D.LAKSHMAN
21BCE9053

IMPORTING THE LIBRARIES

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

IMPORTING THE DATASET

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

```
data.head()
```

	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	female	38.0	1	0	PC 17599	71.2833

```
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
```

```
data.shape
```

```
(891, 12)
```

```
data.describe()
```

CHECKING FOR NULL VALUES

```
data.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
```

```
data.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
```

```
data.corr()
```

```
C:\Users\srich\AppData\Local\Temp\ipykernel_22176\2627137660.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is
  data.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

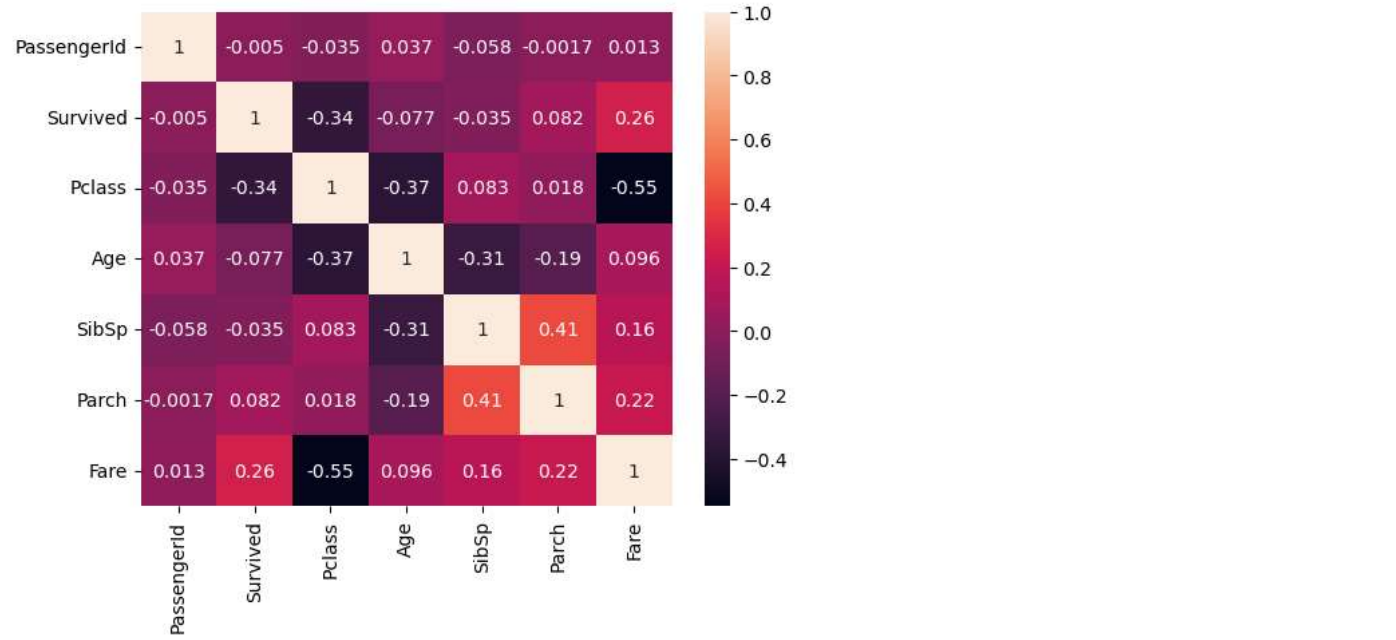
DATA VISUALIZATION

```
plt.scatter(data["Survived"],data["Age"])
```

```
<matplotlib.collections.PathCollection at 0x2863f337710>
```

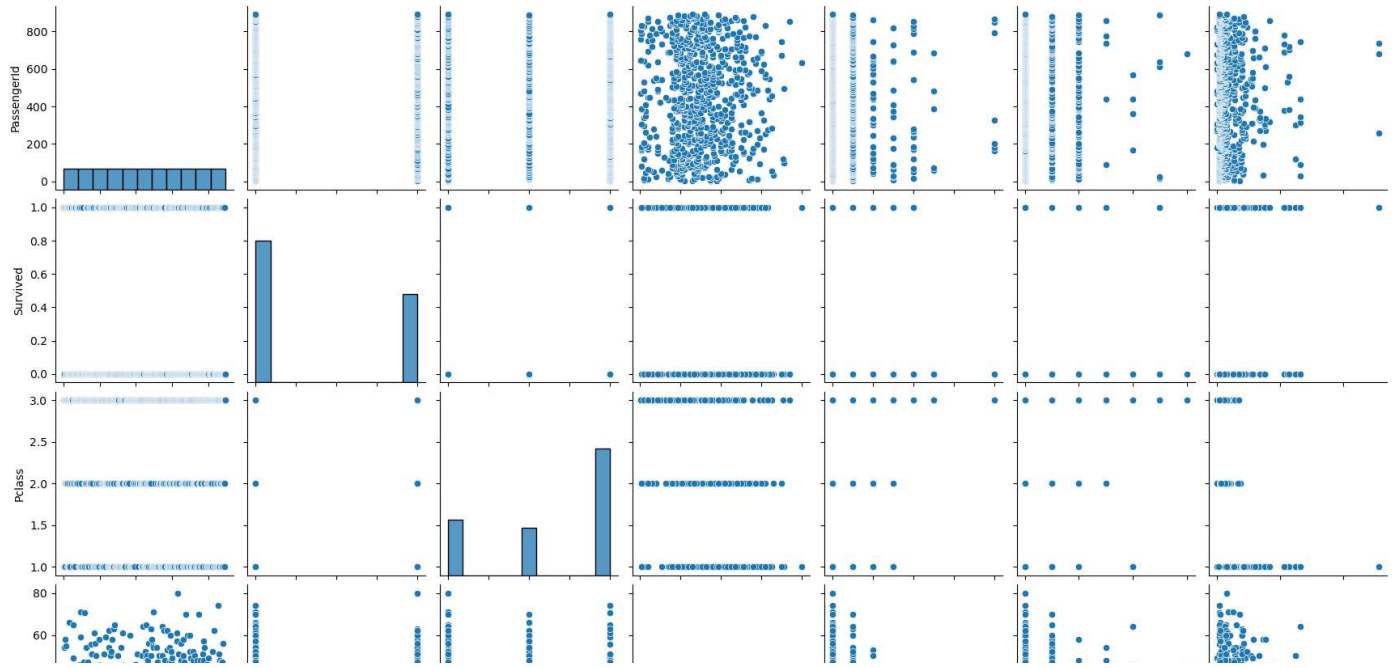

```
sns.heatmap(data.corr(),annot=True)
```

```
C:\Users\srich\AppData\Local\Temp\ipykernel_22176\2578434383.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is
  sns.heatmap(data.corr(),annot=True)
<Axes: >
```



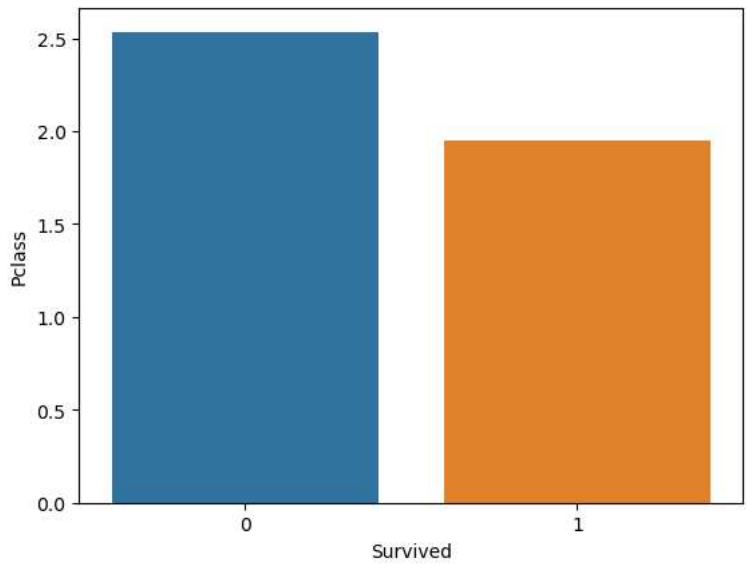

```
sns.pairplot(data)
```

```
<seaborn.axisgrid.PairGrid at 0x2863f49c410>
```




```
sns.barplot(x=data["Survived"],y=data["Pclass"],ci=0)
```

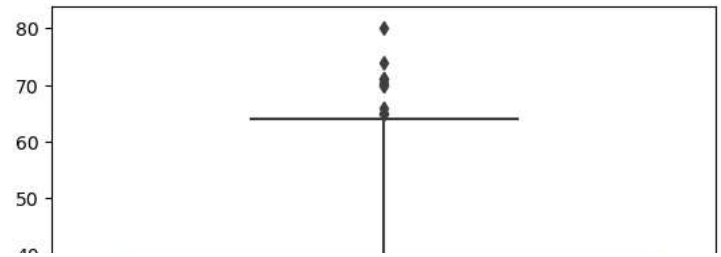
```
C:\Users\srich\AppData\Local\Temp\ipykernel_22176\2456638004.py:1: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=('ci', 0)` for the same effect.

  sns.barplot(x=data["Survived"],y=data["Pclass"],ci=0)
<Axes: xlabel='Survived', ylabel='Pclass'>
```



OUTLIER DETECTION

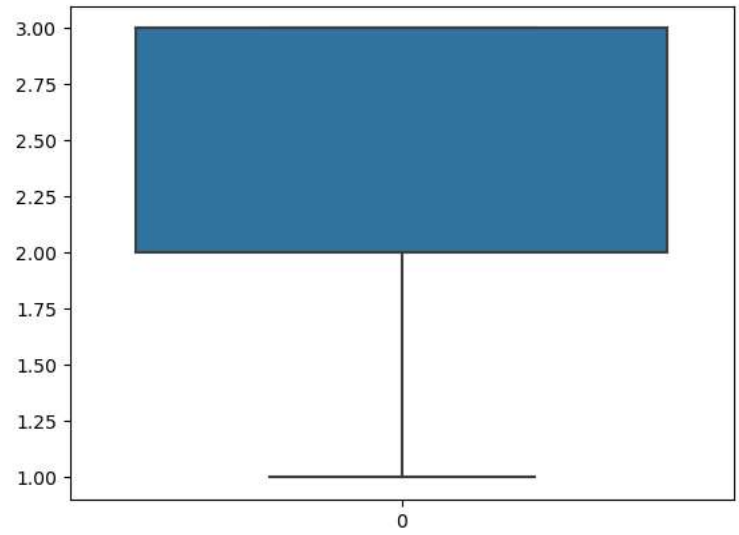
```
sns.boxplot(data.Age)
```

```
<Axes: >
```




```
sns.boxplot(data.Pclass)
```

```
<Axes: >
```



SPLITTING DEPENDENT AND INDEPENDENT VARIABLES

```
data.head()
```

	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
				Futrelle, Mrs. Jacques								

```
x=data.drop(columns=["Survived","PassengerId","Name","Ticket","Cabin"])
```

```
x
```

	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

```
x.shape
```

```
(891, 7)
```

```
type(x)
```

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

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

```
y.head
```

```
<bound method NDFrame.head of 0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64>
```

```
type(y)
```

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

ENCODING

```
x.head()
```

	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

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

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

```
x.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	S
1	1	0	38.0	1	0	71.2833	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

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

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

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

```
{'female': 0, 'male': 1}
```

```
x["Embarked"]=le.fit_transform(x["Embarked"])
```

```
x.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	2
1	1	0	38.0	1	0	71.2833	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

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

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

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

```
{'C': 0, 'Q': 1, 'S': 2, nan: 3}
```

```
x.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	2
1	1	0	38.0	1	0	71.2833	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

Feature Scaling

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

```
 x_Scaled=pd.DataFrame(ms.fit_transform(x),columns=x.columns)
```

```
x_Scaled.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1.0	1.0	0.271174	0.125	0.0	0.014151	0.666667
1	0.0	0.0	0.472229	0.125	0.0	0.139136	0.000000
2	1.0	0.0	0.321438	0.000	0.0	0.015469	0.666667
3	0.0	0.0	0.434531	0.125	0.0	0.103644	0.666667
4	1.0	1.0	0.434531	0.000	0.0	0.015713	0.666667

SPLITTING DATA INTO TRAINING AND TESTING

```
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)
```

```
print(X_train.shape,X_test.shape,y_train.shape,y_test.shape)
```

```
(712, 7) (179, 7) (712,) (179,)
```