Name: Bylapudi Lahari

Email: lahari.21bce9969@vitapstudent.ac.in

Importing the necessary libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Importing the dataset

In [2]:
```
dataset = pd.read_csv("titanic_dataset.csv")
```

In [3]:
```
dataset.head()
```

Out[3]:

	PassengerId	Name	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	Braund, Mr. Owen Harris	3	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	Heikkinen, Miss. Laina	3	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	female	35.0	1	0	113803	53.1000	C123	S
4	5	Allen, Mr. William Henry	3	male	35.0	0	0	373450	8.0500	NaN	S

In [4]:
```
dataset.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   Name         891 non-null    object 
 2   Pclass       891 non-null    int64  
 3   Sex          891 non-null    object 
 4   Age          714 non-null    float64
 5   SibSp        891 non-null    int64  
 6   Parch        891 non-null    int64  
 7   Ticket       891 non-null    object 
 8   Fare         891 non-null    float64
 9   Cabin        204 non-null    object 
 10  Embarked     889 non-null    object 
 11  Survived     891 non-null    int64  
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [5]:

```
dataset.shape
```

Out[5]:

```
(891, 12)
```

In [6]:

```
dataset.describe
```

Out[6]:

```
<bound method NDFrame.describe of      PassengerId
Name  Pclass  \
0              1                            Braund, Mr. Owen Harris       3
1              2  Cumings, Mrs. John Bradley (Florence Briggs Th...       1
2              3                             Heikkinen, Miss. Laina       3
3              4       Futrelle, Mrs. Jacques Heath (Lily May Peel)       1
4              5                           Allen, Mr. William Henry       3
..           ...                                                ...     ...
886          887                              Montvila, Rev. Juozas       2
887          888                       Graham, Miss. Margaret Edith       1
888          889           Johnston, Miss. Catherine Helen "Carrie"       3
889          890                              Behr, Mr. Karl Howell       1
890          891                                Dooley, Mr. Patrick       3

        Sex   Age  SibSp  Parch            Ticket     Fare Cabin Embarked  \
0      male  22.0      1      0         A/5 21171   7.2500   NaN        S
1    female  38.0      1      0          PC 17599  71.2833   C85        C
2    female  26.0      0      0  STON/O2. 3101282   7.9250   NaN        S
3    female  35.0      1      0            113803  53.1000  C123        S
4      male  35.0      0      0            373450   8.0500   NaN        S
..      ...   ...    ...    ...               ...      ...   ...      ...
886    male  27.0      0      0            211536  13.0000   NaN        S
887  female  19.0      0      0            112053  30.0000   B42        S
888  female   NaN      1      2        W./C. 6607  23.4500   NaN        S
889    male  26.0      0      0            111369  30.0000  C148        C
890    male  32.0      0      0            370376   7.7500   NaN        Q

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

[891 rows x 12 columns]>
```

Checking for null values

In [7]:

```
dataset.isnull().any()
```

Out[7]:

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

In [8]:

```
dataset.isnull().sum()
```

Out[8]:

```
PassengerId      0
Name             0
Pclass           0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
Survived         0
dtype: int64
```

In [9]:

```
dataset.corr()
```

Out[9]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare	Survived
PassengerId	1.000000	-0.035144	0.036847	-0.057527	-0.001652	0.012658	-0.005007
Pclass	-0.035144	1.000000	-0.369226	0.083081	0.018443	-0.549500	-0.338481
Age	0.036847	-0.369226	1.000000	-0.308247	-0.189119	0.096067	-0.077221
SibSp	-0.057527	0.083081	-0.308247	1.000000	0.414838	0.159651	-0.035322
Parch	-0.001652	0.018443	-0.189119	0.414838	1.000000	0.216225	0.081629
Fare	0.012658	-0.549500	0.096067	0.159651	0.216225	1.000000	0.257307
Survived	-0.005007	-0.338481	-0.077221	-0.035322	0.081629	0.257307	1.000000

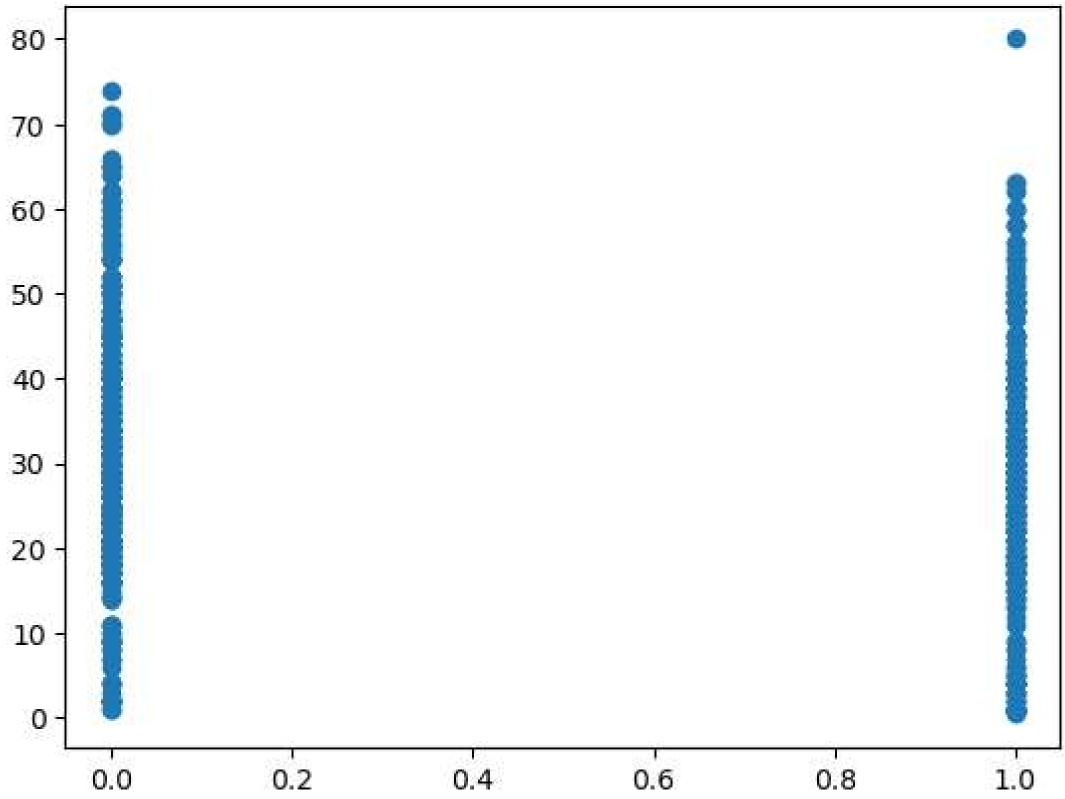
Data Visualization

In [10]:

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

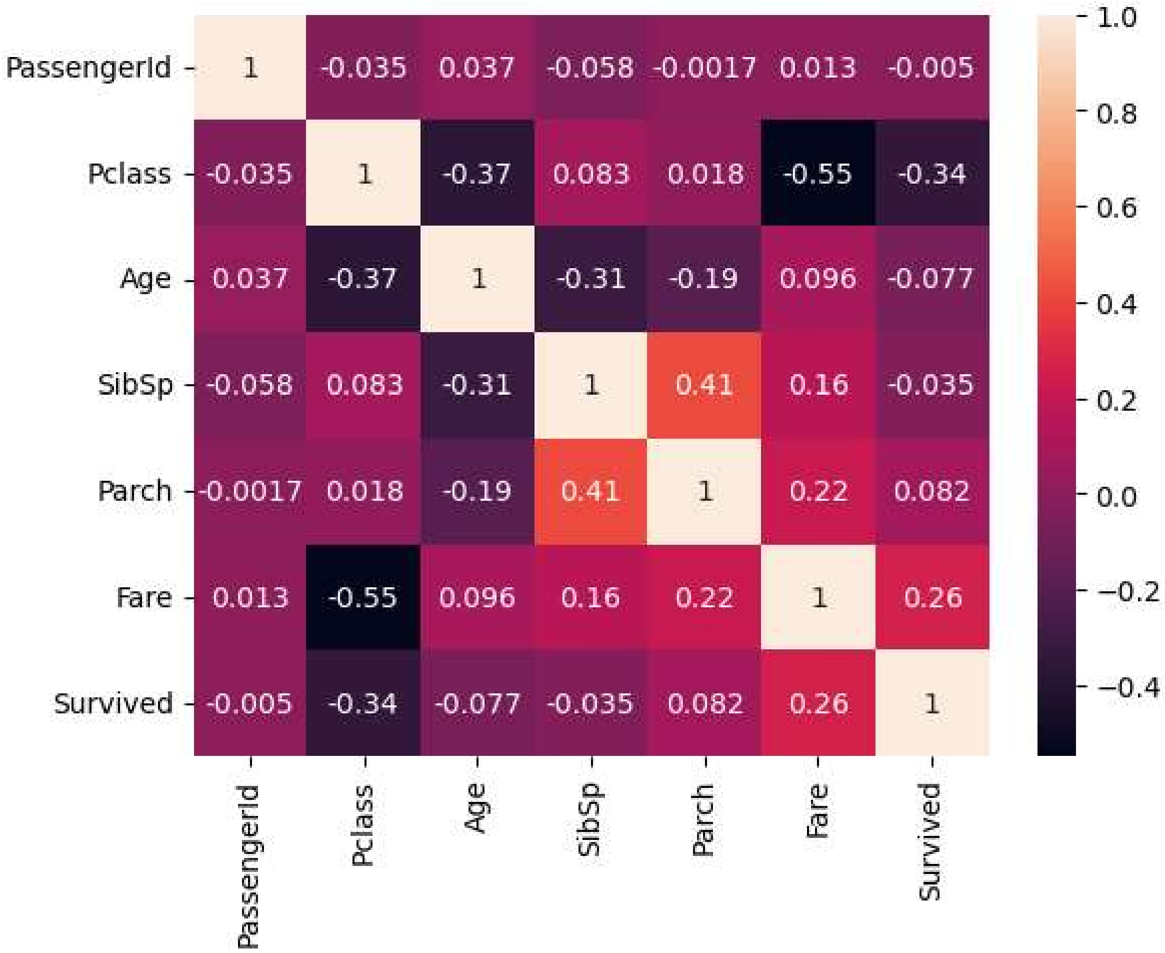
Out[10]:

```
<matplotlib.collections.PathCollection at 0x1e2bfd66f90>
```



```
In [11]: sns.heatmap(dataset.corr(), annot = True)

Out[11]: <Axes: >
```

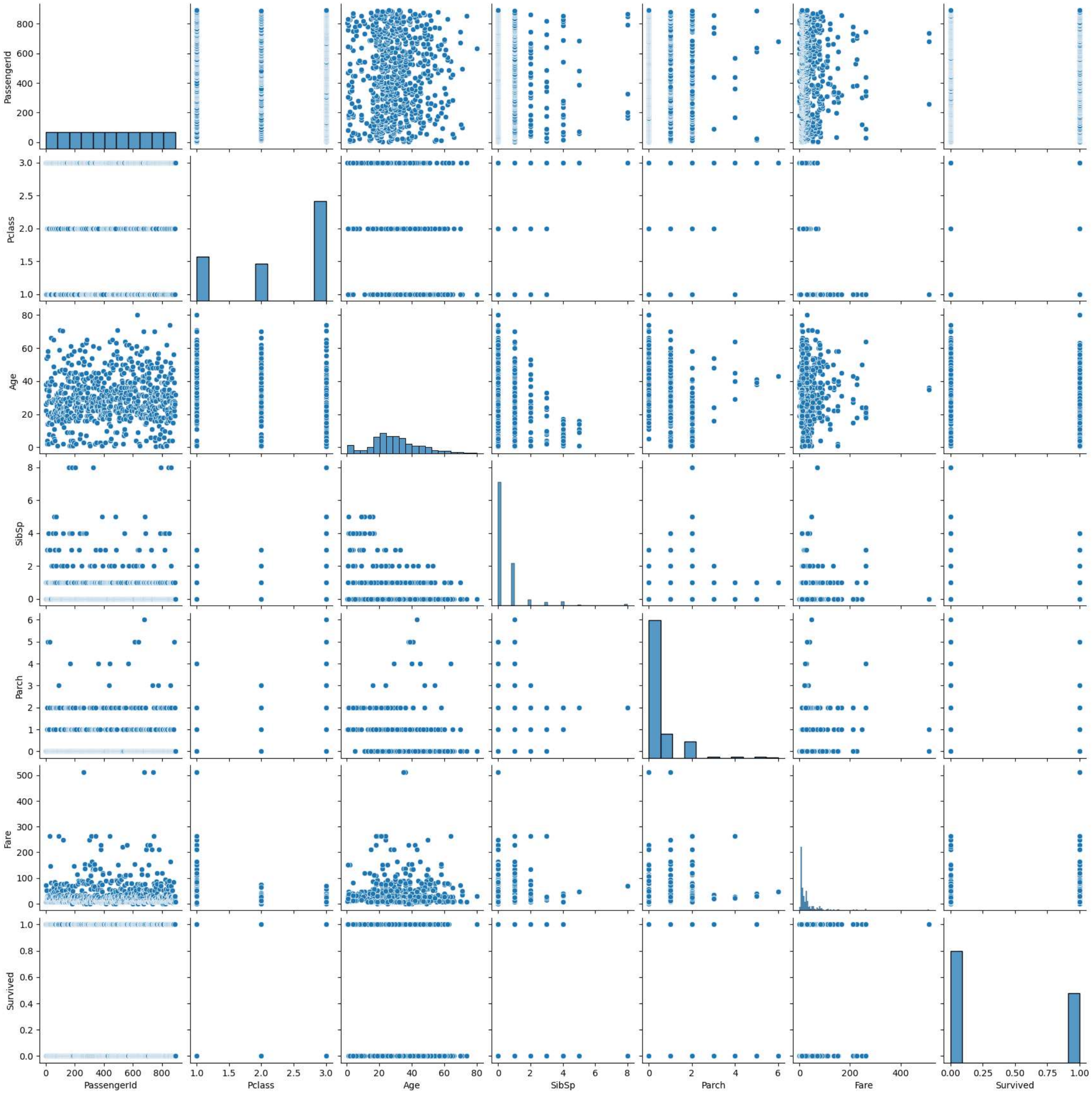


In [12]:

```
sns.pairplot(dataset)
```

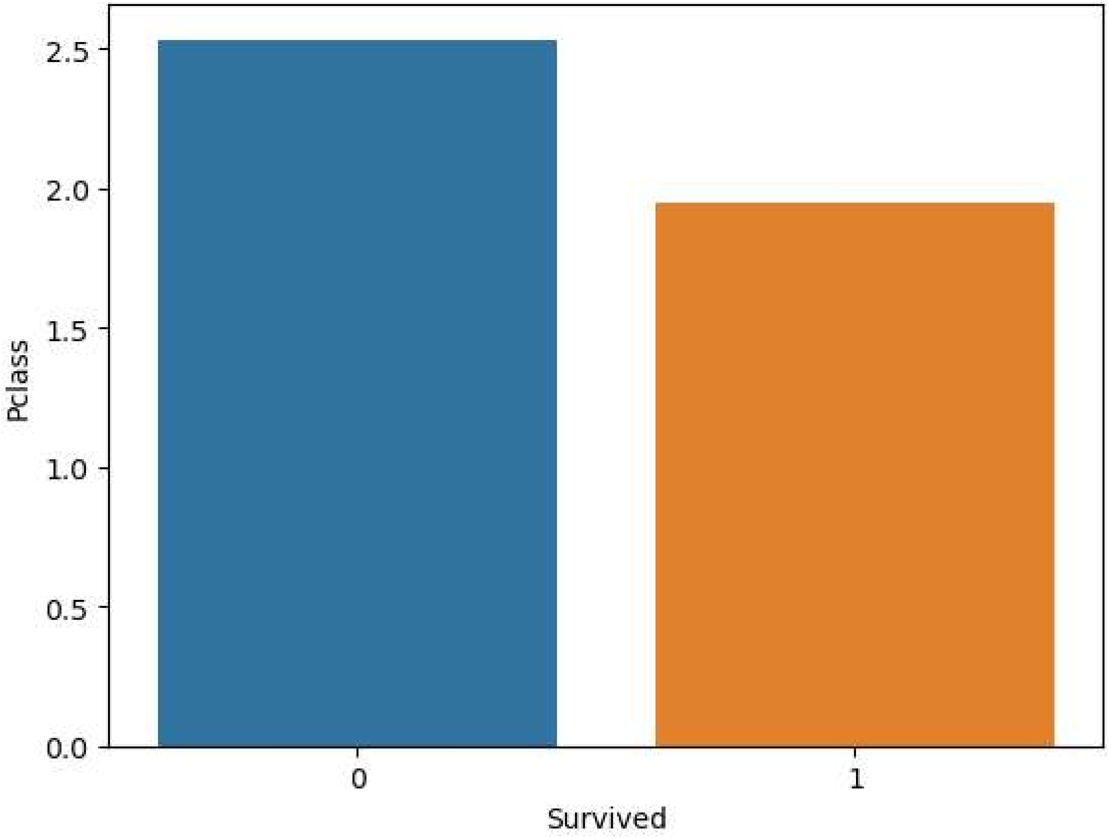
Out[12]:

```
<seaborn.axisgrid.PairGrid at 0x1e2bfe1b490>
```



```
In [13]: sns.barplot(x = dataset["Survived"], y = dataset["Pclass"], ci = 0)
```

```
Out[13]: <Axes: xlabel='Survived', ylabel='Pclass'>
```

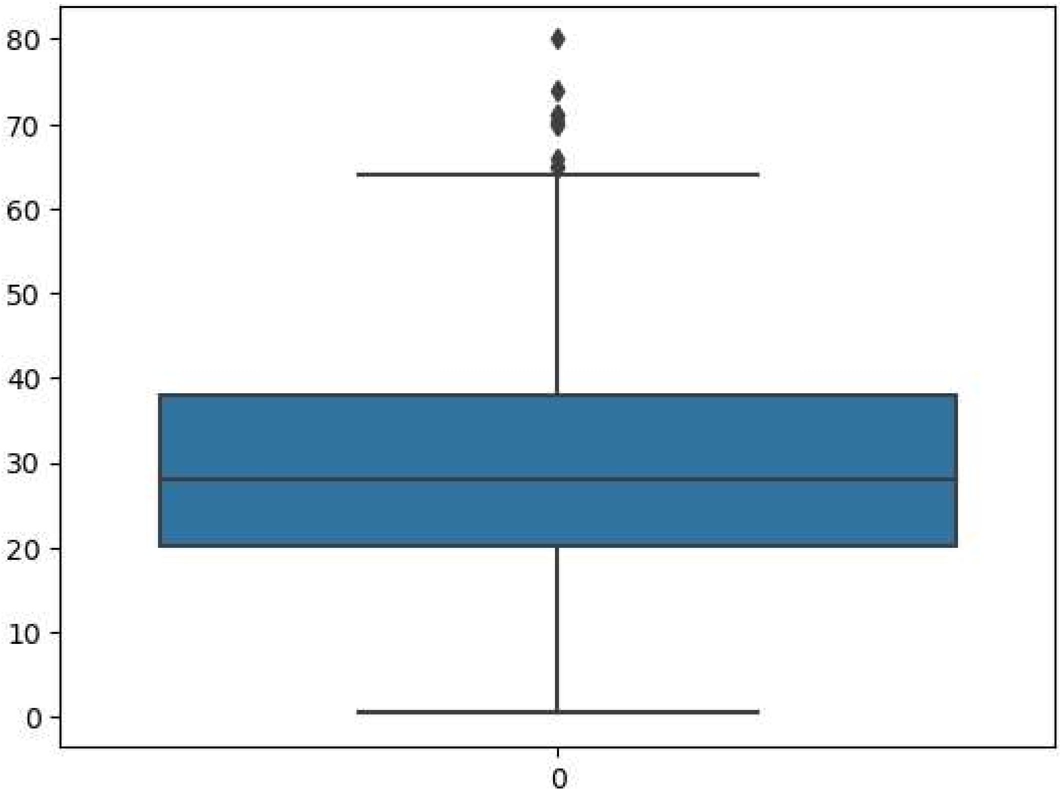


In [14]:

```
sns.boxplot(dataset.Age)
```

Out[14]:

```
<Axes: >
```

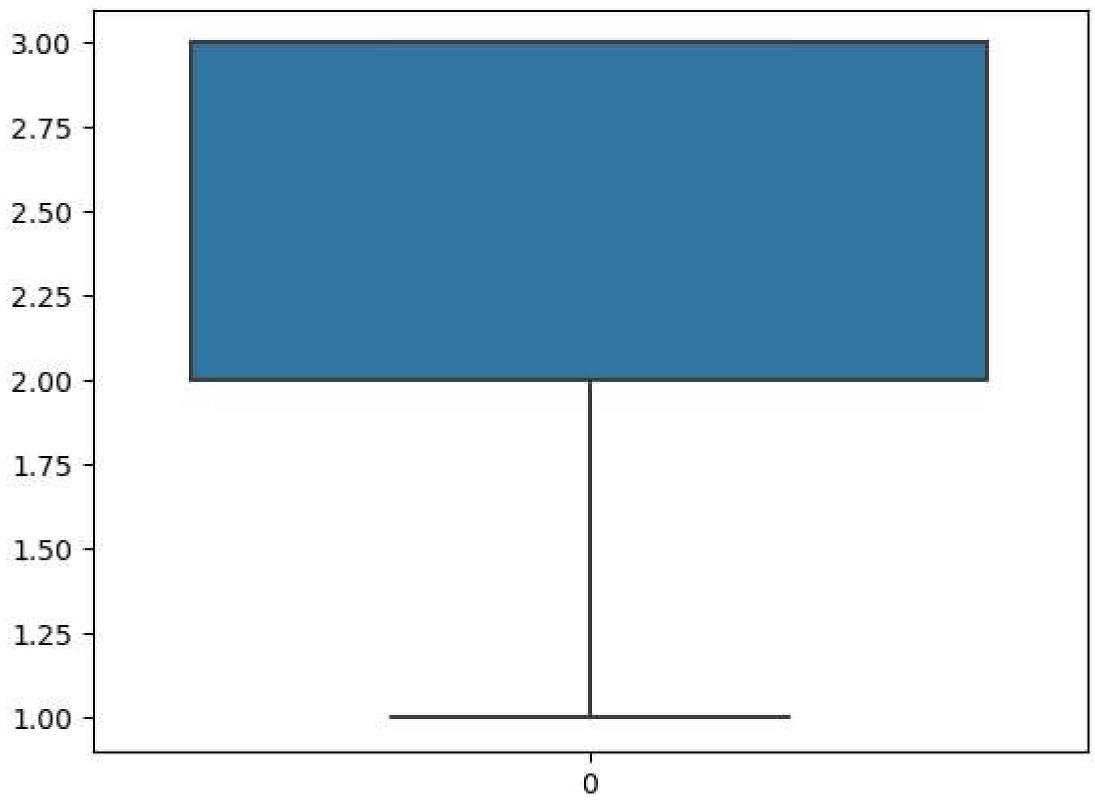


In [15]:

```
sns.boxplot(dataset.Pclass)
```

Out[15]:

```
<Axes: >
```



Splitting dependent and independent variables

In [16]:

```
dataset.head()
```

Out[16]:

	PassengerId	Name	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	Braund, Mr. Owen Harris	3	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	Heikkinen, Miss. Laina	3	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	female	35.0	1	0	113803	53.1000	C123	S
4	5	Allen, Mr. William Henry	3	male	35.0	0	0	373450	8.0500	NaN	S

In [17]:

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

In [18]:

```
x
```

Out[18]:

	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
...	...	...	...	...	...	...	...
886	2	male	27.0	0	0	13.0000	S
887	1	female	19.0	0	0	30.0000	S
888	3	female	NaN	1	2	23.4500	S
889	1	male	26.0	0	0	30.0000	C
890	3	male	32.0	0	0	7.7500	Q

891 rows × 7 columns

In [19]:

```
x.shape
```

Out[19]:

```
(891, 7)
```

In [20]:

```
type(x)
```

Out[20]:

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

In [21]:

```
y = dataset["Survived"]
```

In [22]:

```
y.head()
```

Out[22]:

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

In [23]:

```
type(y)
```

Out[23]:

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

Encoding

In [24]:

```
x.head()
```

Out[24]:

	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

In [25]:

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

In [26]:

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

In [27]:

```
x.head()
```

Out[27]:

	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

In [28]:

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

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

In [29]:

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

Out[29]:

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

In [30]:

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

In [31]:

```
x.head()
```

Out[31]:

	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

In [32]:

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

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

In [33]:

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

Out[33]:

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

In [34]:

```
x.head()
```

Out[34]:

	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

In [35]:

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

In [36]:

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

In [37]:

```
x_Scaled.head()
```

Out[37]:

	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

In [38]:

```
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
```

In [39]:

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

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

In [ ]: