In [6]:

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

Extracting the dataset

In [11]:

```
df=pd.read_csv('Titanic.csv')
df.head()
```

Out[11]:

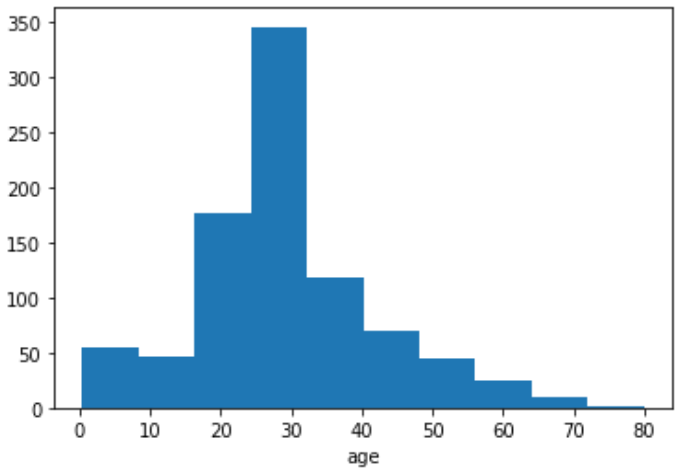
	survived	pclass	age	sibsp	parch	fare	male	age_was_missing	embarked_from_cherbourg	embarked_from_queensto
0	0	3	22.0	1	0	7.2500	1	False	0	
1	1	1	38.0	1	0	71.2833	0	False	1	
2	1	3	26.0	0	0	7.9250	0	False	0	
3	1	1	35.0	1	0	53.1000	0	False	0	
4	0	3	35.0	0	0	8.0500	1	False	0	

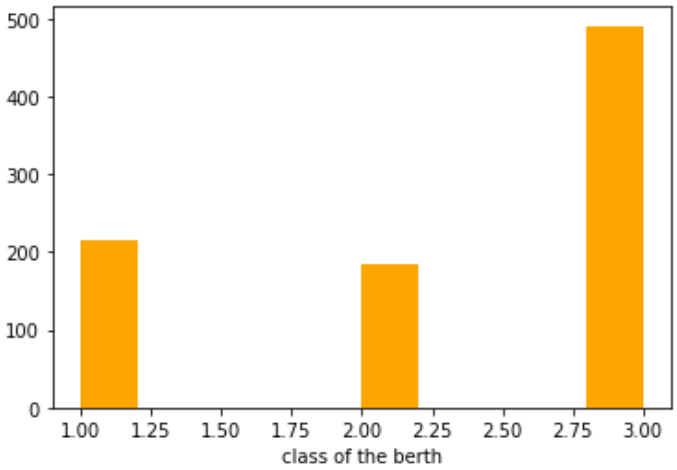
Univariate Analysis :

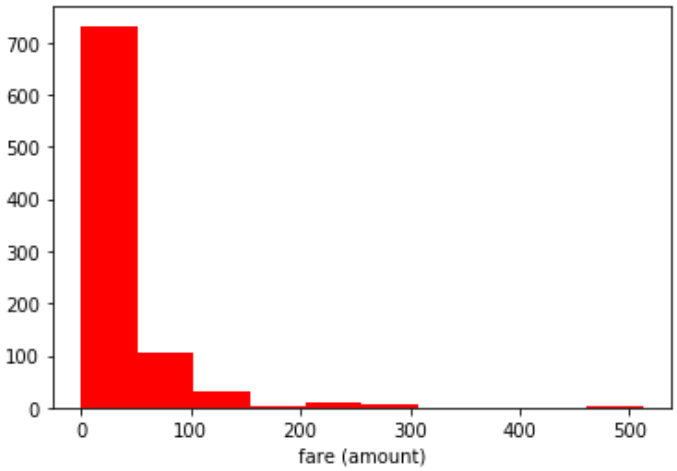
Histograms

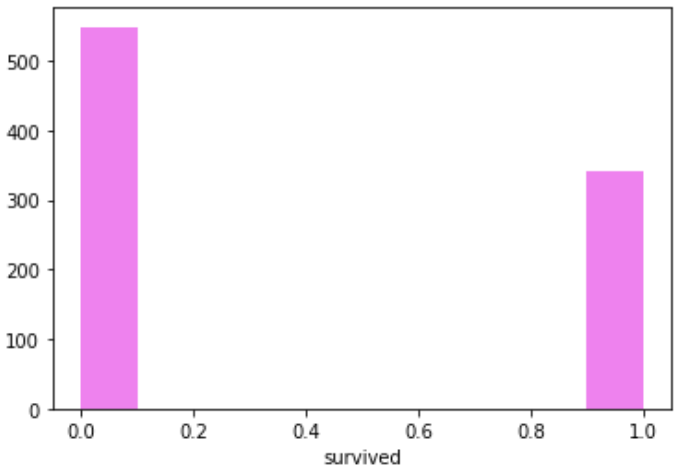
In [200]:

