Data Preprocessing

1. Import the Libraries
2. Importing the dataset.
3. Checking for Null Values.
4. Data Visualization.
5. Outlier Detection
6. Splitting Dependent and Independent variables
7. Perform Encoding
8. Feature Scaling.
9. Splitting Data into Train and Test

Import the Libraries

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

Importing the Dataset

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

```
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
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S

```
data.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
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

Checking for Null Values

```
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.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

```
data.shape
```

```
(891, 12)
```

75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000

```
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
```

```
mean = data["Age"].mean()
```

```
data["Age"] = data["Age"].fillna(mean)
```

Filling Null Values in Cabin with Mode

```
mode1 = data["Cabin"].mode()
```

```
data["Cabin"] = data["Cabin"].fillna(mode1[0])
```

```
data.isnull().any()
```

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

Filling Null Values in Embarked with Mode

```
mode2 = data["Embarked"].mode()
```

```
data["Embarked"] = data['Embarked'].fillna(mode2[0])
```

```
data.isnull().any()
```

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

Data Visualisation

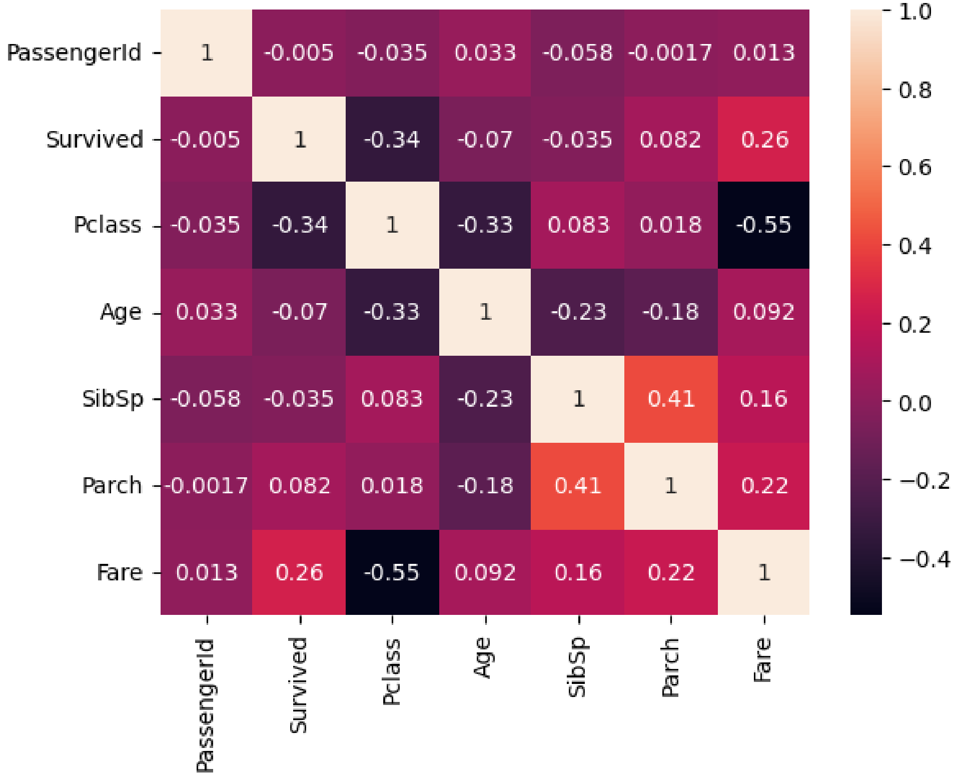
```
corr = data.corr()
corr
```

