In [52]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [53]:
```
from sklearn.datasets import load_breast_cancer
cancer=load_breast_cancer()
dir(cancer)
```

Out[53]:
```
['DESCR',
 'data',
 'data_module',
 'feature_names',
 'filename',
 'frame',
 'target',
 'target_names']
```

In [54]:
```
df=pd.DataFrame(cancer.data,columns=cancer.feature_names)
df.head()
```

Out[54]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	sy
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23.57	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14.91	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22.54	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	

5 rows × 30 columns

In [55]:
```
df['target']=cancer.target
df.head()
```

Out[55]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364

5 rows × 31 columns

In [56]:
```
df.describe()
```

Out[56]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	wor compactne
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	...	569.000000	569.000000	569.000000	569.000000	569.0000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	0.062798	...	25.677223	107.261213	880.583128	0.132369	0.2542
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060	...	6.146258	33.602542	569.356993	0.022832	0.1573
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960	...	12.020000	50.410000	185.200000	0.071170	0.0272
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700	...	21.080000	84.110000	515.300000	0.116600	0.1472
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540	...	25.410000	97.660000	686.500000	0.131300	0.2119
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120	...	29.720000	125.400000	1084.000000	0.146000	0.3391
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440	...	49.540000	251.200000	4254.000000	0.222600	1.0580

8 rows × 31 columns

In [ ]:
```

```

In [57]:
```
x=df.drop(['target'],axis=1)
y=df['target']
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x=sc.fit_transform(x)

from sklearn.model_selection import train_test_split
x_tr,x_te,y_tr,y_te= train_test_split(x,y,test_size=0.2)
```

In [58]:
```
from sklearn.svm import SVC
```

In [59]:
```
model=SVC(kernel='linear')
model.fit(x_tr,y_tr)

model.score(x_te,y_te)
```

Out[59]:
```
0.9736842105263158
```

In [60]:
```
from sklearn.linear_model import LinearRegression
```

In [61]:
```
lr=LinearRegression()
lr.fit(x_tr,y_tr)
lr.score(x_te,y_te)
```

Out[61]:
```
0.7354616770395517
```

In [71]:
```
from sklearn.decomposition import PCA
pca=PCA(0.98)
pca.fit(x)
```

Out[71]:
```
PCA(n_components=0.98)
```

In [72]:
```
x_pca=pca.transform(x)
```

In [73]:
```
x.shape
```

Out[73]:
```
(569, 30)
```

In [74]:
```
x_pca.shape
```

Out[74]:
```
(569, 14)
```

In [75]:
```
pca.explained_variance_ratio_
```

Out[75]:
```
array([0.44272026, 0.18971182, 0.09393163, 0.06602135, 0.05495768,
       0.04024522, 0.02250734, 0.01588724, 0.01389649, 0.01168978,
       0.00979719, 0.00870538, 0.00804525, 0.00523366])
```

In [76]:
```
pca.n_components_
```

Out[76]:
```
14
```

In [77]:
```
x
```

Out[77]:
```
array([[ 1.09706398, -2.07333501,  1.26993369, ...,  2.29607613,
         2.75062224,  1.93701461],
       [ 1.82982061, -0.35363241,  1.68595471, ...,  1.0870843 ,
        -0.24388967,  0.28118999],
       [ 1.57988811,  0.45618695,  1.56650313, ...,  1.95500035,
         1.152255  ,  0.20139121],
       ...,
       [ 0.70228425,  2.0455738 ,  0.67267578, ...,  0.41406869,
        -1.10454895, -0.31840916],
       [ 1.83834103,  2.33645719,  1.98252415, ...,  2.28998549,
         1.91908301,  2.21963528],
       [-1.80840125,  1.22179204, -1.81438851, ..., -1.74506282,
        -0.04813821, -0.75120669]])
```

In [78]:
```
x_pca
```

Out[78]:
```
array([[ 9.19283683,  1.94858307, -1.12316616, ..., -0.85901448,
         0.10338766, -0.6908041 ],
       [ 2.3878018 , -3.76817174, -0.52929269, ...,  0.15792259,
        -0.94352928, -0.65347533],
       [ 5.73389628, -1.0751738 , -0.55174759, ...,  0.12438706,
        -0.4106266 ,  0.01667976],
       ...,
       [ 1.25617928, -1.90229671,  0.56273053, ...,  0.0964729 ,
         0.15741795,  0.28569088],
       [10.37479406,  1.67201011, -1.87702933, ..., -0.69746117,
         1.22519497,  0.21869766],
       [-5.4752433 , -0.67063679,  1.49044308, ..., -0.17949644,
         0.67889748, -1.17072466]])
```

In [ ]:
```

```