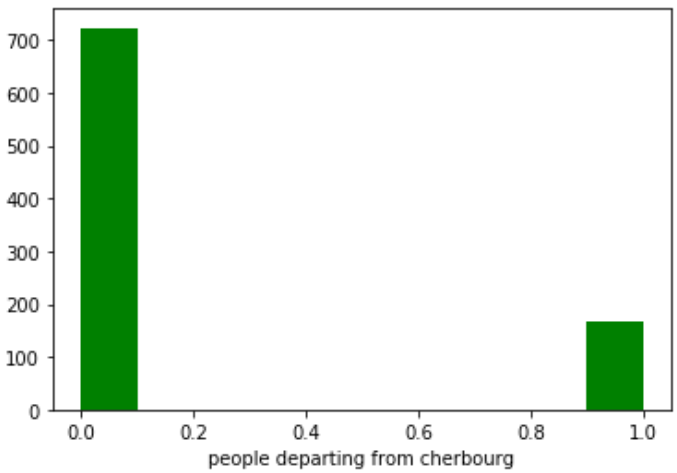
```
plt.hist(df['age'])
plt.xlabel('age')
plt.show()
plt.hist(df['pclass'],color='orange')
plt.xlabel('class of the berth')
plt.show()
plt.hist(df['fare'],color='red')
plt.xlabel('fare (amount)')
plt.show()
plt.hist(df['survived'],color='violet')
plt.xlabel('survived')
plt.show()
plt.hist(df['embarked_from_cherbourg'],color='green')
plt.xlabel('people departing from cherbourg')
plt.show()
```











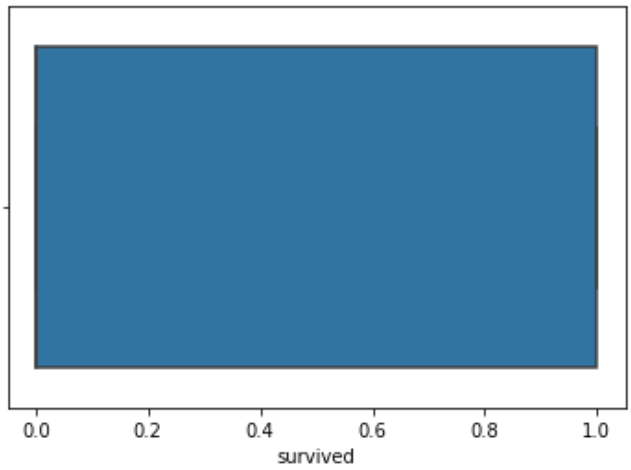
Boxplots for each numerical column

In [9]:

```
sns.boxplot(df['survived'])
```

Out[9]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33efe7f98>
```

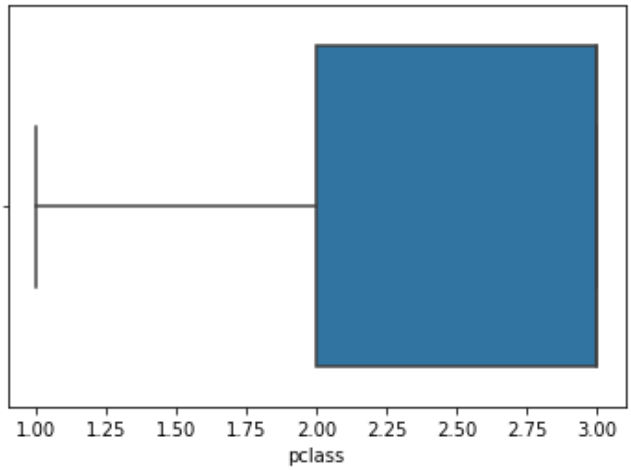


In [13]:

```
sns.boxplot(df['pclass'])
```

Out[13]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f0a93c8>
```

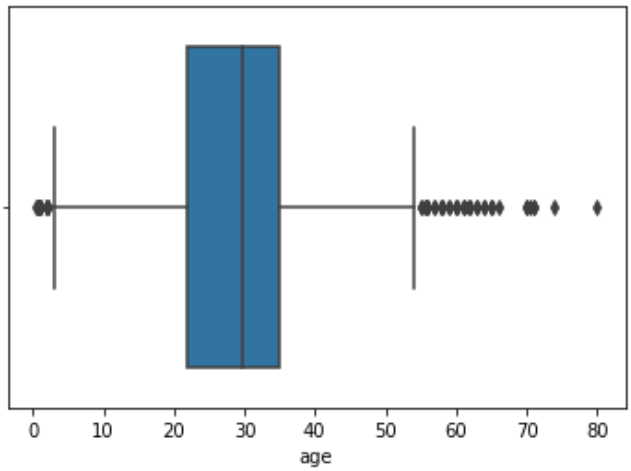


In [14]:

```
sns.boxplot(df['age'])
```

Out[14]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f1140b8>
```

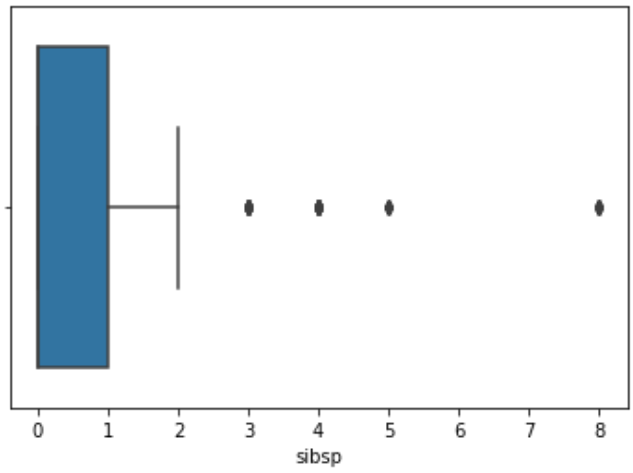


In [15]:

```
sns.boxplot(df['sibsp'])
```

Out[15]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f168128>
```

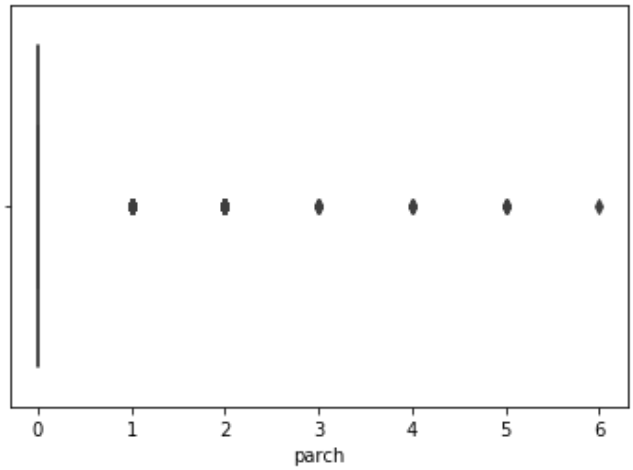


In [16]:

```
sns.boxplot(df['parch'])
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f1e6f98>
```

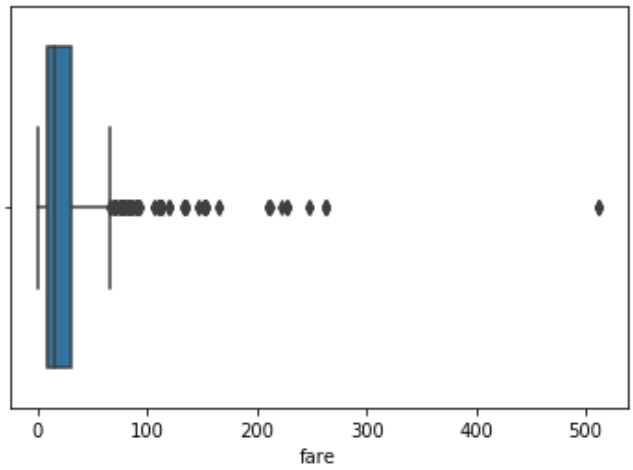


In [17]:

```
sns.boxplot(df['fare'])
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33e32fbe0>
```

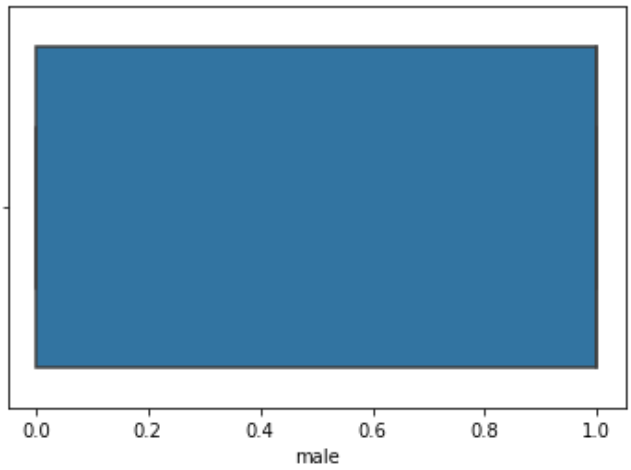


In [18]:

```
sns.boxplot(df['male'])
```

Out[18]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33e391d30>
```

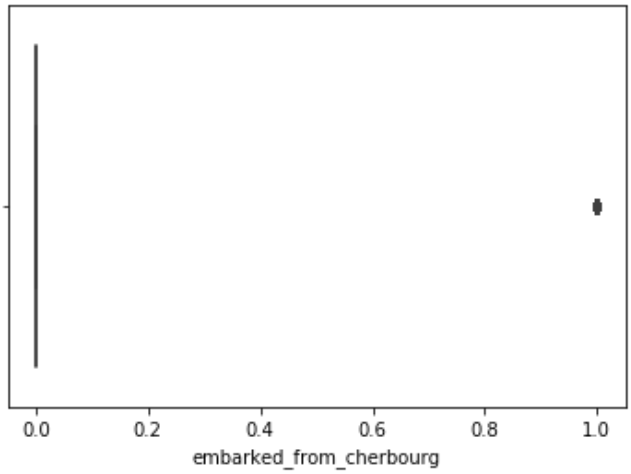


In [19]:

```
sns.boxplot(df['embarked_from_cherbourg'])
```

Out[19]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f211940>
```

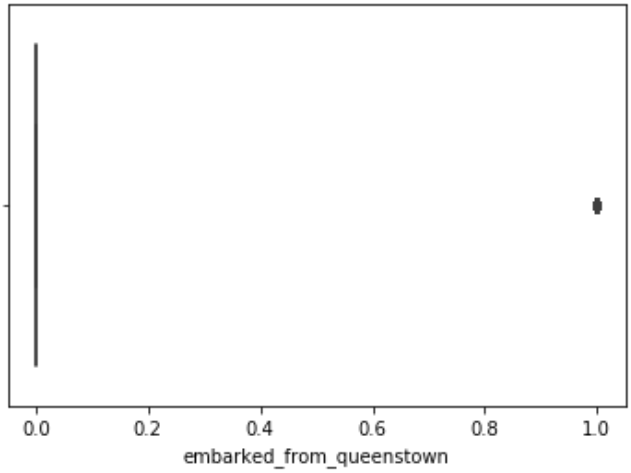


In [20]:

```
sns.boxplot(df['embarked_from_queenstown'])
```

Out[20]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f27a438>
```

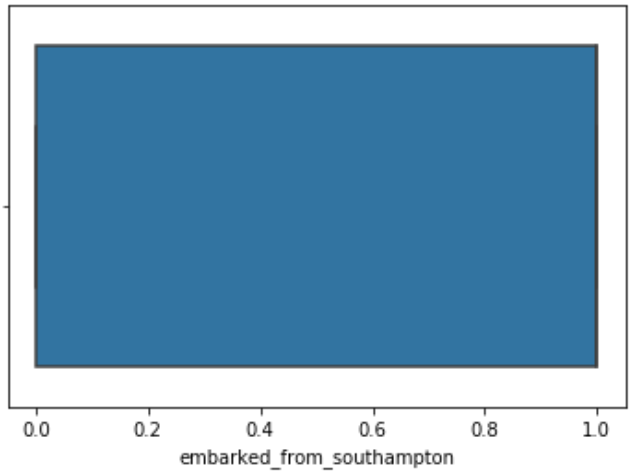


In [21]:

```
sns.boxplot(df['embarked_from_southampton'])
```

Out[21]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f2d03c8>
```



Bivariate Analysis :

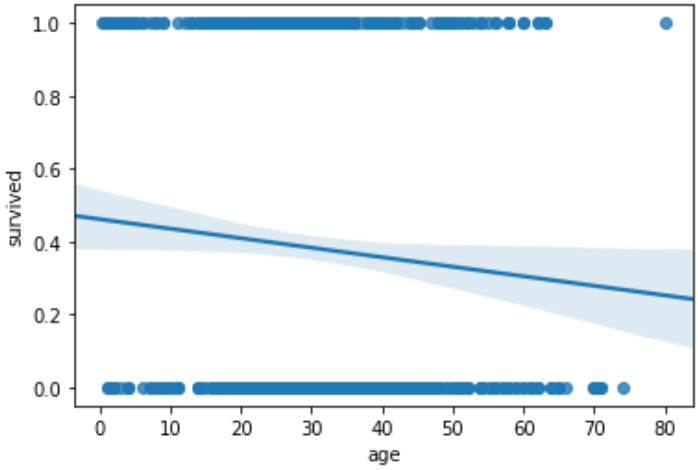
Scatter Plot

In [22]:

```
sns.regplot(x=df['age'],y=df['survived'])
```

Out[22]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f336fd0>
```

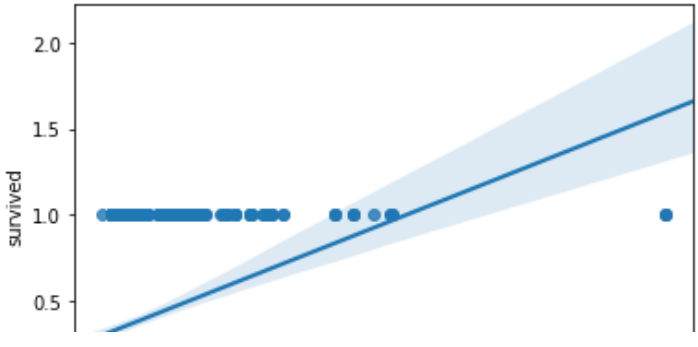


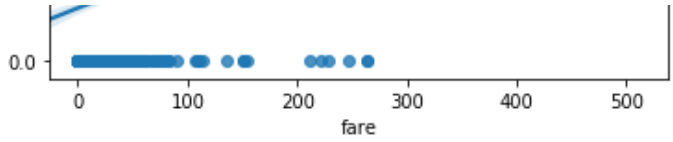
In [23]:

```
sns.regplot(x=df['fare'],y=df['survived'])
```

Out[23]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f361a58>
```



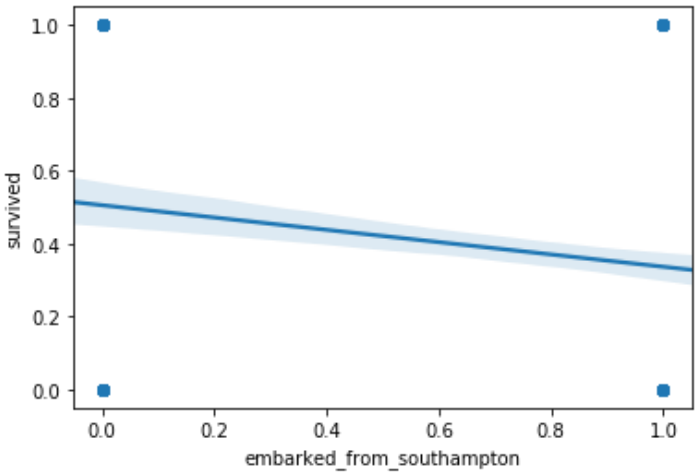


In [24]:

```
sns.regplot(x=df['embarked_from_southampton'],y=df['survived'])
```

Out[24]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f411b00>
```

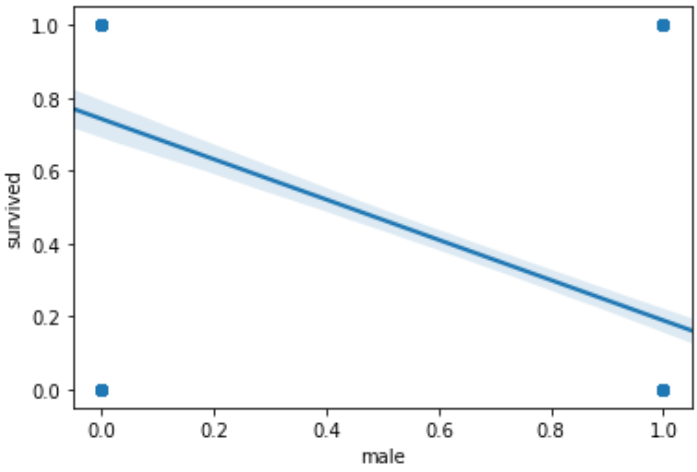


In [25]:

```
sns.regplot(x=df['male'],y=df['survived'])
```

Out[25]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d33f4737b8>
```

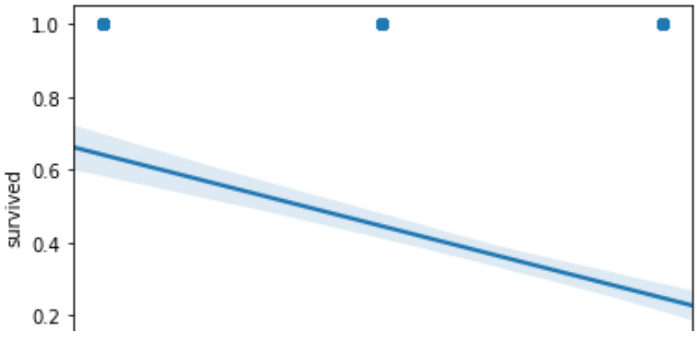


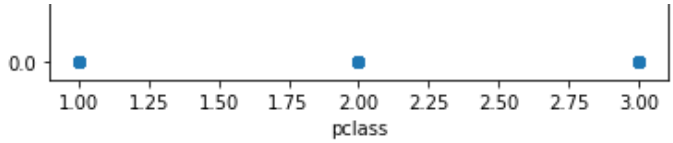
In [202]:

```
sns.regplot(x=df['pclass'],y=df['survived'])
```

Out[202]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d3415eff60>
```





Modelling :

1. Knn Classifier

In [177]:

```
from sklearn.model_selection import train_test_split # Splitting train test
x=df.drop(['survived','age_was_missing'],axis=1)
y=df['survived']
```

In [178]:

```
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=5)
```

In [179]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,stratify=y)
```

In [180]:

```
knn.fit(x_train,y_train)
```

Out[180]:

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

In [203]:

```
knn.score(x_test,y_test) # accuracy
```

Out[203]:

```
0.7798507462686567
```

In [204]:

```
knn.predict(x_test)    # predicting on the test dataset
```

Out[204]:

```
array([1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 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, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0,
       0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
       1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 1, 0, 0], dtype=int64)
```

2. Logistic Regression

In [183]:

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

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30)
```

In [184]:

```
logreg=LogisticRegression()
```

In [185]:

```
logreg.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWa
rning: Default solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence thi
s warning.
  FutureWarning)
```

Out[185]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [205]:

```
logreg.score(x_test,y_test)  # accuracy
```

Out[205]:

```
0.832089552238806
```

In [210]:

```
logreg.predict(x_test)    # predicting on the test dataset
```

Out[210]:

```
array([1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0,
       0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1,
       1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1,
       0, 1, 0, 0], dtype=int64)
```

3. Decision Tree Classifier

In [188]:

```
from sklearn.tree import DecisionTreeClassifier
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [189]:

```
dec=DecisionTreeClassifier()
```

In [190]:

```
dec.fit(x_train,y_train)
```

Out[190]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity decrease=0.0, min impurity split=None,
```

```
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=None, splitter='best')
```

In [207]:

```
dec.score(x_test,y_test)    # accuracy
```

Out[207]:

```
0.9029850746268657
```

In [211]:

```
dec.predict(x_test)        # predicting on the test dataset
```

Out[211]:

```
array([1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0,
       1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
       1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 1, 0, 0], dtype=int64)
```

4. Random Forest Classifier

In [192]:

```
from sklearn.ensemble import RandomForestClassifier
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [193]:

```
ran=RandomForestClassifier()
```

In [194]:

```
ran.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning:
The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[194]:

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                       max_depth=None, max_features='auto', max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=10,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

In [209]:

```
ran.score(x_test,y_test)    # accuracy
```

Out[209]:

```
0.8171641791044776
```

In [212]:

```
ran.predict(x_test)     # predicting on the test dataset
```

Out[212]:

```
array([1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
       1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1,
       0, 1, 0, 0], dtype=int64)
```

In [ ]:

```
```