
```

In [46]:
```
x_scaled=pd.DataFrame(ms.fit_transform(x),columns=x.columns)
x_scaled.head()
```

Out[46]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare
0	0.000000	0.0	1.0	1.0	0.402762	0.2	0.0	0.114429
1	0.002247	1.0	1.0	0.0	0.477417	0.0	0.0	0.125082
2	0.003371	1.0	0.0	0.0	0.645390	0.2	0.0	0.838091
3	0.004494	0.0	1.0	1.0	0.645390	0.0	0.0	0.127055
4	0.005618	0.0	1.0	1.0	0.546456	0.0	0.0	0.133499

o Splitting Data into Train and Test

In [47]:
```
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,random_state =0)
```

In [48]:
```
print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(592, 8) (149, 8) (592,) (149,)
```