Assignment-3

N SANDEEP

21BCB7116

Import the dataset:

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.simplefilter(action="ignore",category=FutureWarning)
```

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

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	5

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embark
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	female	38.0	1	0	PC 17599	71.2833	C85	

```
df.tail()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S

Johnston

```
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

```
df.shape
```

```
(891, 12)
```

```
df.describe()
```

	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

```
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
```

```
df.corr()
```

	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

```
df.corr().Fare.sort_values(ascending=False)
```

```
Fare           1.000000
Survived       0.257307
Parch          0.216225
SibSp          0.159651
Age            0.096067
PassengerId    0.012658
Pclass        -0.549500
Name: Fare, dtype: float64
```

DATA PREPROCESSING:

Checking If there any null values:

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

```
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
```

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

```
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
```

```
df['Age'].fillna(df['Age'].mean(),inplace=True)
df['Embarked'].fillna(df['Embarked'].mode()[0],inplace=True)
```

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

```
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
```

```
df = df.drop(['Cabin'], axis=1)
df
# We dropped cabin beacuse it has highest number of null values.
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embark
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.000000	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	
				Allen, Mr.							

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

```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age            False
SibSp          False
Parch          False
Ticket         False
Fare           False
```

```
Embarked       False
dtype: bool
```

```
# check if there any duplicates

df.duplicated().any()
```

```
False
```

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

```
3
```

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

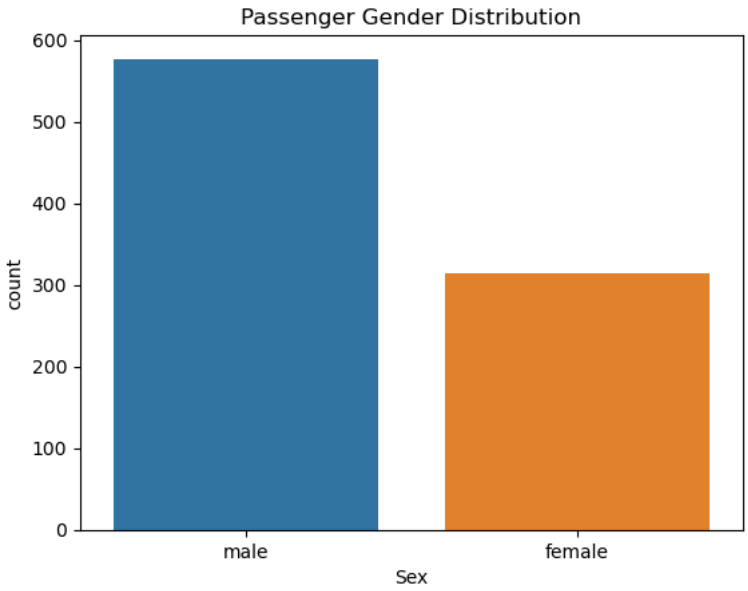
```
array(['S', 'C', 'Q'], dtype=object)
```

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

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

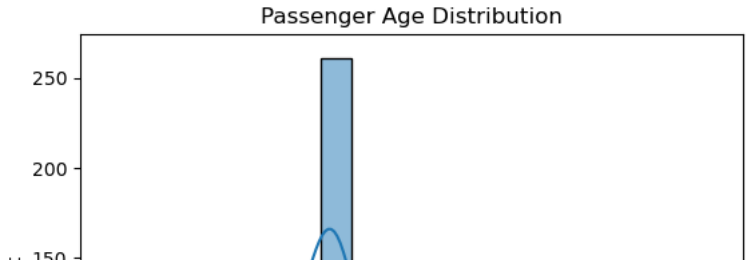
DATA VISUALIZATION:

```
sns.countplot(data=df, x='Sex')
plt.title('Passenger Gender Distribution')
plt.show()
```



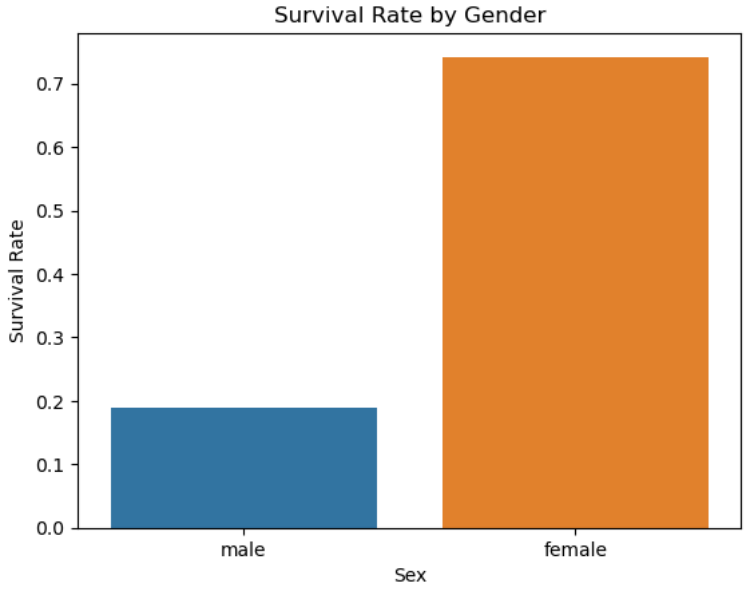
INFERENCE: We can observe that there are more number of male passengers than female passengers

```
sns.histplot(data=df, x='Age', bins=20, kde=True)
plt.title('Passenger Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



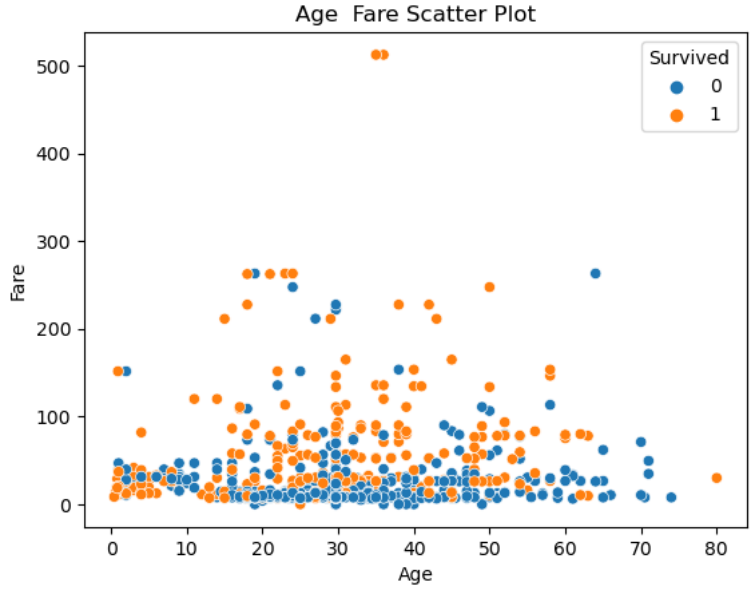
INFERENCE: The histogram of the 'Age' distribution provides insights into the age of Titanic passengers, showing that the majority were 30 to 40 aged adults, but there were also significant numbers of younger and older passengers.

