```
#importing the important 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  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age
SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0
1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0
1
2                             Heikkinen, Miss. Laina  female  26.0
0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0
1
4                           Allen, Mr. William Henry    male  35.0
0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

checking NULL values

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

```
data.isnull().sum()
```

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

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

```
3
```

```
data.drop(['PassengerId','Name','Ticket','Cabin'],axis=1,inplace=True)
#removing unnecessary data
```

```
data.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S
1	1	1	female	38.0	1	0	71.2833	C
2	1	3	female	26.0	0	0	7.9250	S
3	1	1	female	35.0	1	0	53.1000	S
4	0	3	male	35.0	0	0	8.0500	S

```
data.isnull().sum()
```

```
Survived    0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
```

```
Embarked    2
dtype: int64
```

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

```
0      S
1      C
2      S
3      S
4      S
      ..
886    S
887    S
888    S
889    C
890    Q
Name: Embarked, Length: 889, dtype: object
```

```
data.isnull().sum()
```

```
Survived    0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Embarked    2
dtype: int64
```

data visualisation

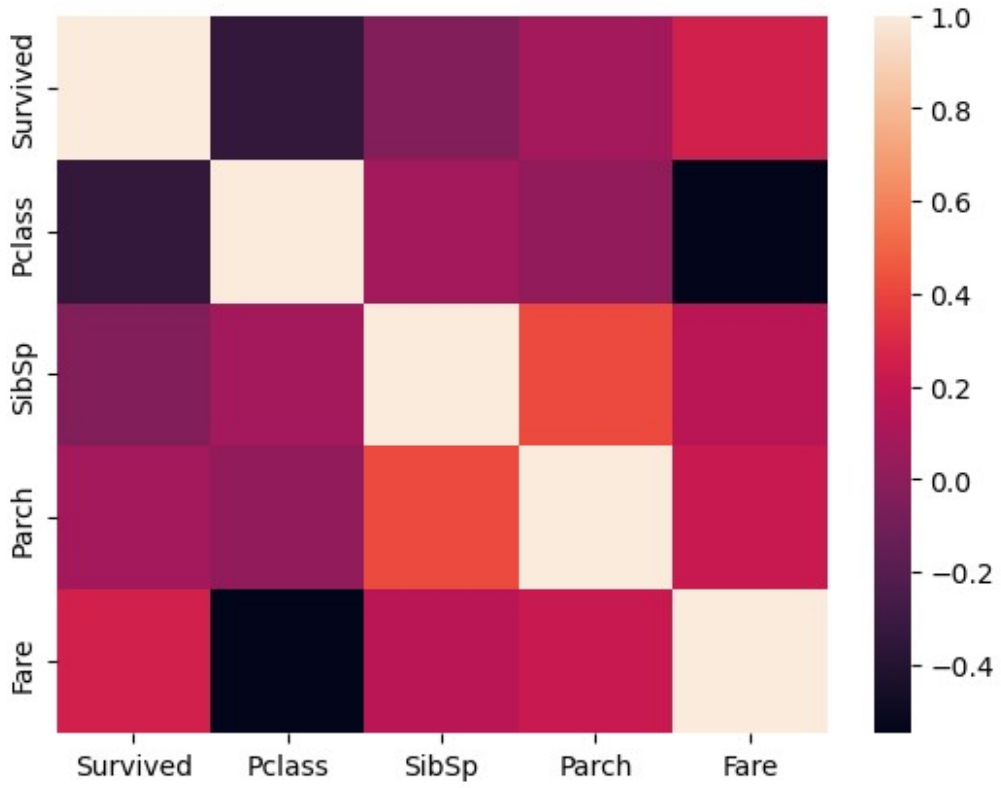
```
corr=data.corr()
corr
```

```
/var/folders/75/l1625v4n7qnbfp296v2f82ch0000gn/T/
ipykernel_6860/2248884307.py:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  corr=data.corr()
```

	Survived	Pclass	SibSp	Parch	Fare
Survived	1.000000	-0.338481	-0.035322	0.081629	0.257307
Pclass	-0.338481	1.000000	0.083081	0.018443	-0.549500
SibSp	-0.035322	0.083081	1.000000	0.414838	0.159651
Parch	0.081629	0.018443	0.414838	1.000000	0.216225
Fare	0.257307	-0.549500	0.159651	0.216225	1.000000

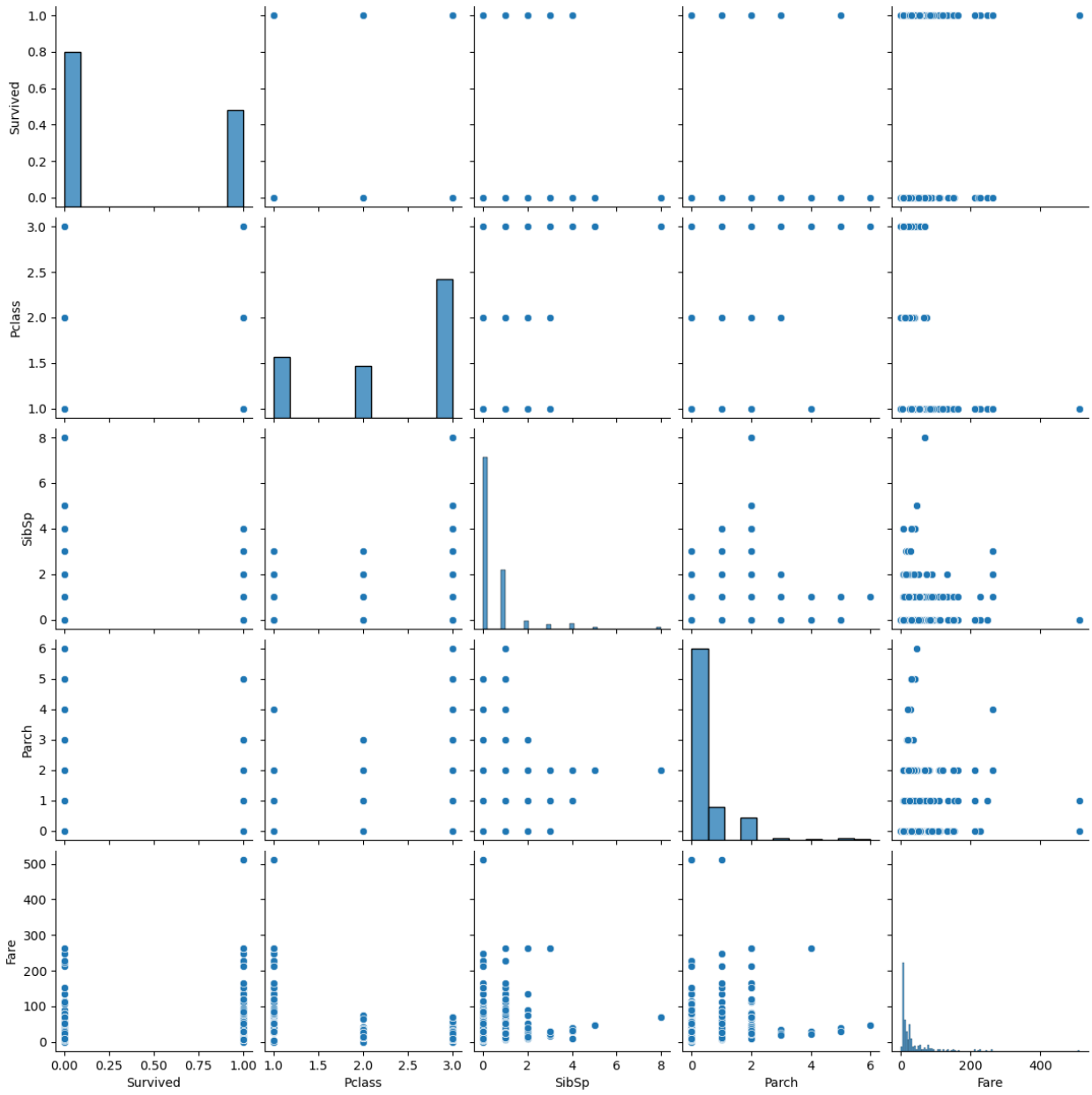
```
sns.heatmap(corr)
```

```
<Axes: >
```



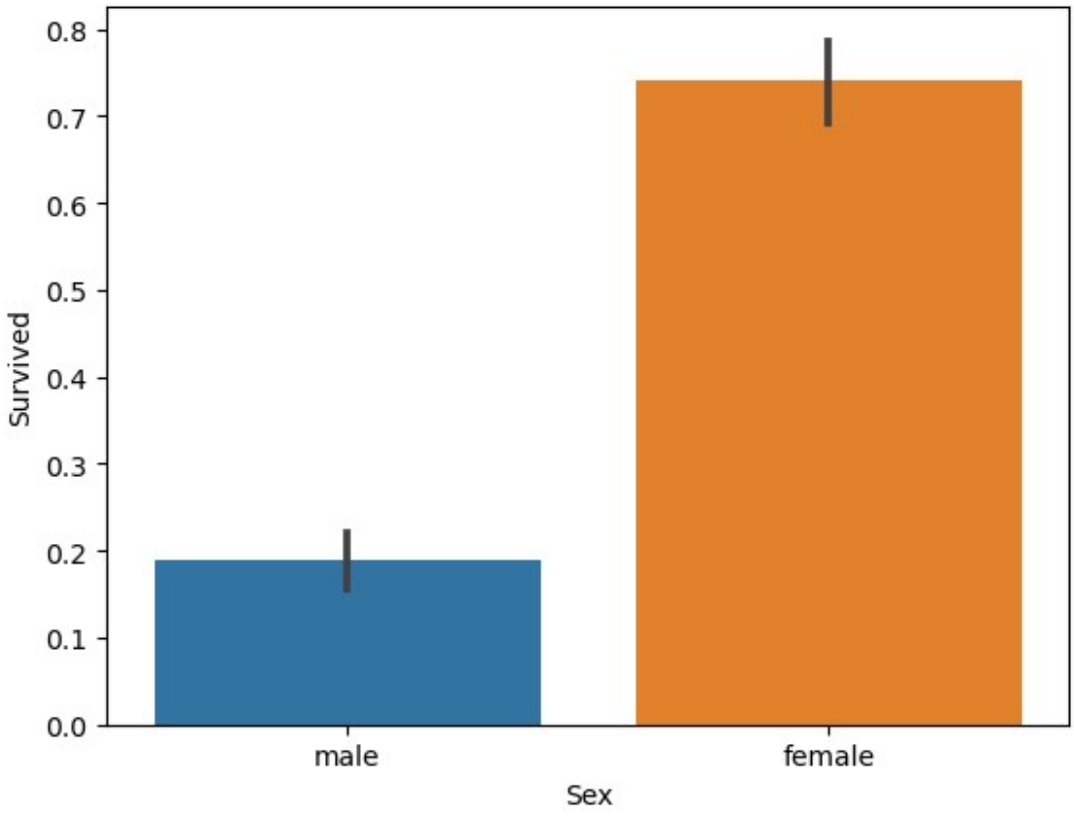

```
sns.pairplot(data)
```

```
<seaborn.axisgrid.PairGrid at 0x176f2ba10>
```



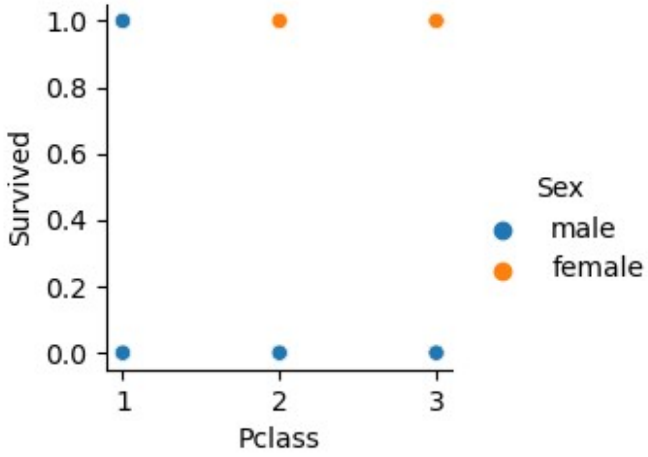

```
sns.barplot(x=data["Sex"],y=data["Survived"])
```

```
<Axes: xlabel='Sex', ylabel='Survived'>
```




```
sns.pairplot(data, x_vars=["Pclass"], y_vars=["Survived"], hue="Sex")
```

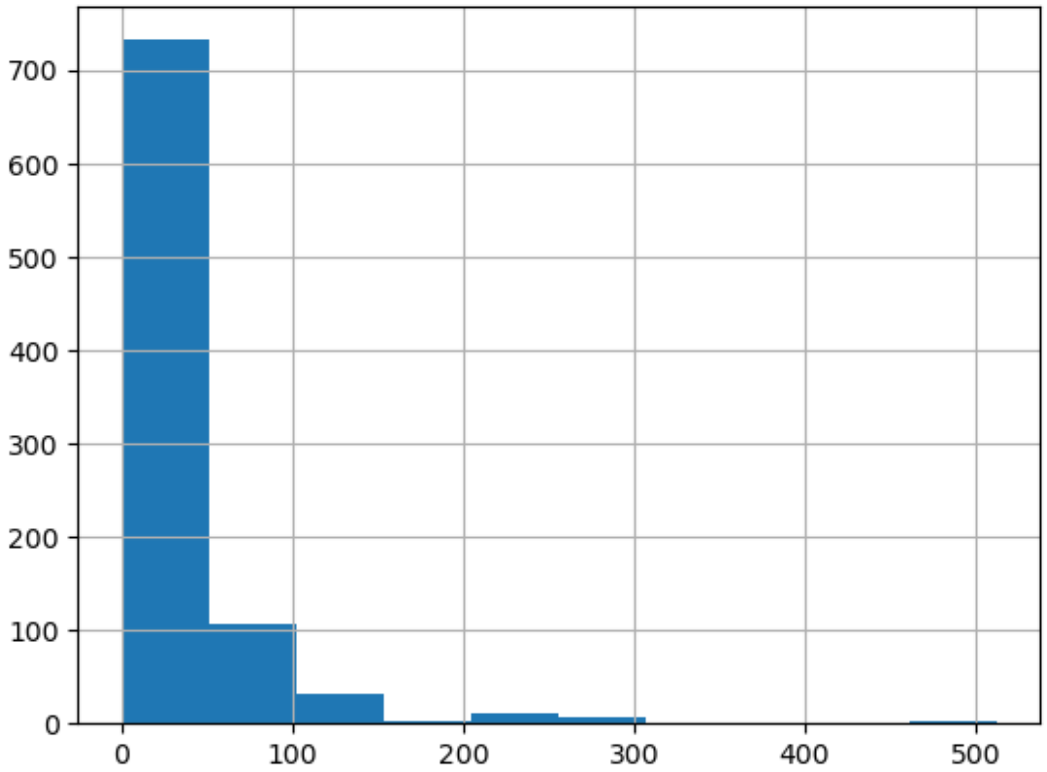
```
<seaborn.axisgrid.PairGrid at 0x2810a0d10>
```



Outlier detection and removal

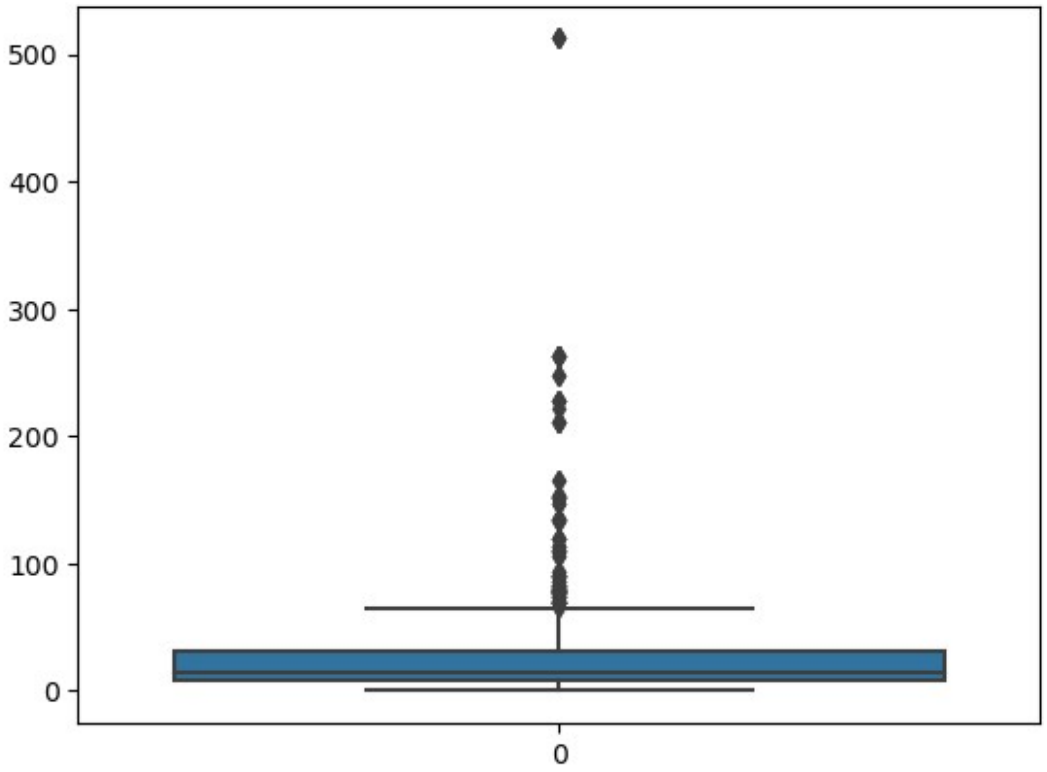
```
data['Fare'].hist()
```

```
<Axes: >
```




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

```
<Axes: >
```




```
q1=data['Fare'].quantile(0.25)
q3=data['Fare'].quantile(0.75)
iqr=q3-q1
iqr
```

```
23.0896
```

```
upper_limit=q3+1.5*iqr
lower_limit=q1-1.5*iqr
```

```
data.median()
```

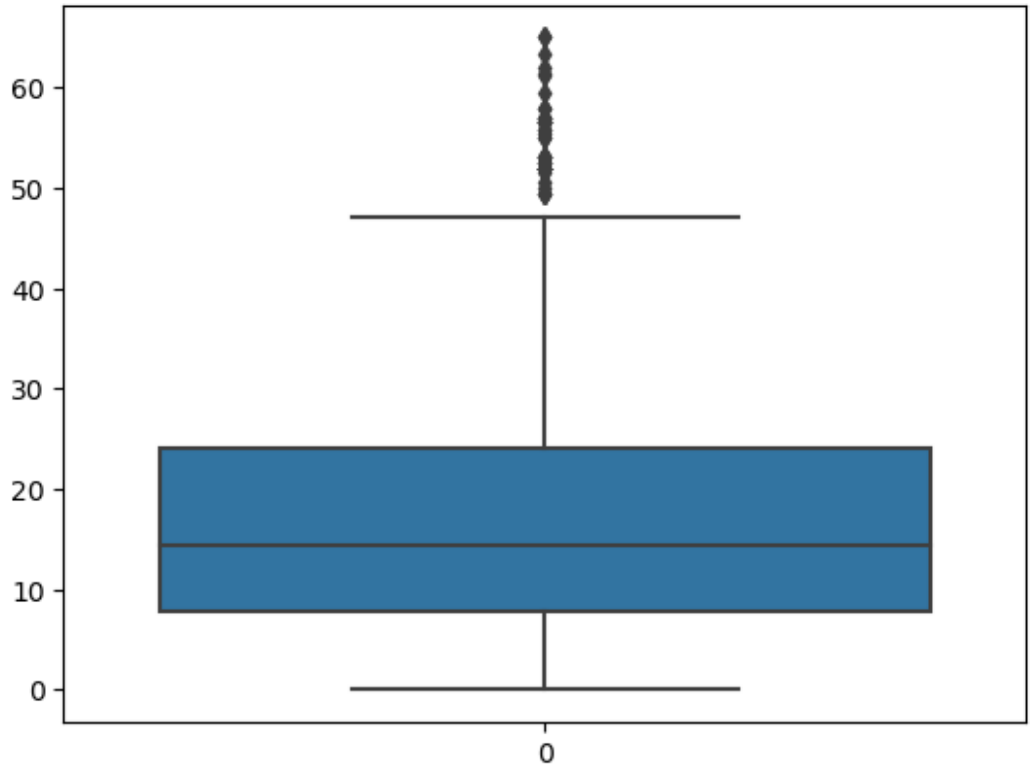
```
/var/folders/75/l1625v4n7qnbfp296v2f82ch0000gn/T/
ipykernel_6860/4184645713.py:1: FutureWarning: The default value of
numeric_only in DataFrame.median is deprecated. In a future version,
it will default to False. In addition, specifying 'numeric_only=None'
is deprecated. Select only valid columns or specify the value of
numeric_only to silence this warning.
  data.median()
```

```
Survived     0.0000
Pclass       3.0000
SibSp        0.0000
Parch        0.0000
Fare        14.4542
dtype: float64
```

```
data['Fare']=np.where(data['Fare']>upper_limit,14.4542,data['Fare'])

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

```
<Axes: >
```



splitting into dependant and independant variables

```
data.head()
```

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

```
x=data.iloc[:,1:]
y=data.iloc[:,:1]

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	14.4542	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
```

```
y.head()
```

```
   Survived
0         0
1         1
2         1
3         1
4         0
```

Ecoding

```
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  14.4542        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
```

```
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  14.4542         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)
```

Splitting into train and test dataset

```
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)
```