```
<ipython-input-21-df690e1cacaf>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecat
  corr = data.corr()
```

	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

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

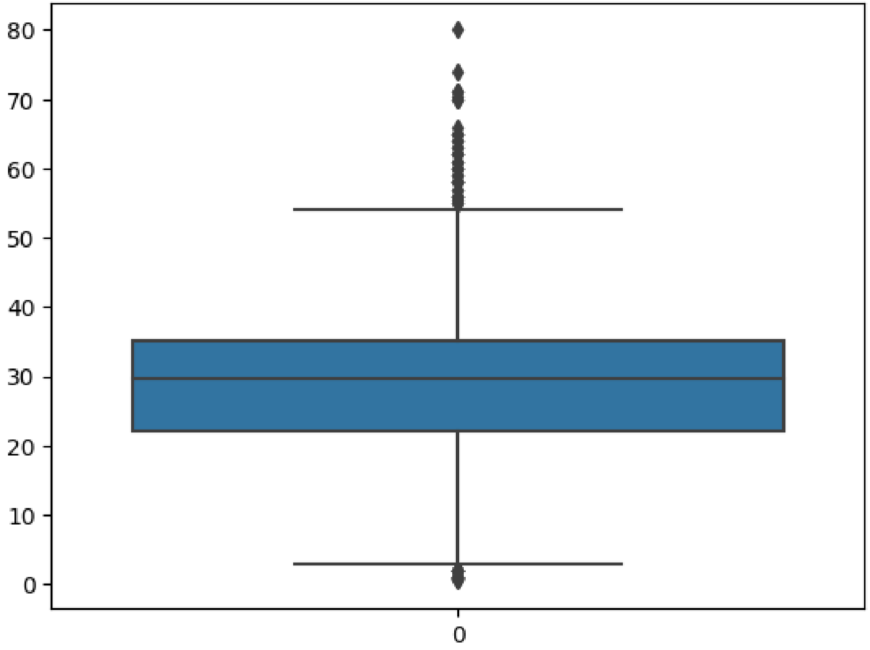
```
<Axes: >
```



Handling The Outliers

```
sns.boxplot(data['Age'])
```

```
<Axes: >
```



```
Age_q1 = data.Age.quantile(0.25)
Age_q3 = data.Age.quantile(0.75)
print(Age_q1)
print(Age_q3)
```

```
22.0
35.0
```

```
IQR_Age=Age_q3-Age_q1
IQR_Age
```

```
13.0
```

```
upperlimit_Age=Age_q3+1.5*IQR_Age
upperlimit_Age
```

```
     54.5
```

```
lowerlimit_Age = Age_q1-1.5*IQR_Age
lowerlimit_Age
```

```
     2.5
```

```
medain_age = data["Age"].median()
medain_age
```

```
     29.69911764705882
```

```
data.median()
```

```
     <ipython-input-48-135339ac59ce>:1: FutureWarning: The default value of numeric_only in DataFrame.median is deprecated. In a future version, it will default to
       data.median()
     PassengerId    446.000000
     Survived         0.000000
     Pclass           3.000000
     Age             29.699118
     SibSp            0.000000
     Parch            0.000000
     Fare            14.454200
     dtype: float64
```

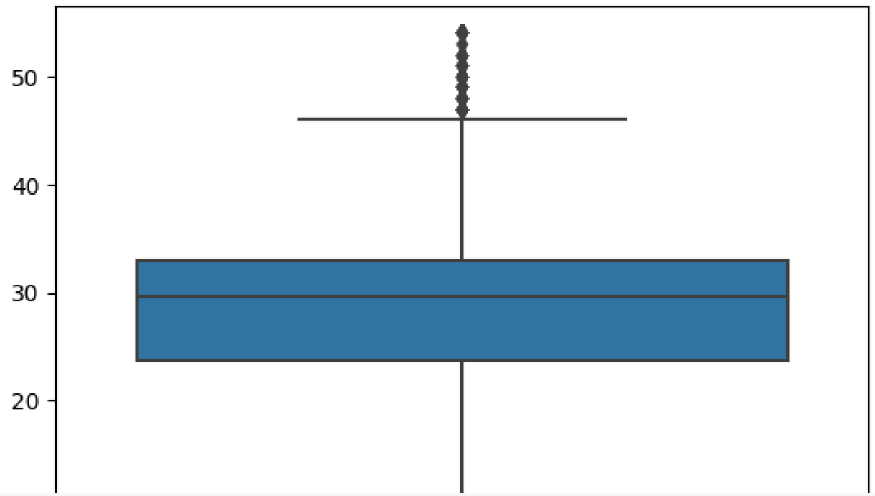
```
data["Age"] = np.where(data["Age"]>upperlimit_Age,medain_age,data["Age"])
```

```
(data["Age"]>54.5).sum()
```

```
     0
```

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

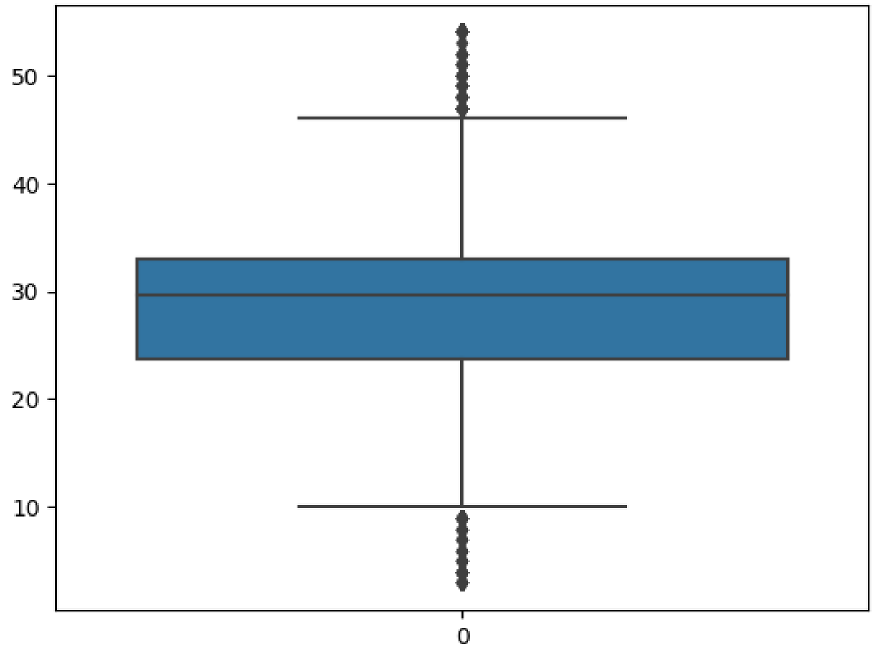
```
<Axes: >
```



```
data["Age"]=np.where(data["Age"]<lowerlimit_Age,medain_age,data["Age"])
```

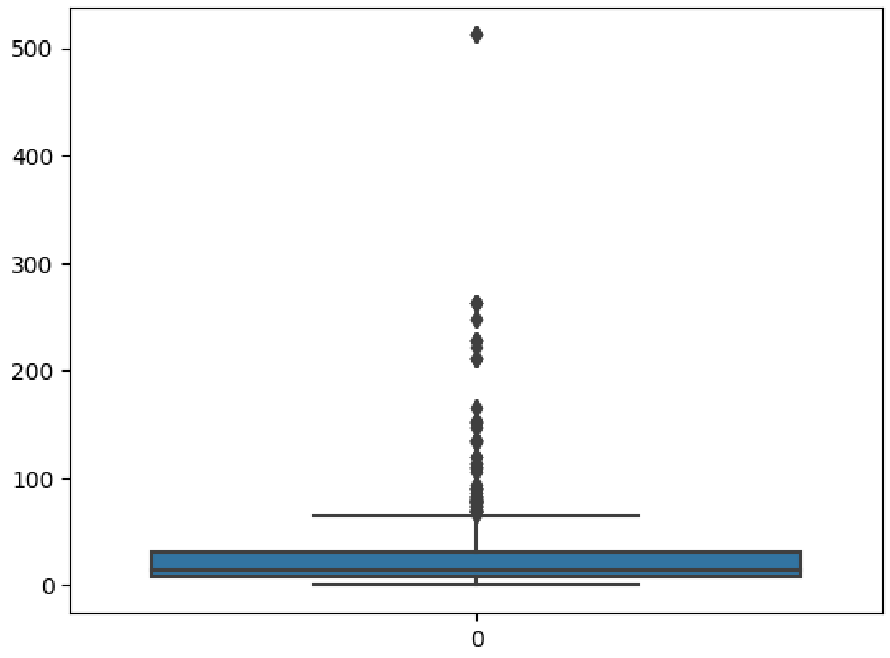
```
sns.boxplot(data["Age"])
```

```
<Axes: >
```



```
sns.boxplot(data["Fare"])
```

```
<Axes: >
```



```
Fare_q1 = data.Fare.quantile(0.25)
Fare_q3 = data.Fare.quantile(0.75)
print(Fare_q1)
print(Fare_q3)
```

