In [1]:

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

In [21]:

```python
var=pd.read_csv('C://Users/Gopi/Desktop/machine learning/csv files/wbcd.csv')
```

In [11]:

```python
var['diagnosis'].value_counts()
```

Out[11]:

```
B    357
M    212
Name: diagnosis, dtype: int64
```

In [23]:

```python
var.columns
```

Out[23]:

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'points_mean', 'symmetry_mean', 'dimension_mean', 'radius_se',
       'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'points_se', 'symmetry_se',
       'dimension_se', 'radius_worst', 'texture_worst', 'perimeter_worst',
       'area_worst', 'smoothness_worst', 'compactness_worst',
       'concavity_worst', 'points_worst', 'symmetry_worst', 'dimension_worst'],
      dtype='object')
```

In [3]:

```python
var.shape
```

Out[3]:

```
(569, 32)
```

In [4]:

```python
var.head()
```

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	c
0	87139402	B	12.32	12.39	78.85	464.1	0.10280	0.06981	
1	8910251	B	10.60	18.95	69.28	346.4	0.09688	0.11470	
2	905520	B	11.04	16.83	70.92	373.2	0.10770	0.07804	
3	868871	B	11.28	13.39	73.00	384.8	0.11640	0.11360	
4	9012568	B	15.19	13.21	97.65	711.8	0.07963	0.06934	

5 rows × 32 columns

In [5]:

```python
var.isnull()
```

Out[5]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	con
0	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	
564	False	False	False	False	False	False	False	False	
565	False	False	False	False	False	False	False	False	
566	False	False	False	False	False	False	False	False	
567	False	False	False	False	False	False	False	False	
568	False	False	False	False	False	False	False	False	

569 rows × 32 columns

In [6]:

```python
var.dropna()
```

Out[6]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
0	87139402	B	12.32	12.39	78.85	464.1	0.10280	0.06981
1	8910251	B	10.60	18.95	69.28	346.4	0.09688	0.11470
2	905520	B	11.04	16.83	70.92	373.2	0.10770	0.07804
3	868871	B	11.28	13.39	73.00	384.8	0.11640	0.11360
4	9012568	B	15.19	13.21	97.65	711.8	0.07963	0.06934
...	...	...	...	...	...	...	...	..
564	911320502	B	13.17	18.22	84.28	537.3	0.07466	0.05994
565	898677	B	10.26	14.71	66.20	321.6	0.09882	0.09159
566	873885	M	15.28	22.41	98.92	710.6	0.09057	0.10520
567	911201	B	14.53	13.98	93.86	644.2	0.10990	0.09242
568	9012795	M	21.37	15.10	141.30	1386.0	0.10010	0.15150

569 rows × 32 columns

In [24]:

```python
del var['id']
```

In [8]:

```python
var.shape
```

Out[8]:

```
(569, 31)
```

In [9]:

```python
var.head()
```

Out[9]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_m
0	B	12.32	12.39	78.85	464.1	0.10280	0.06981	0.03
1	B	10.60	18.95	69.28	346.4	0.09688	0.11470	0.06
2	B	11.04	16.83	70.92	373.2	0.10770	0.07804	0.03
3	B	11.28	13.39	73.00	384.8	0.11640	0.11360	0.04
4	B	15.19	13.21	97.65	711.8	0.07963	0.06934	0.03

5 rows × 31 columns

In [30]:

```python
X = var.iloc[:, 1:].values
X
```

Out[30]:

```
array([[1.232e+01, 1.239e+01, 7.885e+01, ..., 9.391e-02, 2.827e-01,
        6.771e-02],
       [1.060e+01, 1.895e+01, 6.928e+01, ..., 7.926e-02, 2.940e-01,
        7.587e-02],
       [1.104e+01, 1.683e+01, 7.092e+01, ..., 7.431e-02, 2.998e-01,
        7.881e-02],
       ...,
       [1.528e+01, 2.241e+01, 9.892e+01, ..., 1.226e-01, 3.175e-01,
        9.772e-02],
       [1.453e+01, 1.398e+01, 9.386e+01, ..., 1.069e-01, 2.606e-01,
        7.810e-02],
       [2.137e+01, 1.510e+01, 1.413e+02, ..., 1.966e-01, 2.730e-01,
        8.666e-02]])
```

In [32]:

```python
y = var.iloc[:, 0].values
y
```

Out[32]:

```
array(['B', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B',
       'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'M',
       'B', 'M', 'B', 'B', 'B', 'M', 'M', 'B', 'B', 'B', 'M', 'B', 'M',
       'M', 'M', 'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'M', 'B',
       'B', 'B', 'M', 'B', 'B', 'B', 'M', 'M', 'M', 'M', 'M', 'M', 'M',
       'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'M', 'B',
       'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B',
       'B', 'M', 'B', 'M', 'M', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'M',
       'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'M', 'B', 'M', 'M',
       'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'M', 'M', 'M', 'M',
       'M', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'M', 'B', 'M', 'M', 'M',
       'M', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'M', 'B', 'M', 'B', 'M', 'B', 'M', 'B', 'B', 'M',
       'B', 'M', 'B', 'B', 'M', 'M', 'M', 'B', 'M', 'B', 'B', 'M', 'M',
       'M', 'M', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'M',
       'B', 'M', 'B', 'M', 'M', 'B', 'B', 'B', 'B', 'B', 'M', 'M', 'B',
       'B', 'M', 'B', 'B', 'B', 'B', 'M', 'M', 'B', 'B', 'B', 'B', 'M',
       'B', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B', 'M', 'M', 'B', 'B',
       'B', 'B', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'B', 'M', 'B', 'M', 'M', 'B', 'B', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'M', 'B',
       'M', 'M', 'M', 'B', 'B', 'B', 'M', 'B', 'M', 'M', 'M', 'M', 'B',
       'M', 'M', 'B', 'B', 'M', 'M', 'M', 'M', 'B', 'B', 'B', 'M', 'M',
       'B', 'B', 'M', 'B', 'M', 'M', 'B', 'B', 'B', 'M', 'M', 'B', 'B',
       'M', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B',
       'M', 'B', 'B', 'M', 'M', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B',
       'M', 'B', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'M', 'M', 'B', 'B',
       'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'B', 'B', 'B', 'B', 'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B',
       'M', 'B', 'B', 'M', 'B', 'B', 'B', 'M', 'M', 'B', 'B', 'B', 'M',
       'M', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'M',
       'B', 'B', 'M', 'M', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B', 'B',
       'M', 'M', 'M', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B',
       'M', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'M', 'M', 'B', 'B',
       'B', 'B', 'M', 'B', 'B', 'M', 'M', 'B', 'B', 'M', 'B', 'B', 'M',
       'B', 'B', 'B', 'B', 'B', 'M', 'B', 'M', 'B', 'M', 'B', 'M', 'B',
       'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'M', 'M', 'M', 'M', 'B',
       'B', 'B', 'B', 'M', 'M', 'M', 'M', 'M', 'M', 'B', 'B', 'B', 'B',
       'B', 'M', 'M', 'B', 'M', 'M', 'B', 'M', 'M', 'M', 'M', 'M', 'B',
       'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B',
       'M', 'M', 'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'M', 'M', 'B',
       'B', 'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B',
       'B', 'M', 'B', 'B', 'B', 'B', 'B', 'M', 'B', 'M'], dtype=object)
```

In [28]:

```python
norm=(X-X.min())/(X.max()-X.min())
norm
```

Out[28]:

```
array([[2.89609779e-03, 2.91255289e-03, 1.85354960e-02, ...,
        2.20756935e-05, 6.64551011e-05, 1.59167842e-05],
       [2.49177245e-03, 4.45463094e-03, 1.62858486e-02, ...,
        1.86318759e-05, 6.91114245e-05, 1.78349788e-05],
       [2.59520451e-03, 3.95627645e-03, 1.66713681e-02, ...,
        1.74682652e-05, 7.04748472e-05, 1.85260931e-05],
       ...,
       [3.59191349e-03, 5.26798307e-03, 2.32534086e-02, ...,
        2.88199342e-05, 7.46356370e-05, 2.29713211e-05],
       [3.41560884e-03, 3.28631876e-03, 2.20639398e-02, ...,
        2.51292901e-05, 6.12599906e-05, 1.83591913e-05],
       [5.02350729e-03, 3.54960038e-03, 3.32157969e-02, ...,
        4.62153268e-05, 6.41748942e-05, 2.03714151e-05]])
```

In [48]:

```python
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.20)
```

In [49]:

```python
X_train.shape
```

Out[49]:

```
(455, 30)
```

In [50]:

```python
X_test.shape
```

Out[50]:

```
(114, 30)
```

In [51]:

```python
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train,y_train)
y_pred=knn.predict(X_test)
y_pred
```

Out[51]:

```
array(['B', 'B', 'M', 'M', 'B', 'B', 'M', 'M', 'B', 'M', 'M', 'M', 'B',
       'B', 'M', 'B', 'B', 'B', 'M', 'M', 'M', 'B', 'M', 'B', 'M', 'B',
       'M', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B',
       'M', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'M', 'B',
       'B', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'M', 'M', 'M', 'B', 'B', 'B', 'B', 'B', 'M', 'B', 'M', 'B',
       'B', 'B', 'B', 'M', 'B', 'B', 'M', 'M', 'M', 'B', 'B', 'B', 'M',
       'M', 'B', 'M', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'M', 'M', 'M', 'B', 'B', 'B', 'B', 'B', 'B'], dtype=object)
```

In [59]:

```python
from sklearn.metrics import confusion_matrix,accuracy_score
print(confusion_matrix(y_test,y_pred))
```

```
[[68  2]
 [ 5 39]]
```

In [60]:

```python
Accuracy_Score = accuracy_score(y_test, y_pred)
Accuracy_Score
```

Out[60]:

```
0.9385964912280702
```