In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import math
titanic_data=pd.read_csv(r'Titanic-Dataset.csv')
titanic_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
```

In [2]:

```
titanic_data.head(10)
```

Out[2]:

	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 Briggs Th...	female	38.0	1	0	PC 17599	71.2833
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708

In [4]:

```
print(titanic_data.shape)
```

```
(891, 12)
```

In [5]:

```
print("# of passangers in original dataset:",len(titanic_data))
```

```
# of passangers in original dataset: 891
```

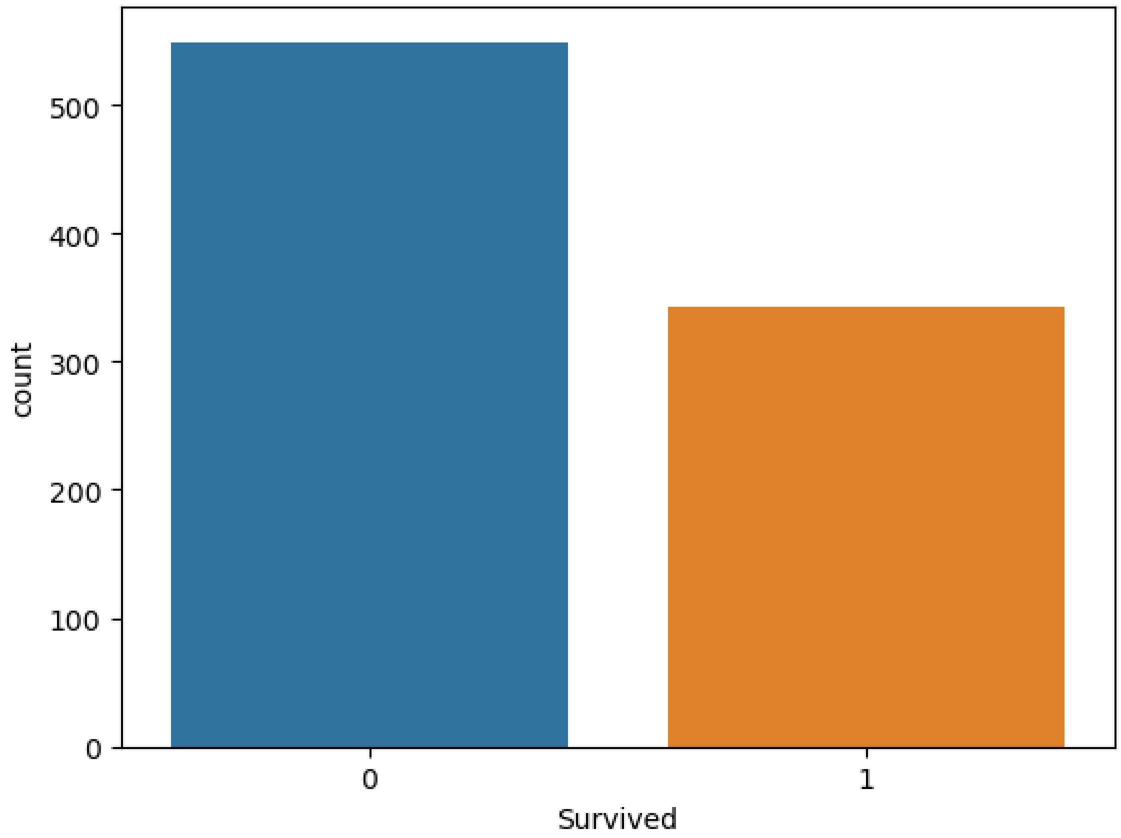
In [7]:

```
#analyzing data

sns.countplot(x="Survived",data=titanic_data)
```

Out[7]:

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

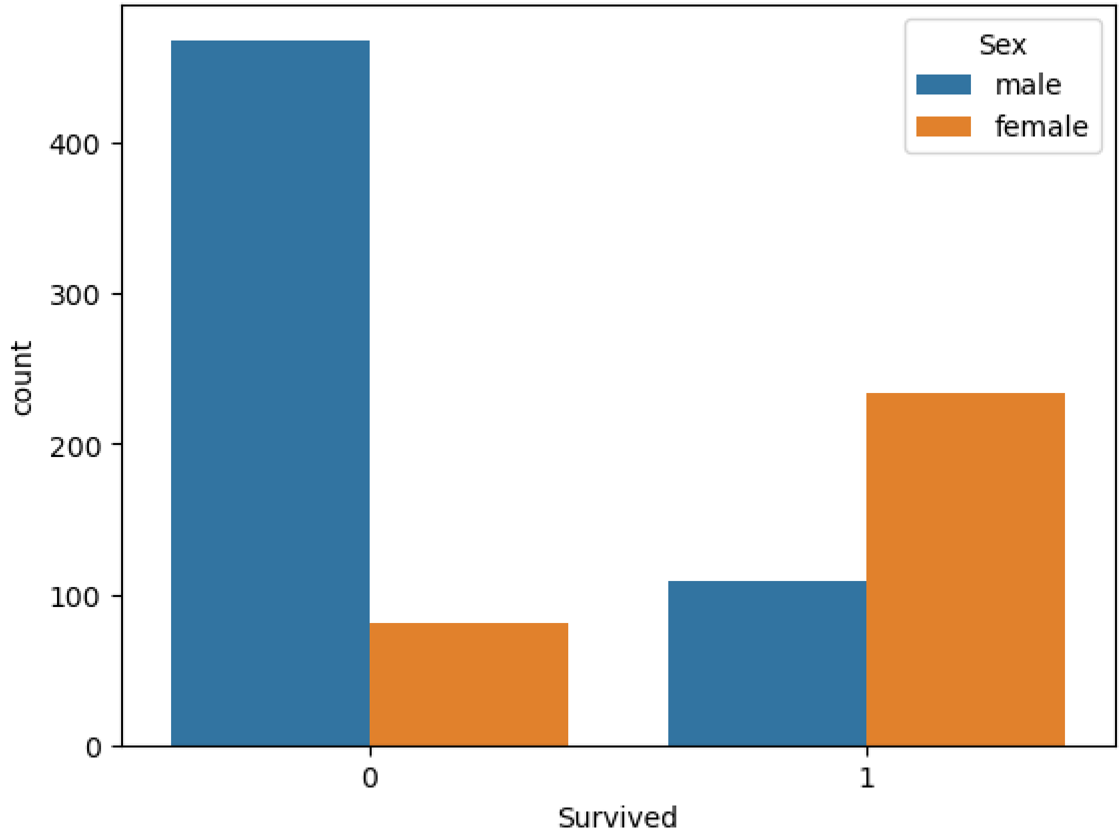


In [9]:

```
#plot to show that female passanger were more likely to survive
sns.countplot(x="Survived",hue="Sex",data=titanic_data)
```

Out[9]:

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

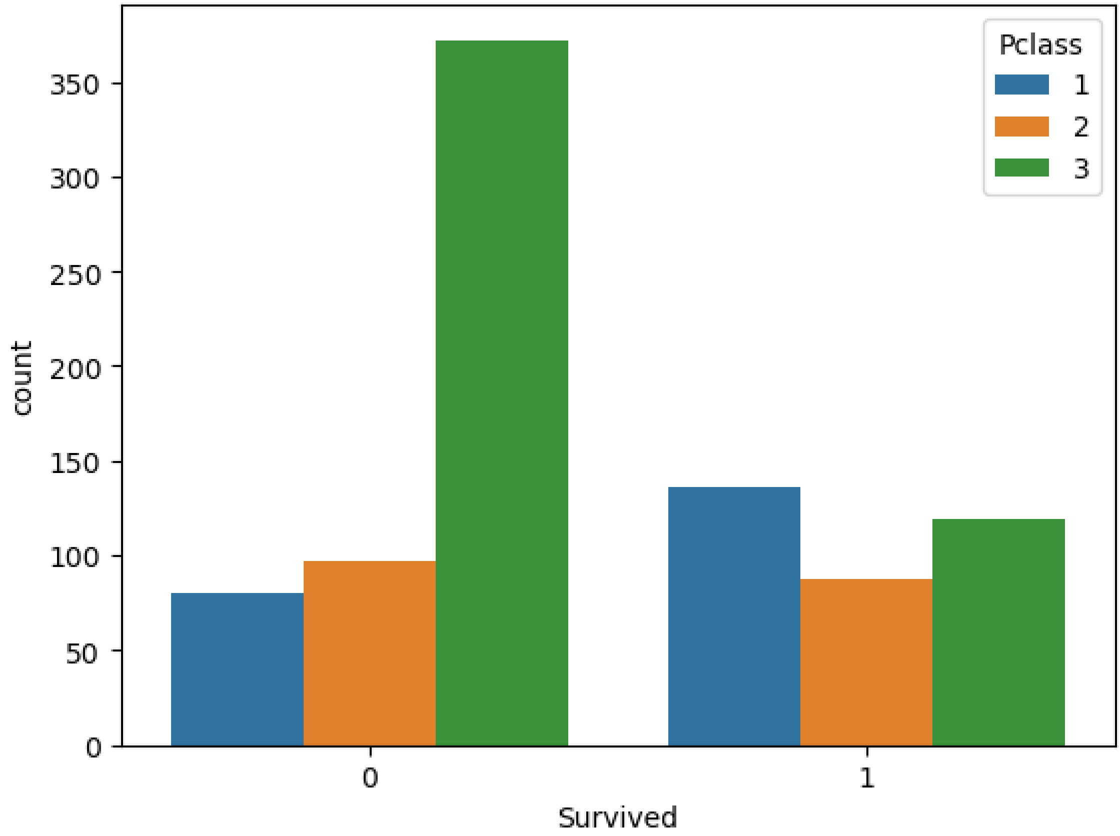


In [10]:

```
#plot to show that first class  passanger were more likely to survive
sns.countplot(x="Survived",hue="Pclass",data=titanic_data)
```

Out[10]:

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

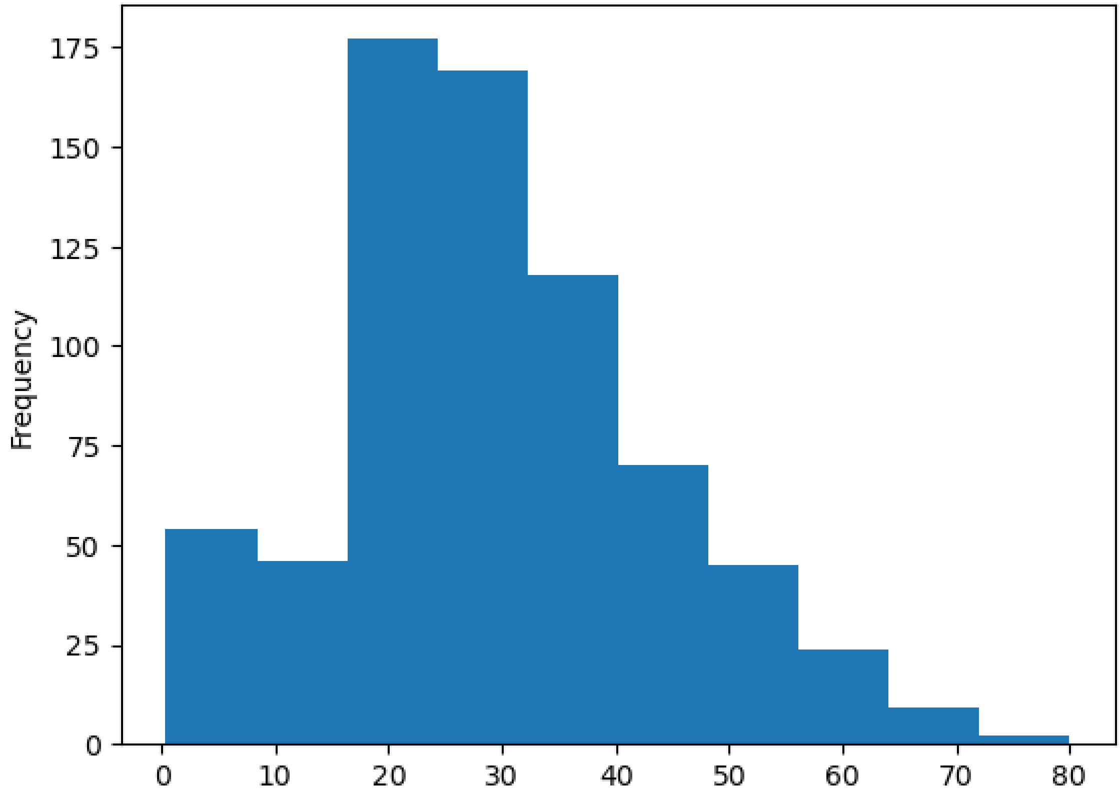


In [11]:

```
#more lower or middle age persons travelling in titanic
titanic_data["Age"].plot.hist()
```

Out[11]:

```
<Axes: ylabel='Frequency'>
```

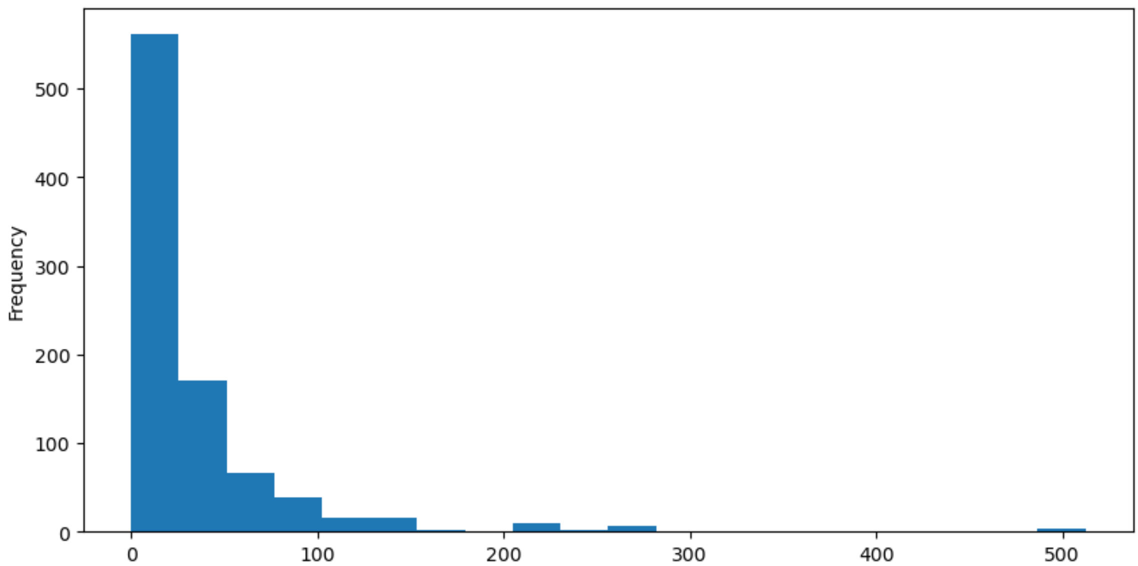


In [12]:

```
#fare value ranges mostly between 0 to 100
titanic_data["Fare"].plot.hist(bins=20,figsize=(10,5))
```

Out[12]:

```
<Axes: ylabel='Frequency'>
```

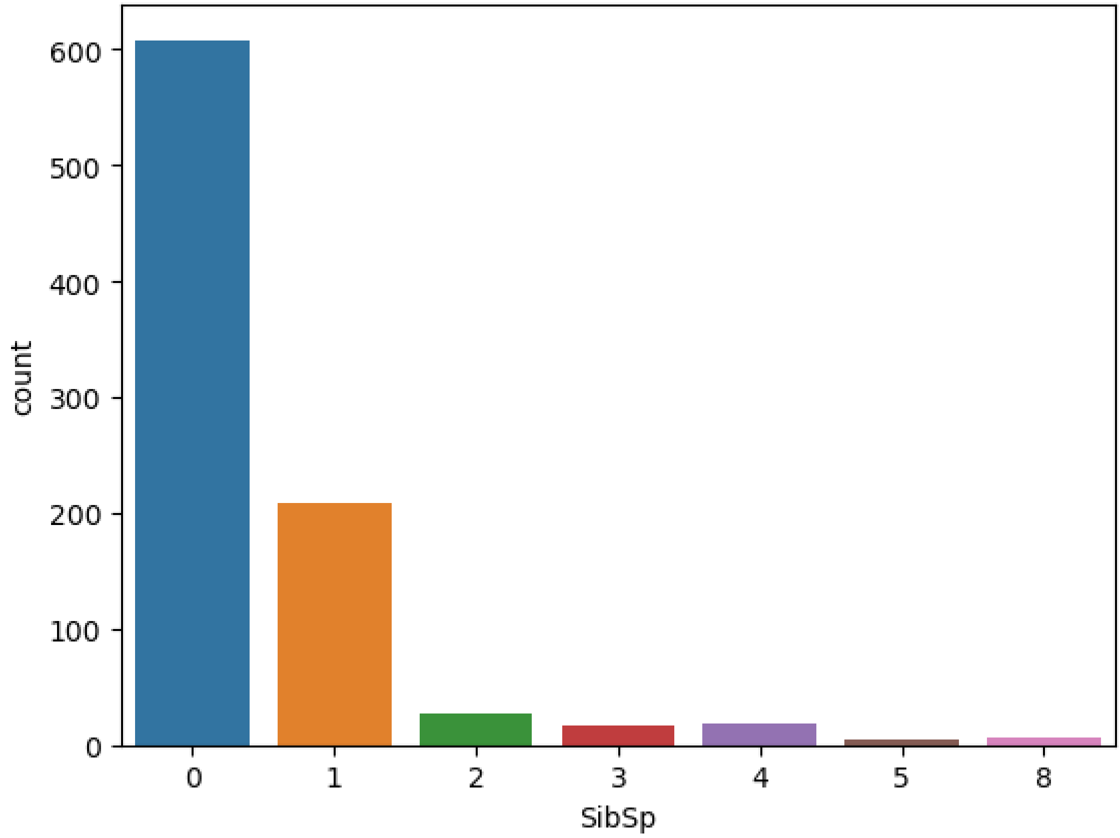


In [13]:

```
#Analyze sibling-spouse
sns.countplot(x="SibSp",data=titanic_data)
```

Out[13]:

```
<Axes: xlabel='SibSp', ylabel='count'>
```



In [14]:

```
#DATA CLEANING- Removing NaN values and neccesary column in the dataset
titanic_data.isnull()
```

Out[14]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	False	False	False	False	False	False	False	False	False	False	True
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True
...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True
887	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True
889	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True

891 rows × 12 columns

In [15]:

```
titanic_data.isnull().sum()
```

Out[15]:

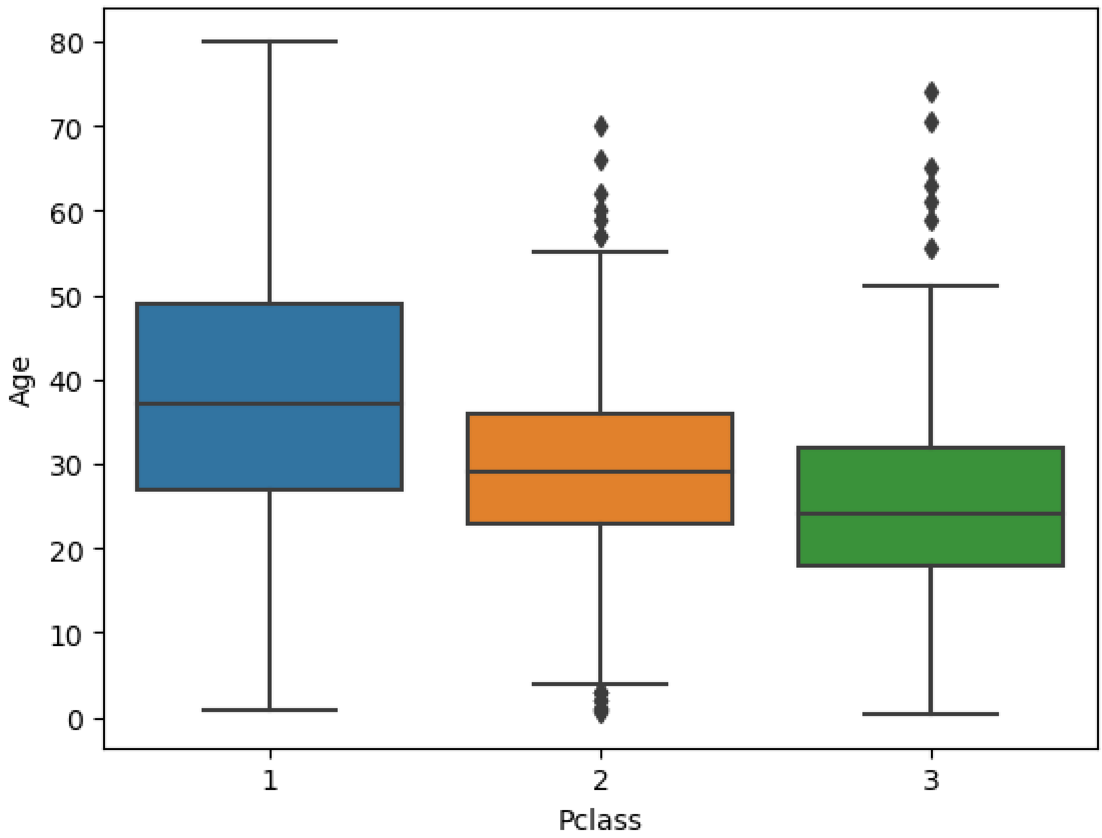
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [16]:

```
sns.boxplot(x="Pclass",y="Age",data=titanic_data)
```

Out[16]:

```
<Axes: xlabel='Pclass', ylabel='Age'>
```



In [17]:

```
titanic_data.drop("Cabin",axis=1,inplace=True)
```

In [18]:

```
titanic_data.head(2)
```

Out[18]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Eml
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 Briggs Th...	female	38.0	1	0	PC 17599	71.2833	

In [19]:

```
titanic_data.dropna(inplace=True)
print(titanic_data.shape)
```

```
(712, 11)
```

In [21]:

```
titanic_data.isnull().sum()
```

Out[21]:

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

In [22]:

```
#string values present a lot, so covert it into categorical value
pd.get_dummies(titanic_data['Sex'])
```

Out[22]:

	female	male
0	0	1
1	1	0
2	1	0
3	1	0
4	0	1
...	...	...
885	1	0
886	0	1
887	1	0
889	0	1
890	0	1

712 rows × 2 columns

In [23]:

```
sex=pd.get_dummies(titanic_data['Sex'],drop_first=True)
sex.head(5)
```

Out[23]:

	male
0	1
1	0
2	0
3	0
4	1

In [24]:

```
pd.get_dummies(titanic_data["Embarked"])
```

Out[24]:

	C	Q	S
0	0	0	1
1	1	0	0
2	0	0	1
3	0	0	1
4	0	0	1
...	...	...	...
885	0	1	0
886	0	0	1
887	0	0	1
889	1	0	0
890	0	1	0

712 rows × 3 columns

In [25]:

```
embark=pd.get_dummies(titanic_data["Embarked"],drop_first=True)
embark.head()
```

Out[25]:

	Q	S
0	0	1
1	0	0
2	0	1
3	0	1
4	0	1

In [26]:

```
pd.get_dummies(titanic_data["Pclass"])
```

Out[26]:

	1	2	3
0	0	0	1
1	1	0	0
2	0	0	1
3	1	0	0
4	0	0	1
...	...	...	...
885	0	0	1
886	0	1	0
887	1	0	0
889	1	0	0
890	0	0	1

712 rows × 3 columns

In [27]:

```
pcls=pd.get_dummies(titanic_data["Pclass"],drop_first=True)
pcls.head()
```

Out[27]:

	2	3
0	0	1
1	0	0
2	0	1
3	0	0
4	0	1

In [30]:

```
titanic_data=pd.concat([titanic_data,sex,embark,pcls],axis=1)
titanic_data.head(5)
```

Out[30]:

	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 Briggs Th...	female	38.0	1	0	PC 17599	71.2833
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500

5 rows × 21 columns

In [31]:

```
titanic_data.drop(['Sex','Embarked','PassengerId','Name','Ticket'],axis=1,inplace=True)
```

In [32]:

```
titanic_data.columns
```

Out[32]:

```
Index(['Survived',   'Pclass',      'Age',    'SibSp',    'Parch',     'Fa
re',
           'male',        'Q',        'S',          2,          3,     'ma
le',
              'Q',        'S',          2,          3],
      dtype='object')
```

In [33]:

```
titanic_data.head(2)
```

Out[33]:

	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S	2	3	male	Q	S	2	3
0	0	3	22.0	1	0	7.2500	1	0	1	0	1	1	0	1	0	1
1	1	1	38.0	1	0	71.2833	0	0	0	0	0	0	0	0	0	0

In [34]:

```
X=titanic_data.drop("Survived",axis=1)
y=titanic_data["Survived"]
```

In [35]:

```
print(X)
```

```
     Pclass   Age  SibSp  Parch     Fare  male  Q  S  2  3  male  Q  S  2  
3
0         3  22.0      1      0   7.2500     1  0  1  0  1     1  0  1  0  
1
1         1  38.0      1      0  71.2833     0  0  0  0  0     0  0  0  0  
0
2         3  26.0      0      0   7.9250     0  0  1  0  1     0  0  1  0  
1
3         1  35.0      1      0  53.1000     0  0  1  0  0     0  0  1  0  
0
4         3  35.0      0      0   8.0500     1  0  1  0  1     1  0  1  0  
1
..      ...   ...    ...    ...      ...   ... .. .. .. ..   ... .. .. .. 
..
885       3  39.0      0      5  29.1250     0  1  0  0  1     0  1  0  0  
1
886       2  27.0      0      0  13.0000     1  0  1  1  0     1  0  1  1  
0
887       1  19.0      0      0  30.0000     0  0  1  0  0     0  0  1  0  
0
889       1  26.0      0      0  30.0000     1  0  0  0  0     1  0  0  0  
0
890       3  32.0      0      0   7.7500     1  1  0  0  1     1  1  0  0  
1

[712 rows x 15 columns]
```

In [37]:

```
X=titanic_data.iloc[:,[1,2,3,4,5,6,7,8,9]].values
Y=titanic_data.iloc[:,0].values
print(X)
```

```
[[ 3. 22.  1. ...  0.  1.  0.]
 [ 1. 38.  1. ...  0.  0.  0.]
 [ 3. 26.  0. ...  0.  1.  0.]
 ...
 [ 1. 19.  0. ...  0.  1.  0.]
 [ 1. 26.  0. ...  0.  0.  0.]
 [ 3. 32.  0. ...  1.  0.  0.]]
```

In [38]:

```
print(Y)
```

```
[0 1 1 1 0 0 0 1 1 1 1 0 0 0 1 0 0 0 1 1 1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 0 0
 0 1 1 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 1 0 1 0 1 1 0 1 0 0 0 0 0 0 0
 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0 1 0 0 0
 1 1 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1
 0 0 1 0 1 1 1 1 0 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 1 0 1 0 0 1
 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 1 0
 0 1 0 0 0 1 0 1 0 1 1 1 1 0 0 0 0 0 1 0 1 1 0 1 1 1 0 0 0 1 1 0 1 1 0 0 1
 1 1 0 1 1 1 0 0 0 0 1 1 0 1 1 0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 0
 0 0 0 1 0 0 0 1 1 0 1 0 0 1 1 1 1 0 1 1 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1 1 1
 1 0 0 0 0 0 0 1 1 1 1 1 0 0 1 0 1 0 0 1 0 0 1 1 1 1 1 1 0 0 1 1 0 1 1 0 0
 0 0 0 1 0 1 1 0 0 0 0 1 0 0 1 1 1 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 1 0 1 1 1
 1 0 0 1 1 0 1 0 1 0 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1 0 1 1 0 1 1 1
 0 0 0 0 0 1 1 1 1 0 0 1 1 1 1 1 0 0 1 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 1 1 1
 0 0 1 0 0 1 0 0 1 1 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 1 0 1 1 0 1 1 1 0 0 0 0
 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1
 1 0 0 0 0 1 1 1 1 1 0 0 0 1 1 0 1 0 0 0 1 0 1 0 0 1 0 0 0 0 0 1 0 1 0 1 0
 0 1 0 0 1 1 0 0 1 1 0 0 0 1 0 1 1 0 1 0 0 0 0 0 1 0 1 1 1 1 0 0 0 1 0 1 0
 0 0 0 1 1 0 0 0 1 1 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 1 1 0 0 1
 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 1 1 1 1 0 0 1 0 1 1 0 1 0 1 0 0 1 1 0 0 1 1
 0 0 0 0 0 0 1 1 0]
```

In [39]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.25,random_state=1)
```

In [40]:

```
#Feature scalling
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
X_train=sc.fit_transform(X_train)
X_test-sc.transform(X_test)
print(X_train)
print(Y_train)
```

```
[[-1.43765909  1.27600045  0.51248812 ... -0.21693046 -1.78097586
  -0.55571893]
 [-0.25656685  0.33402559 -0.55653007 ... -0.21693046  0.56148993
   1.79947082]
 [-1.43765909  1.4778522  -0.55653007 ... -0.21693046  0.56148993
  -0.55571893]
 ...
 [ 0.92452538  1.94883963 -0.55653007 ... -0.21693046  0.56148993
  -0.55571893]
 [-0.25656685  0.40130951 -0.55653007 ... -0.21693046 -1.78097586
   1.79947082]
 [ 0.92452538 -0.60794926 -0.55653007 ... -0.21693046  0.56148993
  -0.55571893]]
[1 0 1 1 1 1 0 0 1 0 0 0 1 0 0 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0
 0 0 0 0 0 0 1 0 1 1 0 1 0 1 1 0 0 0 1 1 0 1 1 1 0 1 0 0 1 0 1 0 0 0 1 1 0
 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 0 1 0 1 1 1 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0
 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0 1 1 1 1 0 0 0 1 0 0 1 1 0 0 0 0
 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1 0 1 0 1 1 0 1 1 0 0 1 0 1 0 1 1 0
 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 1 1 1 0 0 0 1 0 1 0 0
 0 1 1 1 1 1 1 0 0 1 1 1 0 0 0 0 1 0 1 1 1 1 0 1 0 1 0 0 1 1 1 1 1 1 0 0 0
 0 0 1 0 0 0 1 0 0 1 1 1 0 1 1 1 0 0 0 1 1 0 1 1 0 0 0 0 0 1 1 0 0 1 1 0 0
 0 0 0 1 0 1 0 1 0 1 1 1 1 1 0 0 0 0 0 1 0 1 0 1 0 0 1 0 0 0 0 0 0 1 0 1 0
 0 1 1 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 1 1 0 1 1
 1 0 0 1 1 1 0 0 1 0 0 1 0 0 1 0 1 1 1 1 0 0 0 0 0 0 1 1 1 0 0 1 0 0 1 1 0
 0 1 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1 1 0 0 1 0 0 0 0 1 1 0 1 0 1
 0 1 0 1 0 0 0 1 0 0 1 1 0 1 0 0 0 0 1 0 0 1 1 1 1 0 1 1 0 0 1 0 0 1 0 0 0
 1 0 0 1 1 1 0 1 1 0 0 0 0 1 1 0 1 0 1 1 0 1 0 1 0 1 0 0 0 1 0 1 1 1 0 0 1
 1 0 1 1 0 0 0 0 1 0 0 0 1 0 0 0]
```

In [41]:

```
from sklearn.linear_model import LogisticRegression
```

In [42]:

```
classifier=LogisticRegression()
```

In [43]:

```
classifier.fit(X_train,Y_train)
```

Out[43]:

```
▾ LogisticRegression
LogisticRegression()
```

In [44]:

```
Y_pred=classifier.predict(X_test)
```

In [46]:

```
from sklearn.metrics import classification_report
print(classification_report(Y_test,Y_pred))
```

```
              precision    recall  f1-score   support

           0       0.59      0.98      0.74       102
           1       0.75      0.08      0.14        76

    accuracy                           0.60       178
   macro avg       0.67      0.53      0.44       178
weighted avg       0.66      0.60      0.48       178
```

In [47]:

```
from sklearn.metrics import confusion_matrix
confusion_matrix(Y_test,Y_pred)
```

Out[47]:

```
array([[100,   2],
       [ 70,   6]], dtype=int64)
```

In [51]:

```
from sklearn.metrics import accuracy_score
accuracy_score(Y_test,Y_pred)
```

Out[51]:

```
0.5955056179775281
```

In [53]:

```
classifier.score(X_train,Y_train)
```

Out[53]:

```
0.8033707865168539
```

In [54]:

```
classifier.score(X_test,Y_test)
```

Out[54]:

```
0.5955056179775281
```

In [ ]:

```

```