```
7.9104
31.0
```

```
IQR_Fare=Fare_q3-Fare_q1
IQR_Fare
```

```
23.0896
```

```
upperlimit_Fare=Fare_q3+1.5*IQR_Fare
upperlimit_Fare
```

```
65.6344
```

```
lower_limit_Fare = Fare_q1-1.5*IQR_Fare
lower_limit_Fare
```

```
-26.724
```

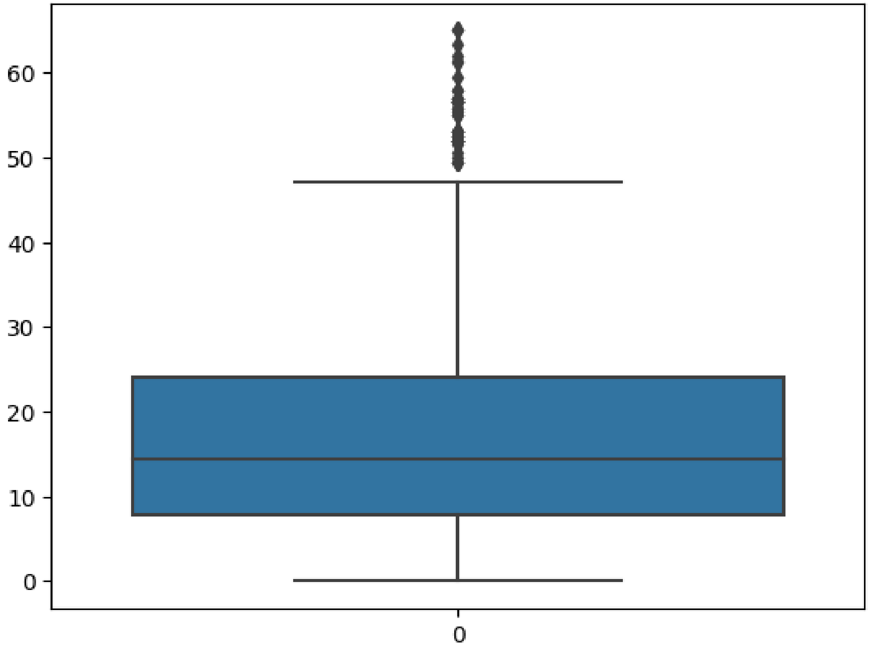
```
median_Fare=data["Fare"].median()
median_Fare
```

```
14.4542
```

```
data["Fare"] = np.where((data["Fare"]>upperlimit_Fare),median_Fare,data["Fare"])
```

```
sns.boxplot(data["Fare"])
```

```
<Axes: >
```



```
(data["Fare"]>65).sum()
```

```
0
```

Dropping the Columns

```
data.drop(['Name'],axis=1,inplace=True)
```

```
data
```

```
     PassengerId  Survived  Pclass     Sex        Age  SibSp  Parch     Fare    Cabin  Embarked
0              1         0       3    male  22.000000      1      0   7.2500  B96 B98         S
1              2         1       1  female  38.000000      1      0  14.4542      C85         C
2              3         1       3  female  26.000000      0      0   7.9250       G6         S
3              4         1       1  female  35.000000      1      0  53.1000     C123         S
4              5         0       3    male  35.000000      0      0   8.0500  B96 B98         S
...          ...       ...     ...     ...        ...    ...    ...      ...      ...       ...
886          887         0       2    male  27.000000      0      0  13.0000  B96 B98         S
887          888         1       1  female  19.000000      0      0  30.0000      B42         S
888          889         0       3  female  29.699118      1      2  23.4500  B96 B98         S
889          890         1       1    male  26.000000      0      0  30.0000     C148         C
890          891         0       3    male  32.000000      0      0   7.7500  B96 B98         Q

891 rows × 10 columns
```

```
data.drop(["Ticket"],axis = 1,inplace = True)
```

```
data
```

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	1	0	3	male	22.000000	1	0	7.2500	B96 B98	S
1	2	1	1	female	38.000000	1	0	14.4542	C85	C
2	3	1	3	female	26.000000	0	0	7.9250	G6	S
3	4	1	1	female	35.000000	1	0	53.1000	C123	S
4	5	0	3	male	35.000000	0	0	8.0500	B96 B98	S
...	...	...	...	...	...	...	...	...	...	...
887	888	1	1	female	19.000000	0	0	30.0000	B42	S

```
data.drop(["PassengerId"],axis = 1,inplace = True)
```

```
data
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	0	3	male	22.000000	1	0	7.2500	B96 B98	S
1	1	1	female	38.000000	1	0	14.4542	C85	C
2	1	3	female	26.000000	0	0	7.9250	G6	S
3	1	1	female	35.000000	1	0	53.1000	C123	S
4	0	3	male	35.000000	0	0	8.0500	B96 B98	S
...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.000000	0	0	13.0000	B96 B98	S
887	1	1	female	19.000000	0	0	30.0000	B42	S
888	0	3	female	29.699118	1	2	23.4500	B96 B98	S
889	1	1	male	26.000000	0	0	30.0000	C148	C
890	0	3	male	32.000000	0	0	7.7500	B96 B98	Q

891 rows × 9 columns

```
data.drop(['Cabin'],axis = 1,inplace = True)
```

```
data
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
1	1	1	female	38.000000	1	0	14.4542	C
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

Seperate the data into dependent variables and Independent Variables

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

```
y.head()
```

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

```
data
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
1	1	1	female	38.000000	1	0	14.4542	C
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
889	1	1	male	26.000000	0	0	30.0000	C

891 rows × 8 columns

Encoding

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

```
l1 = LabelEncoder()
```

```
data["Sex"] = l1.fit_transform(data["Sex"])
```

```
data["Sex"]
```

```
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: int64
```

```
data.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	1	22.0	1	0	7.2500	S
1	1	1	0	38.0	1	0	14.4542	C
2	1	3	0	26.0	0	0	7.9250	S
3	1	1	0	35.0	1	0	53.1000	S

```
data["Embarked"] = l1.fit_transform(data["Embarked"])
```

```
data.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	1	22.0	1	0	7.2500	2
1	1	1	0	38.0	1	0	14.4542	0
2	1	3	0	26.0	0	0	7.9250	2
3	1	1	0	35.0	1	0	53.1000	2
4	0	3	1	35.0	0	0	8.0500	2

```
data["Pclass"].nunique()
```

```
3
```

```
data["Pclass"].unique()
```

```
array([3, 1, 2])
```

```
data["Sex"].unique()
```

```
array([1, 0])
```

```
data["Embarked"].unique()
```

```
array([2, 0, 1])
```

Splitting the train and test data

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(data,y,test_size=0.3,random_state=0)
```

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
((623, 8), (268, 8), (623,), (268,))
```

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
```

```
sc = StandardScaler()
```

```
x_train = sc.fit_transform(x_train)
```

```
x_train
```

```
array([[ 1.25474307, -1.5325562 ,  0.72592065, ..., -0.47299765,
         0.67925137,  0.56710989],
       [ 1.25474307, -1.5325562 , -1.37756104, ..., -0.47299765,
        -0.26059483, -2.03075381],
       [-0.79697591,  0.84844757,  0.72592065, ...,  1.93253327,
         2.26045064,  0.56710989],
       ...,
       [-0.79697591,  0.84844757,  0.72592065, ..., -0.47299765,
        -0.78281017, -0.73182196],
       [ 1.25474307,  0.84844757, -1.37756104, ..., -0.47299765,
        -0.03170555,  0.56710989],
       [-0.79697591, -0.34205431,  0.72592065, ...,  0.72976781,
         1.64661898,  0.56710989]])
```

```
x_test = sc.fit_transform(x_test)
```

```
x_test
```

```
array([[-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.15813988, -1.76531134],
       [-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.72165412,  0.63014911],
       [-0.77151675,  0.77963055,  0.76537495, ...,  0.87064484,
         1.03823178, -0.56758111],
       ...,
       [-0.77151675,  0.77963055,  0.76537495, ..., -0.47809977,
        -0.15847431, -1.76531134],
       [ 1.29614814,  0.77963055, -1.30654916, ..., -0.47809977,
        -0.72607524,  0.63014911],
       [-0.77151675, -1.64991582,  0.76537495, ..., -0.47809977,
         0.92369033, -1.76531134]])
```