```
sns.barplot(data=df, x='Sex', y='Survived', ci=None)
plt.title('Survival Rate by Gender')
plt.ylabel('Survival Rate')
plt.show()
```



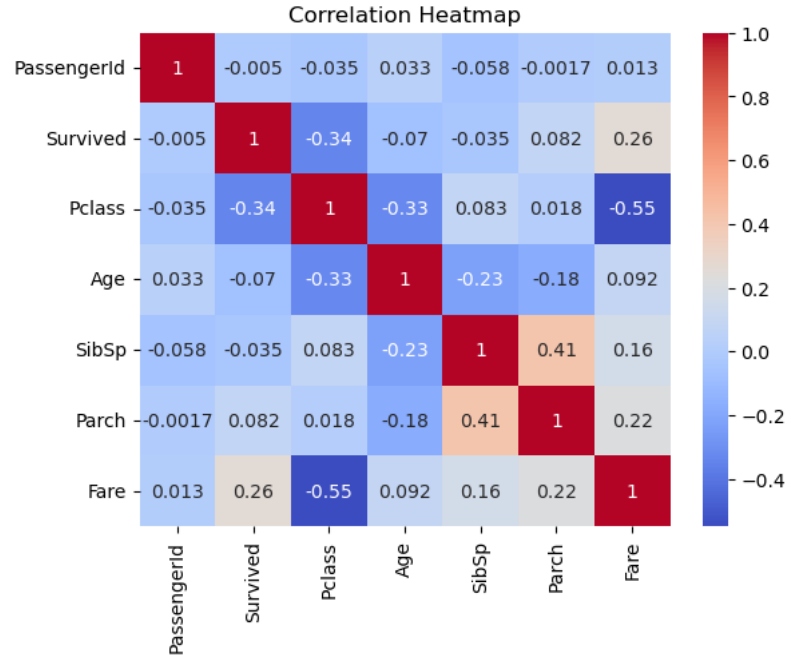
INFERENCE: we can observe that female passengers have high survival rate than male passengers

```
sns.scatterplot(data=df, x='Age', y='Fare', hue='Survived')
plt.title('Age  Fare Scatter Plot')
plt.xlabel('Age')
plt.ylabel('Fare')
plt.show()
```



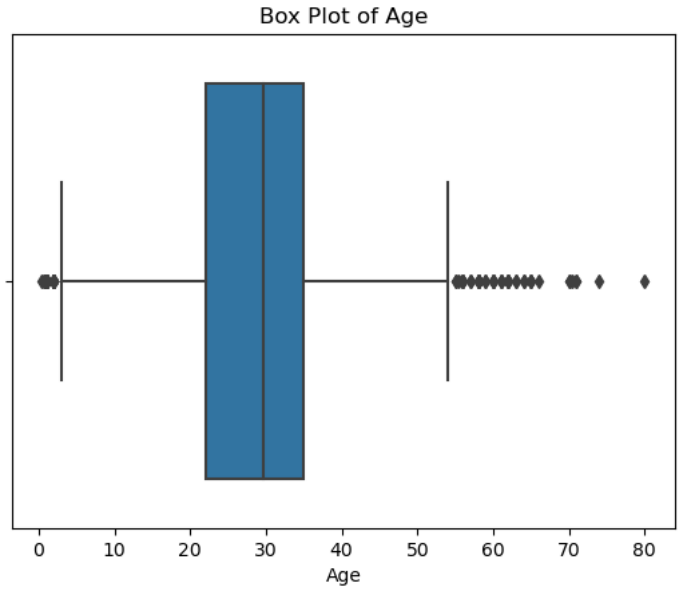

```
correlation_matrix = df.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
```

```
plt.title('Correlation Heatmap')
plt.show()
```



OUTLIER DETECTION

```
sns.boxplot(data=df, x='Age')
plt.title('Box Plot of Age')
plt.show()
```




```
df.shape
```

```
(891, 11)
```

```
q1=df.Age.quantile(0.25)
q3=df.Age.quantile(0.75)
print(q1)
print(q3)
```

```
22.0
35.0
```

```
IQR=q3-q1
IQR
```

```
13.0
```

```
upper_limit=q3+1.5*IQR
upper_limit
```

```
54.5
```

```
lower_limit=q3-1.5*IQR
lower_limit
```

```
15.5
```

```
from scipy import stats

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

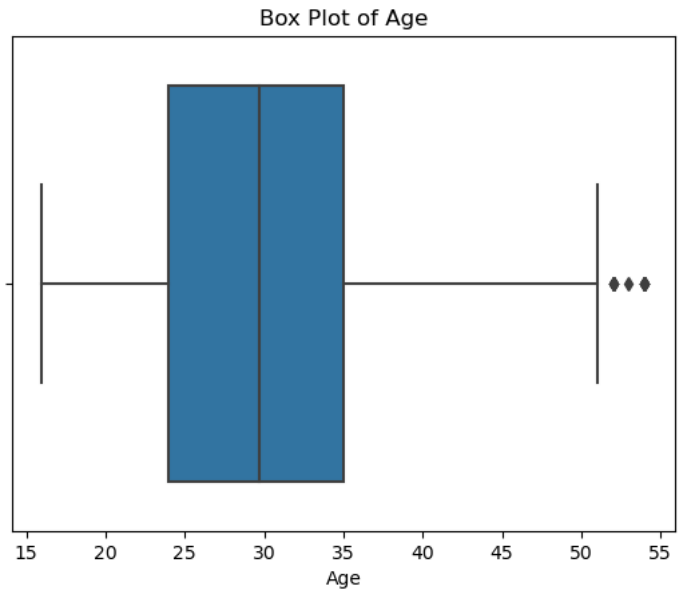
```
z_scores
```

```
0      0.592481
1      0.638789
2      0.284663
3      0.407926
4      0.407926
         ...
886    0.207709
887    0.823344
888    0.000000
889    0.284663
890    0.177063
Name: Age, Length: 891, dtype: float64
```

```
df_no_outliers = df[(df['Age'] >= lower_limit) & (df['Age'] <= upper_limit)]
print("Original dataset shape:", df.shape)
print("Dataset shape after removing outliers:", df_no_outliers.shape)
```

```
Original dataset shape: (891, 11)
Dataset shape after removing outliers: (766, 11)
```

```
sns.boxplot(data=df_no_outliers, x='Age')
plt.title('Box Plot of Age')
plt.show()
```



SPLITTING DEPENDENT AND INDEPENDENT VARIABLES:

```
df.head()
```

	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
				Cumings, Mrs.							

```
X=df.drop (columns=["Fare"],axis=1)
X.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	S

```
X.shape
```

```
(891, 10)
```

```
type(X)
```

```
pandas.core.frame.DataFrame
```

```
y=df["Fare"]
y.head()
```

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

ENCODING:

```
X.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Embarked
0	1	0	3	Braund, Mr. Owen Harris	1	22.0	1	0	A/5 21171	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	38.0	1	0	PC 17599	C
2	3	1	3	Heikkinen, Miss. Laina	0	26.0	0	0	STON/O2. 3101282	S
				Futrelle, Mrs. Jacques						

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
X["Sex"]=le.fit_transform(X["Sex"])
X.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Embarked
0	1	0	3	Braund, Mr. Owen Harris	1	22.0	1	0	A/5 21171	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	38.0	1	0	PC 17599	C
2	3	1	3	Heikkinen, Miss. Laina	0	26.0	0	0	STON/O2. 3101282	S
				Futrelle, Mrs. Jacques						

```
X["Embarked"]=le.fit_transform(X["Embarked"])
X.head()
```

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

```
print(le.classes_)
```

```
['C' 'Q' 'S']
```

```
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

```
{'C': 0, 'Q': 1, 'S': 2}
```

FEATURE SCALING:

```
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
```

```
from sklearn.preprocessing import MinMaxScaler
numerical_features = ['Age', 'Fare']
data = df[numerical_features]
ms = MinMaxScaler()
scaled_data = ms.fit_transform(data)
df_scaled = pd.DataFrame(scaled_data, columns=numerical_features)
print(df_scaled.head())
```

```
        Age      Fare
0  0.271174  0.014151
1  0.472229  0.139136
2  0.321438  0.015469
3  0.434531  0.103644
4  0.434531  0.015713
```

SPLITTING DATA INTO TRAIN AND TEST:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

```
print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(712, 10) (179, 10) (712,) (179,)
```

```
X = df.drop('Survived', axis=1)
y = df['Survived']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(712, 10) (179, 10) (712,) (179,)
```

```
df= df.drop(['PassengerId', 'Name', 'Ticket'], axis=1)
df
```

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