In [3]:

```
import pandas as pd
import numpy as np
```

In [5]:

```
df = pd.read_csv("D:\\DATA SCIENCE (ASY)\\Datasets\\sonar.all-data.csv",header=None)
```

In [6]:

```
df
```

Out[6]:

	0	1	2	3	4	5	6	7	8	9	...	51	52	53	54	55	56	57	58	59	60
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	0.0084	0.0090	0.0032	R
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	0.0049	0.0052	0.0044	R
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	0.0164	0.0095	0.0078	R
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	0.0044	0.0040	0.0117	R
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	0.0048	0.0107	0.0094	R
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
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	0.0115	0.0193	0.0157	M
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	0.0032	0.0062	0.0067	M
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	0.0138	0.0077	0.0031	M
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	0.0079	0.0036	0.0048	M
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	0.0036	0.0061	0.0115	M

208 rows × 61 columns

In [7]:

```
df.head
```

Out[7]:

```
<bound method NDFrame.head of          0       1       2       3       4       5       6       7       8   \
0    0.0200  0.0371  0.0428  0.0207  0.0954  0.0986  0.1539  0.1601  0.3109
1    0.0453  0.0523  0.0843  0.0689  0.1183  0.2583  0.2156  0.3481  0.3337
2    0.0262  0.0582  0.1099  0.1083  0.0974  0.2280  0.2431  0.3771  0.5598
3    0.0100  0.0171  0.0623  0.0205  0.0205  0.0368  0.1098  0.1276  0.0598
4    0.0762  0.0666  0.0481  0.0394  0.0590  0.0649  0.1209  0.2467  0.3564
..      ...     ...     ...     ...     ...     ...     ...     ...     ...
203  0.0187  0.0346  0.0168  0.0177  0.0393  0.1630  0.2028  0.1694  0.2328
204  0.0323  0.0101  0.0298  0.0564  0.0760  0.0958  0.0990  0.1018  0.1030
205  0.0522  0.0437  0.0180  0.0292  0.0351  0.1171  0.1257  0.1178  0.1258
206  0.0303  0.0353  0.0490  0.0608  0.0167  0.1354  0.1465  0.1123  0.1945
207  0.0260  0.0363  0.0136  0.0272  0.0214  0.0338  0.0655  0.1400  0.1843

         9   ...      51      52      53      54      55      56      57  \
0    0.2111  ...  0.0027  0.0065  0.0159  0.0072  0.0167  0.0180  0.0084
1    0.2872  ...  0.0084  0.0089  0.0048  0.0094  0.0191  0.0140  0.0049
2    0.6194  ...  0.0232  0.0166  0.0095  0.0180  0.0244  0.0316  0.0164
3    0.1264  ...  0.0121  0.0036  0.0150  0.0085  0.0073  0.0050  0.0044
4    0.4459  ...  0.0031  0.0054  0.0105  0.0110  0.0015  0.0072  0.0048
..      ...  ...     ...     ...     ...     ...     ...     ...     ...
203  0.2684  ...  0.0116  0.0098  0.0199  0.0033  0.0101  0.0065  0.0115
204  0.2154  ...  0.0061  0.0093  0.0135  0.0063  0.0063  0.0034  0.0032
205  0.2529  ...  0.0160  0.0029  0.0051  0.0062  0.0089  0.0140  0.0138
206  0.2354  ...  0.0086  0.0046  0.0126  0.0036  0.0035  0.0034  0.0079
207  0.2354  ...  0.0146  0.0129  0.0047  0.0039  0.0061  0.0040  0.0036

         58      59  60
0    0.0090  0.0032   R
1    0.0052  0.0044   R
2    0.0095  0.0078   R
3    0.0040  0.0117   R
4    0.0107  0.0094   R
..      ...     ...  ..
203  0.0193  0.0157   M
204  0.0062  0.0067   M
205  0.0077  0.0031   M
206  0.0036  0.0048   M
207  0.0061  0.0115   M

[208 rows x 61 columns]>
```

In [8]:

```
X = df.drop(60, axis=1)
```

In [9]:

```
y = df[60]
```

In [10]:

```
from sklearn.model_selection import train_test_split
```

In [11]:

```
Xtrain,Xtest,ytrain,ytest = train_test_split(X,y,test_size =0.20)
```

In [12]:

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

In [13]:

```
lr = LogisticRegression()
```

In [14]:

```
lr.fit(Xtrain,ytrain)
```

Out[14]:

```
LogisticRegression()
```

In [15]:

```
lr.score(Xtest,ytest)
```

Out[15]:

```
0.7380952380952381
```

In [16]:

```
from sklearn.model_selection import KFold,cross_val_score
```

In [17]:

```
kf = KFold(n_splits=10)
```

In [18]:

```
lr1 = LogisticRegression()
```

In [19]:

```
import warnings
warnings.filterwarnings("ignore")
```

In [20]:

```
lrscore = cross_val_score(lr1,X,y,cv =kf)
```

In [21]:

```
lrscore
```

Out[21]:

```
array([0.38095238, 0.61904762, 0.52380952, 0.57142857, 0.42857143,
       0.38095238, 0.76190476, 0.47619048, 0.75      , 0.35      ])
```

In [22]:

```
lrscore.mean()
```

Out[22]:

```
0.5242857142857142
```

In [23]:

```
from sklearn.preprocessing import StandardScaler
```

In [24]:

```
sc = StandardScaler()
```

In [25]:

```
sc.fit(X)
```

Out[25]:

```
StandardScaler()
```

In [26]:

```
ary1 = sc.transform(X)
```

In [27]:

```
lreg = LogisticRegression()
```

In [28]:

```
lregscore = cross_val_score(lreg,ary1,y,cv =kf)
```

In [29]:

```
lregscore.mean()
```

Out[29]:

```
0.5685714285714286
```

In [30]:

```
from sklearn.tree import DecisionTreeClassifier
```

In [31]:

```
dtr = DecisionTreeClassifier()
```

In [32]:

```
dtr_score = cross_val_score(dtr,ary1,y,cv =kf)
```

In [33]:

```
dtr_score.mean()
```

Out[33]:

```
0.5554761904761905
```

In [34]:

```
from sklearn.svm import SVC
```

In [35]:

```
svc = SVC()
```

In [36]:

```
svc
```

Out[36]:

```
SVC()
```

In [37]:

```
svc_score = cross_val_score(svc,ary1,y,cv =kf)
```

In [38]:

```
svc_score.mean()
```

Out[38]:

```
0.4961904761904762
```

In [39]:

```
from sklearn.model_selection import train_test_split
```

In [41]:

```
Xtrain,Xtest,ytrain,ytest = train_test_split(X,y,test_size =0.2,random_state =1)
```

In [42]:

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

In [43]:

```
lr2 = LogisticRegression()
```

In [44]:

```
lr2.fit(Xtrain,ytrain)
```

Out[44]:

```
LogisticRegression()
```

In [45]:

```
lr2.score(Xtest,ytest)
```

Out[45]:

```
0.8095238095238095
```

In [ ]: