Importing necessary libraries

In [3]:

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

Importing the dataset

In [4]:

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

In [5]:

```
data.head()
```

Out[5]:

	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 [6]:

```
data.tail()
```

Out[6]:

	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 [7]:

```
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

In [8]:

```
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
```

Checking for null values

This can be done by,

1. deleting null values

2. deleting row/column

3. replace with mean/median or mode

In [9]:

```
data.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
```

We find that columns Age, Cabin and Embarked contain null values.

In [10]:

```
data.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]:

```
new_data=data
new_data['Age']=new_data['Age'].fillna(new_data['Age'].mean())
new_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          891 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 [12]:

```
new_data['Cabin']=new_data['Cabin'].fillna('Unknown',inplace=True)
```

In [13]:

```
new_data['Embarked']=new_data['Embarked'].fillna('Embarked',inplace=True)
```

In [14]:

```
new_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          891 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        0 non-null      object 
 11  Embarked     0 non-null      object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Data Visualization

In [15]:

```
corr=data.corr()
corr
```

Out[15]:

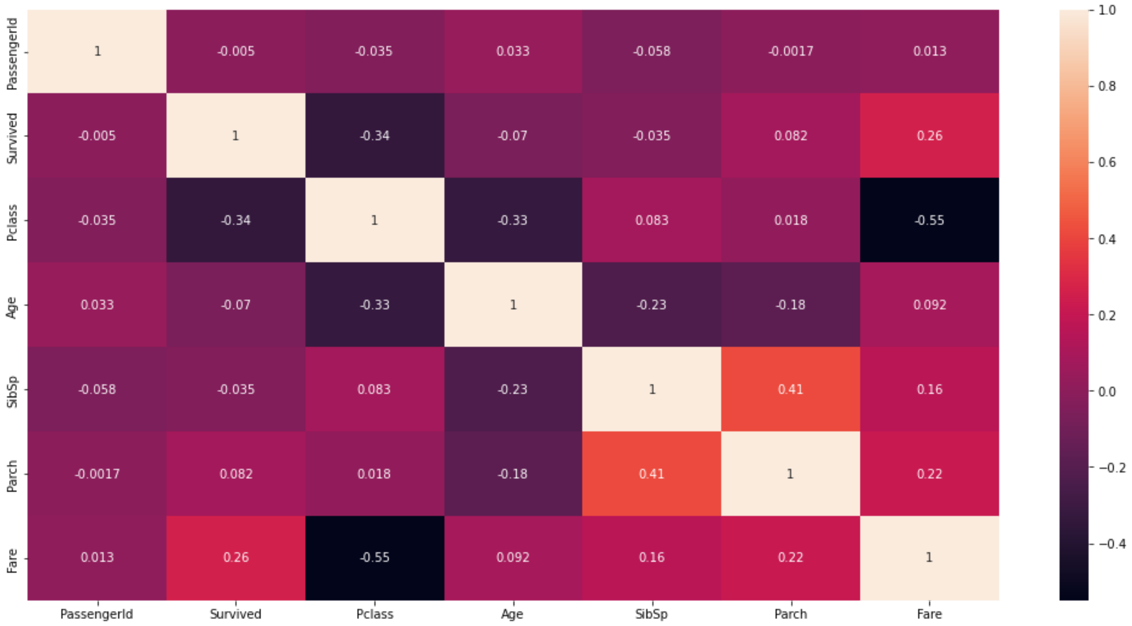
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.033207	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.069809	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.331339	0.083081	0.018443	-0.549500
Age	0.033207	-0.069809	-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 [16]:

```
plt.subplots(figsize=(18,9))
sns.heatmap(corr,annot=True)
```

Out[16]:

```
<AxesSubplot:>
```



Outlier Detection

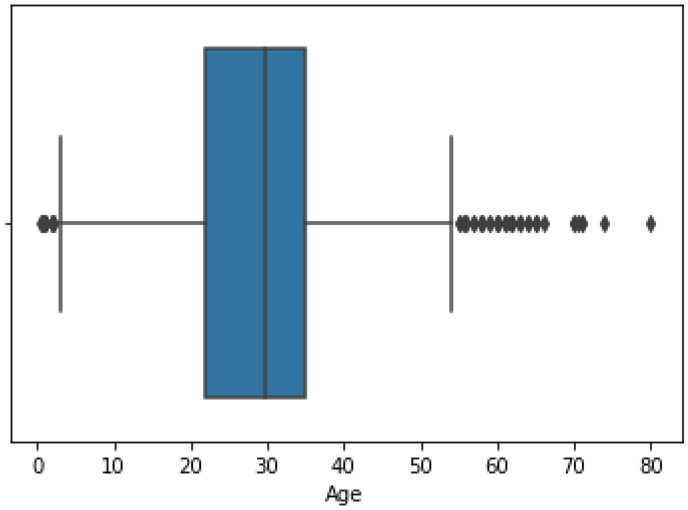
In [17]:

```
sns.boxplot(data.Age)
```

```
C:\Users\ishan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Fut
ureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing othe
r arguments without an explicit keyword will result in an error or misint
erpretation.
  warnings.warn(
```

Out[17]:

```
<AxesSubplot:xlabel='Age'>
```



Splitting Dependent and Independent variables

In [18]:

```
x=data.iloc[:,2:9]
y=data.iloc[:,9]
```

In [19]:

```
x.head()
```

Out[19]:

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

In [20]:

```
y.head()
```

Out[20]:

```
0     7.2500
1    71.2833
2     7.9250
3    53.1000
4     8.0500
Name: Fare, dtype: float64
```

Perform Encoding

We can perform label Encoding on Sex column

In [21]:

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

In [22]:

```
le=LabelEncoder()
```

In [23]:

```
x['Sex']=le.fit_transform(x['Sex'])
```

In [24]:

```
x['Sex']
```

Out[24]:

```
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 [25]:

```
x['Sex'].value_counts()
```

Out[25]:

```
1    577
0    314
Name: Sex, dtype: int64
```

In [26]:

```
x.head()
```

Out[26]:

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

In [27]:

```
x.Pclass.value_counts()
```

Out[27]:

```
3    491
1    216
2    184
Name: Pclass, dtype: int64
```

We can perform one hot encoding on Pclass cloumn

In [28]:

```
x.shape
```

Out[28]:

```
(891, 7)
```

In [29]:

```
Pclass=pd.get_dummies(x['Pclass'])
```

In [38]:

```
Pclass
```

Out[38]:

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

891 rows × 3 columns

In [44]:

```
x.head()
```

Out[44]:

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

Splitting into training and testing dataset

In [48]:

```
#890 rows
#training data 700-800
#testing data 200-300
from sklearn.model_selection import train_test_split
```

In [52]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
x_train.shape,y_train.shape,x_test.shape,y_test.shape
```

Out[52]:

```
((712, 12), (712,), (179, 12), (179,))
```

Feature Scaling

In [1]:

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [32]:

```
x[['Age','SibSp']]=sc.fit_transform(x[['Age','SibSp']])
```

In [33]:

```
x.head()
```

Out[33]:

	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	3	Braund, Mr. Owen Harris	1	-0.592481	0.432793	0	A/5 21171
1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	0.638789	0.432793	0	PC 17599
2	3	Heikkinen, Miss. Laina	0	-0.284663	-0.474545	0	STON/O2. 3101282
3	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	0.407926	0.432793	0	113803
4	3	Allen, Mr. William Henry	1	0.407926	-0.474545	0	373450

In [ ]:

```

```