In [23]:
```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
td = pd.read_csv(r'C:\Users\BABJEE\Downloads\archive\tested.csv')
td
```

Out[23]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

418 rows × 12 columns

In [24]:
```python
td.shape
```

Out[24]:
```
(418, 12)
```

In [25]:
```python
td.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  418 non-null    int64  
 1   Survived     418 non-null    int64  
 2   Pclass       418 non-null    int64  
 3   Name         418 non-null    object 
 4   Sex          418 non-null    object 
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64  
 7   Parch        418 non-null    int64  
 8   Ticket       418 non-null    object 
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object 
 11  Embarked     418 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

In [26]:
```python
td.isnull().sum()
```

Out[26]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [27]:
```python
td=td.drop(columns='Cabin',axis=1)
```

In [29]:
```python
td['Age'].fillna(td['Age'].mean(),inplace=True)
```

In [30]:
```python
td['Embarked'].fillna(td['Embarked'].mode()[0],inplace=True)
```

In [31]:
```python
td['Fare'].fillna(td['Fare'].mode()[0],inplace=True)
```

In [32]:
```python
td.isnull().sum().sum()
```

Out[32]:
```
0
```

In [34]:
```python
td['Survived'].value_counts()
```

Out[34]:
```
Survived
0    266
1    152
Name: count, dtype: int64
```

In [35]:
```python
td.describe()
```

Out[35]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	418.000000	418.000000	418.000000	418.000000	418.000000
mean	1100.500000	0.363636	2.265550	30.272590	0.447368	0.392344	35.560497
std	120.810458	0.481622	0.841838	12.634534	0.896760	0.981429	55.857145
min	892.000000	0.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	0.000000	1.000000	23.000000	0.000000	0.000000	7.895800
50%	1100.500000	0.000000	3.000000	30.272590	0.000000	0.000000	14.454200
75%	1204.750000	1.000000	3.000000	35.750000	1.000000	0.000000	31.471875
max	1309.000000	1.000000	3.000000	76.000000	8.000000	9.000000	512.329200

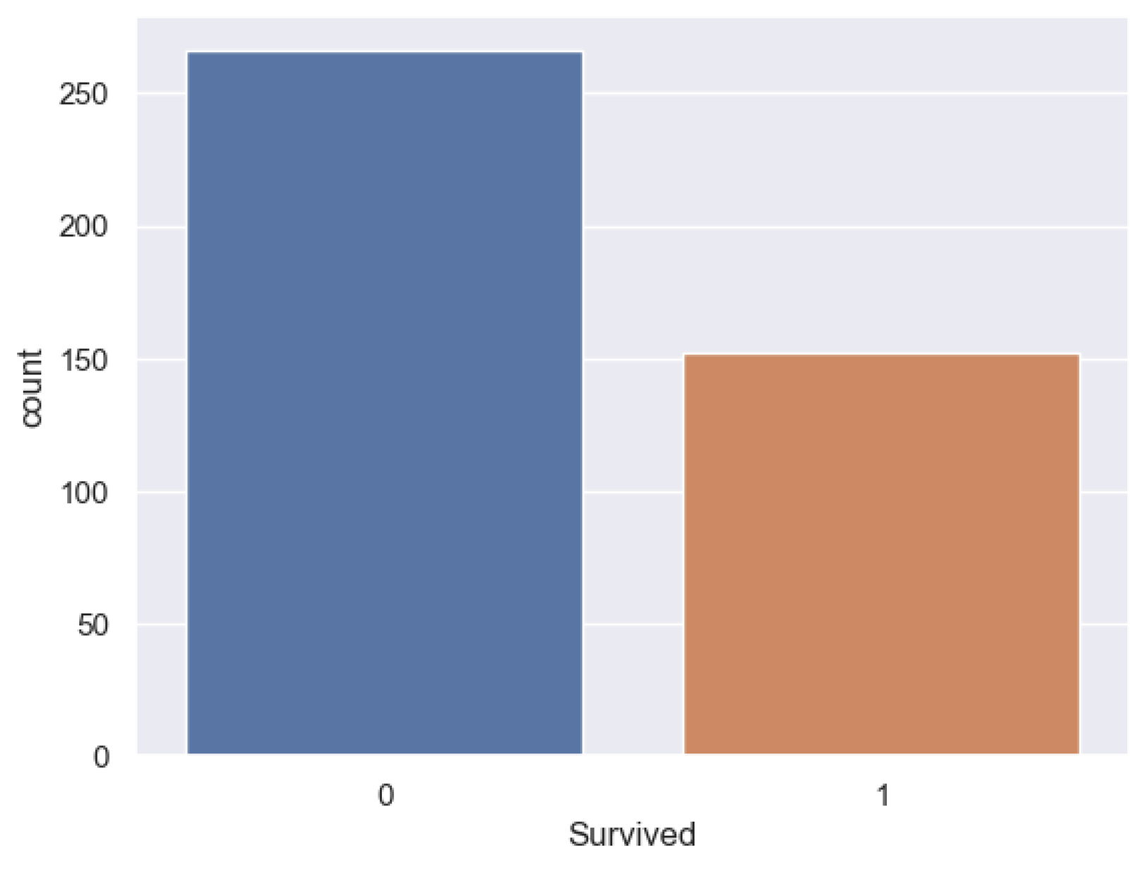
In [37]:
```python
import matplotlib.pyplot as plt
import seaborn as sns
```

```
Matplotlib is building the font cache; this may take a moment.
```

In [38]:
```python
sns.set()
```

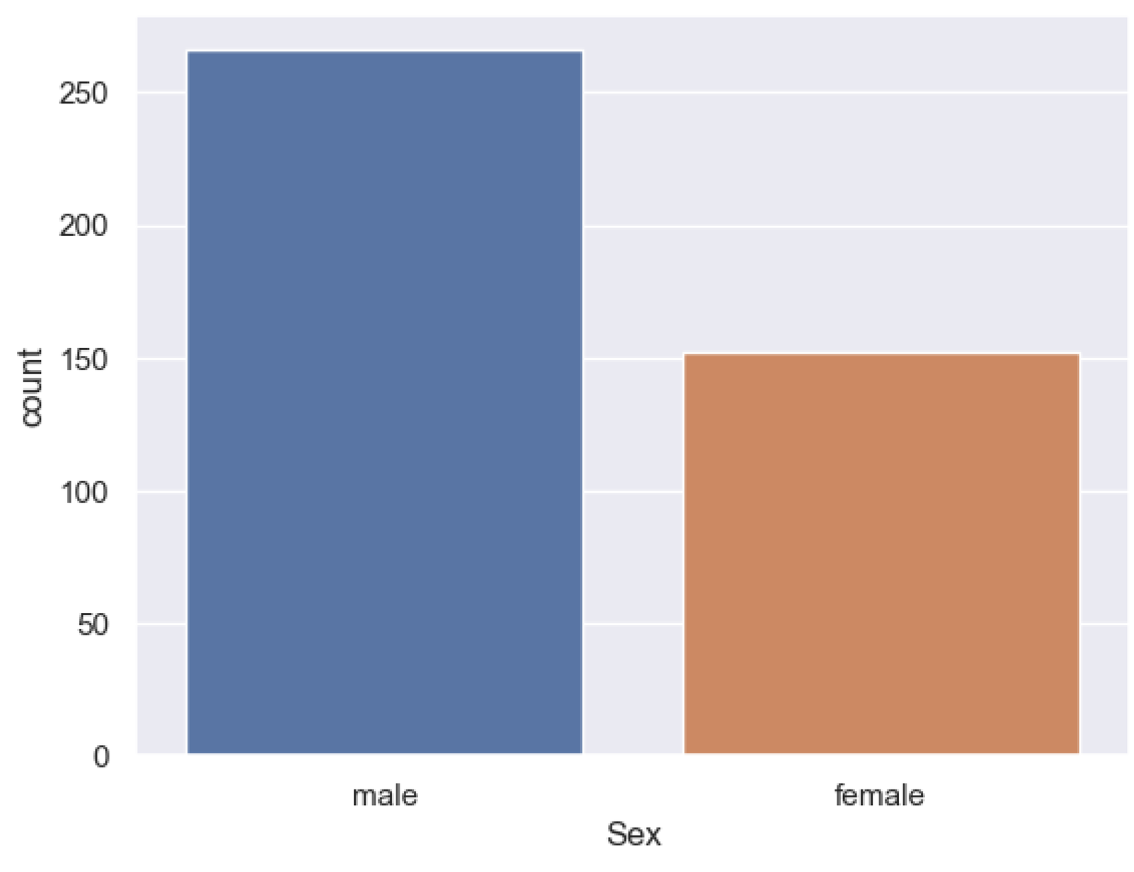
In [39]:
```python
sns.countplot(x='Survived',data=td)
```

Out[39]:
```
<Axes: xlabel='Survived', ylabel='count'>
```



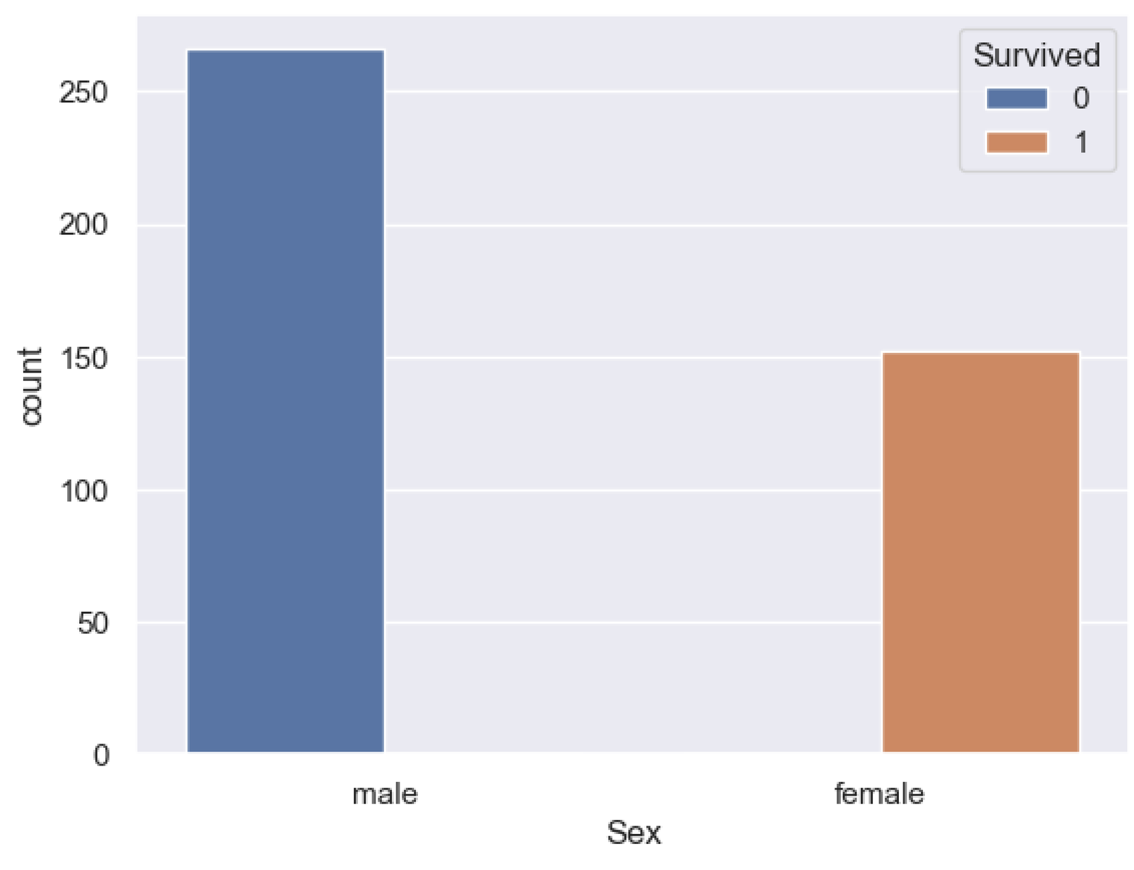
In [41]:
```python
sns.countplot(x='Sex',data=td)
```

Out[41]:
```
<Axes: xlabel='Sex', ylabel='count'>
```



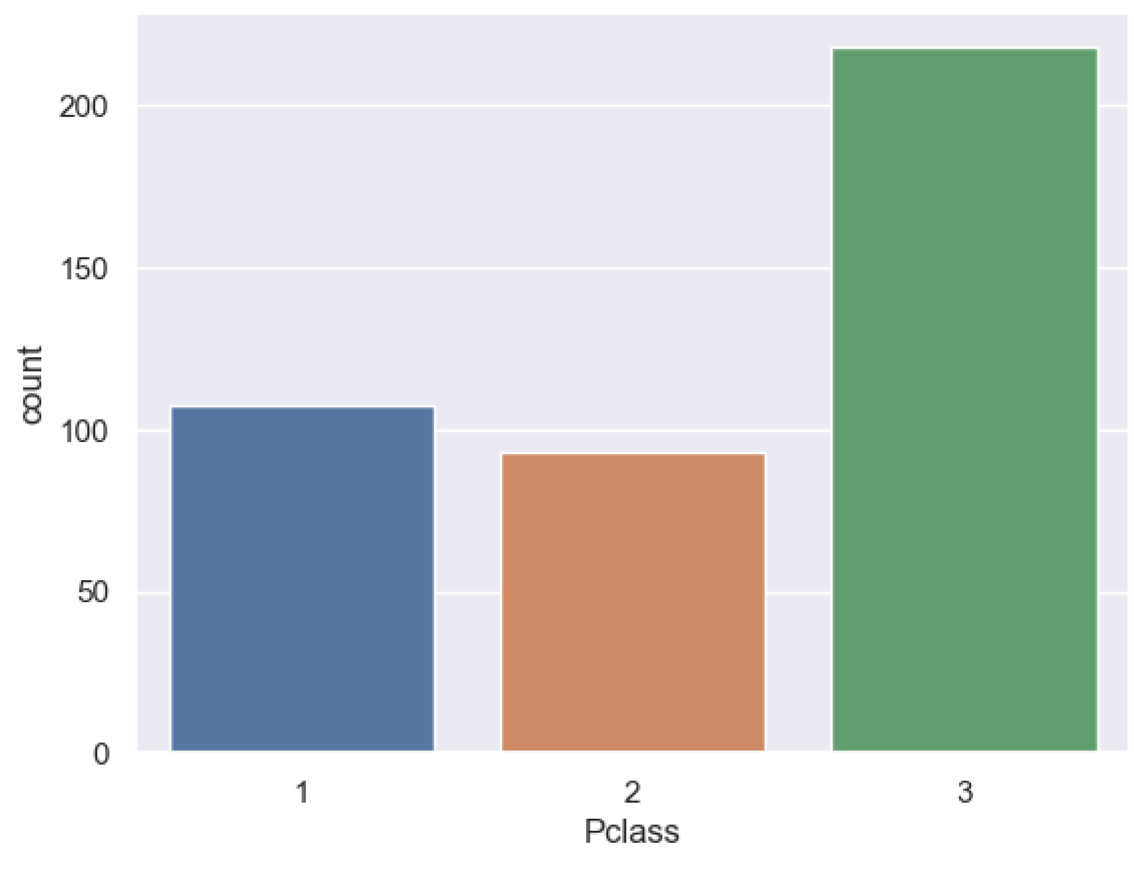
In [42]:
```python
sns.countplot(x='Sex',hue='Survived',data=td)
```

Out[42]:
```
<Axes: xlabel='Sex', ylabel='count'>
```



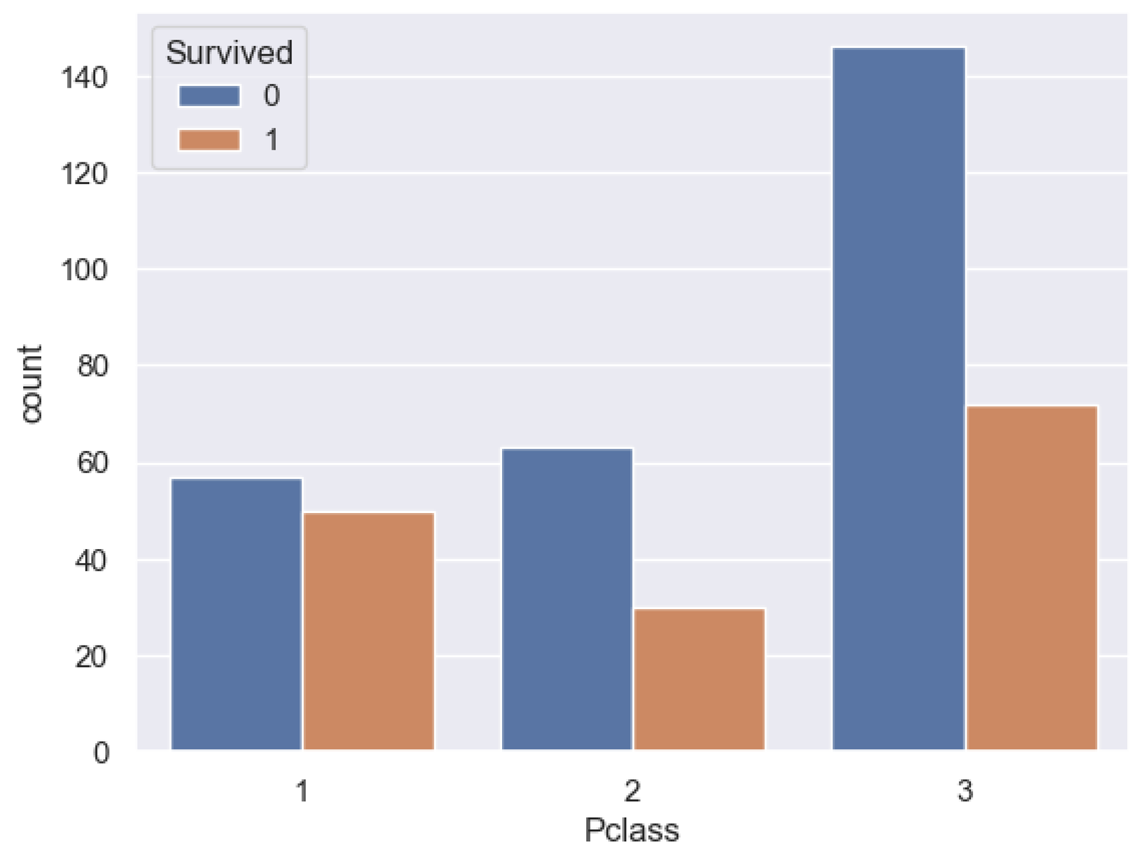
In [43]:
```python
sns.countplot(x='Pclass',data=td)
```

Out[43]:
```
<Axes: xlabel='Pclass', ylabel='count'>
```



In [44]:
```python
sns.countplot(x='Pclass',hue='Survived',data=td)
```

Out[44]:
```
<Axes: xlabel='Pclass', ylabel='count'>
```



In [45]:
```python
td['Sex'].value_counts()
```

Out[45]:
```
Sex
male      266
female    152
Name: count, dtype: int64
```

In [46]:
```python
td['Embarked'].value_counts()
```

Out[46]:
```
Embarked
S    270
C    102
Q     46
Name: count, dtype: int64
```

In [47]:
```python
td.replace({'Sex':{'male':0,'female':1},'Embarked':{'S':0,'C':1,'Q':2}},inplace=True)
td
```

Out[47]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	892	0	3	Kelly, Mr. James	0	34.50000	0	0	330911	7.8292	2
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	1	47.00000	1	0	363272	7.0000	0
2	894	0	2	Myles, Mr. Thomas Francis	0	62.00000	0	0	240276	9.6875	2
3	895	0	3	Wirz, Mr. Albert	0	27.00000	0	0	315154	8.6625	0
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	1	22.00000	1	1	3101298	12.2875	0
...	...	...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	0	30.27259	0	0	A.5. 3236	8.0500	0
414	1306	1	1	Oliva y Ocana, Dona. Fermina	1	39.00000	0	0	PC 17758	108.9000	1
415	1307	0	3	Saether, Mr. Simon Sivertsen	0	38.50000	0	0	SOTON/O.Q. 3101262	7.2500	0
416	1308	0	3	Ware, Mr. Frederick	0	30.27259	0	0	359309	8.0500	0
417	1309	0	3	Peter, Master. Michael J	0	30.27259	1	1	2668	22.3583	1

418 rows × 11 columns

In [49]:
```python
X=td.drop(columns=['PassengerId','Name','Ticket'],axis=1)
Y=td['Survived']
print(X)
print(Y)
```

```
     Survived  Pclass  Sex       Age  SibSp  Parch      Fare  Embarked
0           0       3    0  34.50000      0      0    7.8292         2
1           1       3    1  47.00000      1      0    7.0000         0
2           0       2    0  62.00000      0      0    9.6875         2
3           0       3    0  27.00000      0      0    8.6625         0
4           1       3    1  22.00000      1      1   12.2875         0
..        ...     ...  ...       ...    ...    ...       ...       ...
413         0       3    0  30.27259      0      0    8.0500         0
414         1       1    1  39.00000      0      0  108.9000         1
415         0       3    0  38.50000      0      0    7.2500         0
416         0       3    0  30.27259      0      0    8.0500         0
417         0       3    0  30.27259      1      1   22.3583         1

[418 rows x 8 columns]
0      0
1      1
2      0
3      0
4      1
      ..
413    0
414    1
415    0
416    0
417    0
Name: Survived, Length: 418, dtype: int64
```

In [50]:
```python
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=2)
print(X.shape,X_train.shape,X_test.shape)
```

```
(418, 8) (334, 8) (84, 8)
```

In [53]:
```python
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
model.fit(X_train,Y_train)
```

Out[53]:
```
▾ LogisticRegression
LogisticRegression()
```

In [54]:
```python
X_train_prediction=model.predict(X_train)
print(X_train_prediction)
```

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

In [56]:
```python
train_data_accuracy=accuracy_score(Y_train,X_train_prediction)
print('Accuracy Score of training data:',train_data_accuracy)
```

```
Accuracy Score of training data: 1.0
```

In [57]:
```python
X_test_prediction=model.predict(X_test)
print(X_test_prediction)
```

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

In [59]:
```python
test_data_accuracy=accuracy_score(Y_test,X_test_prediction)
print('Acuuracy score of testing data:',test_data_accuracy)
```

```
Acuuracy score of testing data: 1.0
```

In [ ]: