

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_val_score
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
sonar=pd.read_csv('sonar.csv')
```

In [3]:

```
sonar
```

Out[3]:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V52	V53	V54	V55	V56	V57
0	0.0200	0.0371	0.0428	0.0207	0.0954	0.0986	0.1539	0.1601	0.3109	0.2111	...	0.0027	0.0065	0.0159	0.0072	0.0167	0.0180
1	0.0453	0.0523	0.0843	0.0689	0.1183	0.2583	0.2156	0.3481	0.3337	0.2872	...	0.0084	0.0089	0.0048	0.0094	0.0191	0.0140
2	0.0262	0.0582	0.1099	0.1083	0.0974	0.2280	0.2431	0.3771	0.5598	0.6194	...	0.0232	0.0166	0.0095	0.0180	0.0244	0.0316
3	0.0100	0.0171	0.0623	0.0205	0.0205	0.0368	0.1098	0.1276	0.0598	0.1264	...	0.0121	0.0036	0.0150	0.0085	0.0073	0.0050
4	0.0762	0.0666	0.0481	0.0394	0.0590	0.0649	0.1209	0.2467	0.3564	0.4459	...	0.0031	0.0054	0.0105	0.0110	0.0015	0.0072
...
203	0.0187	0.0346	0.0168	0.0177	0.0393	0.1630	0.2028	0.1694	0.2328	0.2684	...	0.0116	0.0098	0.0199	0.0033	0.0101	0.0065
204	0.0323	0.0101	0.0298	0.0564	0.0760	0.0958	0.0990	0.1018	0.1030	0.2154	...	0.0061	0.0093	0.0135	0.0063	0.0063	0.0034
205	0.0522	0.0437	0.0180	0.0292	0.0351	0.1171	0.1257	0.1178	0.1258	0.2529	...	0.0160	0.0029	0.0051	0.0062	0.0089	0.0140
206	0.0303	0.0353	0.0490	0.0608	0.0167	0.1354	0.1465	0.1123	0.1945	0.2354	...	0.0086	0.0046	0.0126	0.0036	0.0035	0.0034
207	0.0260	0.0363	0.0136	0.0272	0.0214	0.0338	0.0655	0.1400	0.1843	0.2354	...	0.0146	0.0129	0.0047	0.0039	0.0061	0.0040

208 rows × 61 columns

In [4]:

```
sonar.describe()
```

Out[4]:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...
count	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	...
mean	0.029164	0.038437	0.043832	0.053892	0.075202	0.104570	0.121747	0.134799	0.178003	0.208259	...
std	0.022991	0.032960	0.038428	0.046528	0.055552	0.059105	0.061788	0.085152	0.118387	0.134416	...
min	0.001500	0.000600	0.001500	0.005800	0.006700	0.010200	0.003300	0.005500	0.007500	0.011300	...
25%	0.013350	0.016450	0.018950	0.024375	0.038050	0.067025	0.080900	0.080425	0.097025	0.111275	...
50%	0.022800	0.030800	0.034300	0.044050	0.062500	0.092150	0.106950	0.112100	0.152250	0.182400	...
75%	0.035550	0.047950	0.057950	0.064500	0.100275	0.134125	0.154000	0.169600	0.233425	0.268700	...

max	0.137100 V1	0.233900 V2	0.305900 V3	0.426400 V4	0.401000 V5	0.382300 V6	0.372900 V7	0.459000 V8	0.682800 V9	0.710600 V10	...
-----	----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------	-----------------	-----

8 rows × 61 columns



In [5]:

```
sonar.columns
```

Out[5]:

```
Index(['V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10', 'V11',  
      'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20', 'V21',  
      'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'V29', 'V30', 'V31',  
      'V32', 'V33', 'V34', 'V35', 'V36', 'V37', 'V38', 'V39', 'V40', 'V41',  
      'V42', 'V43', 'V44', 'V45', 'V46', 'V47', 'V48', 'V49', 'V50', 'V51',  
      'V52', 'V53', 'V54', 'V55', 'V56', 'V57', 'V58', 'V59', 'V60', 'Class'],  
      dtype='object')
```

In [6]:

```
sonar.skew()
```

Out[6]:

```
V1      2.131088  
V2      2.155644  
V3      2.652518  
V4      3.401697  
V5      2.018141  
...  
V57     1.653090  
V58     2.098330  
V59     1.737506  
V60     2.775754  
Class    0.135903  
Length: 61, dtype: float64
```

In [7]:

```
for col in sonar.columns:  
    if sonar.skew().loc[col]>0.55:  
        sonar[col]=np.log1p(sonar[col])
```

In [8]:

```
sonar.skew()
```

Out[8]:

```
V1      2.036001  
V2      1.969917  
V3      2.344713  
V4      2.818320  
V5      1.698684  
...  
V57     1.629182  
V58     2.058207  
V59     1.713349  
V60     2.711412  
Class    0.135903  
Length: 61, dtype: float64
```

In [9]:

```
sonar.Class.unique()
```

Out[9]:

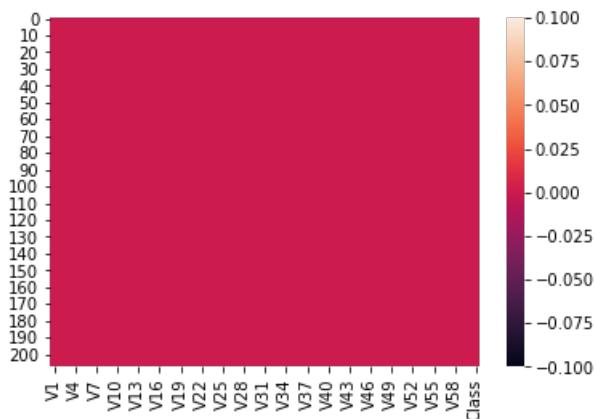
```
array([1, 0], dtype=int64)
```

In [10]:

```
sns.heatmap(sonar.isnull())
```

Out[10]:

<matplotlib.axes._subplots.AxesSubplot at 0x1c0511ae520>



In [11]:

```
from scipy.stats import zscore
z_score=abs(zscore(sonar))
print(sonar.shape)
sonar_df=sonar.loc[(z_score<3).all(axis=1)]
print(sonar_df.shape)
```

(208, 61)

(173, 61)

In [12]:

```
sonar_df
```

Out[12]:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V52	V53	
0	0.019803	0.036428	0.041909	0.020489	0.091120	0.094037	0.143148	0.148506	0.270714	0.191529	...	0.002696	0.006479	0.015
1	0.044304	0.050978	0.080935	0.066630	0.111810	0.229762	0.195238	0.298696	0.287957	0.252469	...	0.008365	0.008861	0.004
3	0.009950	0.016955	0.060436	0.020293	0.020293	0.036139	0.104180	0.120091	0.058080	0.119027	...	0.012027	0.003594	0.014
4	0.073436	0.064476	0.046979	0.038644	0.057325	0.062881	0.114132	0.220500	0.304834	0.368732	...	0.003095	0.005385	0.010
5	0.028199	0.044304	0.027323	0.017250	0.037681	0.094401	0.113418	0.168307	0.191033	0.265360	...	0.004490	0.001399	0.003
...
203	0.018527	0.034015	0.016660	0.017545	0.038547	0.151003	0.184652	0.156491	0.209288	0.237756	...	0.011533	0.009752	0.019
204	0.031789	0.010049	0.029365	0.054867	0.073250	0.091485	0.094401	0.096945	0.098034	0.195073	...	0.006081	0.009257	0.013
205	0.050883	0.042772	0.017840	0.028782	0.034498	0.110736	0.118405	0.111362	0.118494	0.225461	...	0.015873	0.002896	0.005
206	0.029850	0.034691	0.047837	0.059023	0.016562	0.126985	0.136714	0.106430	0.177728	0.211395	...	0.008563	0.004589	0.012
207	0.025668	0.035657	0.013508	0.026837	0.021174	0.033241	0.063444	0.131028	0.169152	0.211395	...	0.014494	0.012818	0.004

173 rows × 61 columns

In [13]:

```
sonar_df=pd.DataFrame(data=sonar_df)
sonar_df
```

Out[13]:

V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V52 V53

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V52	V53	
0	0.019803	0.036428	0.041909	0.020489	0.091120	0.094037	0.143148	0.148506	0.270714	0.191529	...	0.002696	0.006479	0.015
1	0.044304	0.050978	0.080935	0.066630	0.111810	0.229762	0.195238	0.298696	0.287957	0.252469	...	0.008365	0.008861	0.004
3	0.009950	0.016955	0.060436	0.020293	0.020293	0.036139	0.104180	0.120091	0.058080	0.119027	...	0.012027	0.003594	0.014
4	0.073436	0.064476	0.046979	0.038644	0.057325	0.062881	0.114132	0.220500	0.304834	0.368732	...	0.003095	0.005385	0.010
5	0.028199	0.044304	0.027323	0.017250	0.037681	0.094401	0.113418	0.168307	0.191033	0.265360	...	0.004490	0.001399	0.003
...	
203	0.018527	0.034015	0.016660	0.017545	0.038547	0.151003	0.184652	0.156491	0.209288	0.237756	...	0.011533	0.009752	0.019
204	0.031789	0.010049	0.029365	0.054867	0.073250	0.091485	0.094401	0.096945	0.098034	0.195073	...	0.006081	0.009257	0.013
205	0.050883	0.042772	0.017840	0.028782	0.034498	0.110736	0.118405	0.111362	0.118494	0.225461	...	0.015873	0.002896	0.005
206	0.029850	0.034691	0.047837	0.059023	0.016562	0.126985	0.136714	0.106430	0.177728	0.211395	...	0.008563	0.004589	0.012
207	0.025668	0.035657	0.013508	0.026837	0.021174	0.033241	0.063444	0.131028	0.169152	0.211395	...	0.014494	0.012818	0.004

173 rows × 61 columns



In [14]:

```
x=sonar_df.iloc[:,0:-1]
```

In [15]:

```
x
```

Out[15]:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V51	V52	
0	0.019803	0.036428	0.041909	0.020489	0.091120	0.094037	0.143148	0.148506	0.270714	0.191529	...	0.022935	0.002696	0.006
1	0.044304	0.050978	0.080935	0.066630	0.111810	0.229762	0.195238	0.298696	0.287957	0.252469	...	0.012423	0.008365	0.008
3	0.009950	0.016955	0.060436	0.020293	0.020293	0.036139	0.104180	0.120091	0.058080	0.119027	...	0.023814	0.012027	0.003
4	0.073436	0.064476	0.046979	0.038644	0.057325	0.062881	0.114132	0.220500	0.304834	0.368732	...	0.015480	0.003095	0.005
5	0.028199	0.044304	0.027323	0.017250	0.037681	0.094401	0.113418	0.168307	0.191033	0.265360	...	0.010346	0.004490	0.001
...
203	0.018527	0.034015	0.016660	0.017545	0.038547	0.151003	0.184652	0.156491	0.209288	0.237756	...	0.020097	0.011533	0.009
204	0.031789	0.010049	0.029365	0.054867	0.073250	0.091485	0.094401	0.096945	0.098034	0.195073	...	0.005087	0.006081	0.009
205	0.050883	0.042772	0.017840	0.028782	0.034498	0.110736	0.118405	0.111362	0.118494	0.225461	...	0.015381	0.015873	0.002
206	0.029850	0.034691	0.047837	0.059023	0.016562	0.126985	0.136714	0.106430	0.177728	0.211395	...	0.004191	0.008563	0.004
207	0.025668	0.035657	0.013508	0.026837	0.021174	0.033241	0.063444	0.131028	0.169152	0.211395	...	0.017938	0.014494	0.012

173 rows × 60 columns



In [16]:

```
x.shape
```

Out[16]:

```
(173, 60)
```

In [17]:

```
pca=PCA(n_components=12)
```

In [18]:

```
x=pca.fit_transform(x)
```

In [19]:

```
x.shape
```

```
Out[19]:  
(173, 12)
```

```
In [20]:
```

```
y=sonar_df.iloc[:,-1]
```

```
In [21]:
```

```
y
```

```
Out[21]:
```

```
0      1  
1      1  
3      1  
4      1  
5      1  
      ..  
203    0  
204    0  
205    0  
206    0  
207    0  
Name: Class, Length: 173, dtype: int64
```

```
In [22]:
```

```
y.shape
```

```
Out[22]:
```

```
(173,)
```

```
In [23]:
```

```
sonar.corr()
```

```
Out[23]:
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V52	V53	
V1	1.000000	0.735222	0.570685	0.485743	0.336469	0.244380	0.260978	0.351766	0.348769	0.312130	...	0.353387	0.312563	0.3
V2	0.735222	1.000000	0.773040	0.590833	0.404392	0.334846	0.278614	0.337122	0.318649	0.269347	...	0.434345	0.348536	0.3
V3	0.570685	0.773040	1.000000	0.765006	0.524464	0.351185	0.192678	0.246778	0.263476	0.226283	...	0.396473	0.335055	0.3
V4	0.485743	0.590833	0.765006	1.000000	0.708444	0.362016	0.252227	0.260257	0.252504	0.242135	...	0.379065	0.370981	0.3
V5	0.336469	0.404392	0.524464	0.708444	1.000000	0.605497	0.340052	0.212995	0.188492	0.194106	...	0.267553	0.315872	0.2
...
V57	0.316519	0.286026	0.388956	0.351808	0.216137	0.164213	0.183444	0.259359	0.179931	0.128099	...	0.190659	0.306836	0.3
V58	0.371517	0.358558	0.342265	0.357230	0.236755	0.208964	0.241244	0.283423	0.223121	0.203778	...	0.309571	0.369978	0.4
V59	0.358920	0.354793	0.431001	0.430703	0.288460	0.224321	0.179633	0.186581	0.082903	0.052438	...	0.297710	0.346919	0.4
V60	0.348455	0.358427	0.375295	0.400941	0.247061	0.183374	0.220841	0.142316	0.082875	0.093219	...	0.196169	0.282185	0.2
Class	0.272710	0.232735	0.194547	0.257889	0.227755	0.141348	0.120424	0.196120	0.337322	0.354554	...	0.289856	0.141687	0.1

61 rows × 61 columns

In [24]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.22,random_state=45)
```

In [25]:

```
lr=LogisticRegression()
```

In [26]:

```
lr.fit(x_train,y_train)
lr.score(x_train,y_train)
pred=lr.predict(x_test)
print(accuracy_score(y_test,pred))
print(confusion_matrix(y_test,pred))
print(classification_report(y_test,pred))
```

0.8205128205128205

```
[[17  5]
```

```
 [ 2 15]]
```

	precision	recall	f1-score	support
0	0.89	0.77	0.83	22
1	0.75	0.88	0.81	17
accuracy			0.82	39
macro avg	0.82	0.83	0.82	39
weighted avg	0.83	0.82	0.82	39

In [27]:

```
svc=SVC(kernel='rbf')
svc.fit(x_train,y_train)
svc.score(x_train,y_train)
predsvc=svc.predict(x_test)
print(accuracy_score(y_test,predsvc))
print(confusion_matrix(y_test,predsvc))
print(classification_report(y_test,predsvc))
```

0.8461538461538461

```
[[16  6]
```

```
 [ 0 17]]
```

	precision	recall	f1-score	support
0	1.00	0.73	0.84	22
1	0.74	1.00	0.85	17
accuracy			0.85	39
macro avg	0.87	0.86	0.85	39
weighted avg	0.89	0.85	0.85	39

In [28]:

```
dtc=DecisionTreeClassifier()
dtc.fit(x_train,y_train)
dtc.score(x_train,y_train)
preddtc=dtc.predict(x_test)
print(accuracy_score(y_test,preddtc))
print(confusion_matrix(y_test,preddtc))
print(classification_report(y_test,preddtc))
```

0.7948717948717948

```
[[18  4]
```

```
 [ 4 13]]
```

	precision	recall	f1-score	support
0	0.82	0.82	0.82	22

1	0.76	0.76	0.76	17
accuracy			0.79	39
macro avg	0.79	0.79	0.79	39
weighted avg	0.79	0.79	0.79	39

In [29]:

```
gnb=GaussianNB()
gnb.fit(x_train,y_train)
gnb.score(x_train,y_train)
predgnb=gnb.predict(x_test)
print(accuracy_score(y_test,predgnb))
print(confusion_matrix(y_test,predgnb))
print(classification_report(y_test,predgnb))
```

```
0.8717948717948718
[[19  3]
 [ 2 15]]
```

	precision	recall	f1-score	support
0	0.90	0.86	0.88	22
1	0.83	0.88	0.86	17
accuracy			0.87	39
macro avg	0.87	0.87	0.87	39
weighted avg	0.87	0.87	0.87	39

In [30]:

```
import joblib
joblib.dump(gnb,'sonar.pkl')
```

Out[30]:

```
['sonar.pkl']
```

In []: