In [7]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [8]:
```
df=pd.read_csv(r"c:\Users\Mastan Reddy\Downloads\archive (4).zip")
df
```

Out[8]:

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.03760	...	-0.511
0	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	...	-0.265
1	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	...	-0.402
2	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	...	0.906
3	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	...	-0.651
4	1	0	0.02337	-0.00592	-0.09924	-0.11949	-0.00763	-0.11824	0.14706	0.06637	...	-0.015
...	...	...	...	...	...	...	...	...	...	...	...	
345	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	...	-0.042
346	1	0	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.013
347	1	0	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	...	0.031
348	1	0	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	-0.020
349	1	0	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	...	-0.151

350 rows × 35 columns

In [9]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [10]:
```
print('This DataFreme has %d Rows and %d columns'%(df.shape))
```

```
This DataFreme has 350 Rows and 35 columns
```

In [11]:
```
df.head()
```

Out[11]:

	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.1	0.03760	0.85243.1	-0
0	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	0.50874	-0
1	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	0.73082	0
2	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	0.00000	0
3	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	0.52798	-0
4	1	0	0.02337	-0.00592	-0.09924	-0.11949	-0.00763	-0.11824	0.14706	0.06637	0.03786	-0

In [12]:
```
features_matrix=df.iloc[:,0:34]
```

In [13]:
```
target_vector=df.iloc[:,-1]
```

In [14]:
```
print('The Features Matrix Has %d Rows And %d Column(s)'%(features_matrix.shap
print('The Target Matrix Has %d Rows and %d columns(s)'%(np.array(target_vecto
```

```
The Features Matrix Has 350 Rows And 34 Column(s)
The Target Matrix Has 350 Rows and 1 columns(s)
```

In [15]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [16]:
```
algorithm=LogisticRegression(max_iter=1000)
```

In [17]:
```
logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_ve
```

In [18]:
```
observation=[[1,0,0.99539,-0.5889,0.8524299999999999,0.02306,0.8339799999999,
             0.59755,-0.44945,0.60536,-0.38223,0.8435600000000001,-0.38542,0.
             0.56811,-0.51171,0.41078000000000003,-0.46168000000000003,0.21256
```

In [19]:
```
predictions=logistic_Regression_Model.predict(observation)
print('The Model predicted the observation to belog to class %s'%(predictions)
```

```
The Model predicted the observation to belog to class ['g']
```

In [20]:
```
print('The algorithm was trained to predict one of the two classes:%s'%(algori
```

```
The algorithm was trained to predict one of the two classes:['b' 'g']
```

In [21]:

```
print("""The model says the probability of the obserbvation we passedbelonging
      %(algorithm.predict_proba(observation)[0][0]))
print()
print("""The model says the probability of the observation we passed belonging
     %(algorithm.predict_proba(observation)[observation[0][1]]))
```

```
The model says the probability of the obserbvation we passedbelonging to clas
s['b']is 0.0

The model says the probability of the observation we passed belonging to clas
s['g']is [0. 1.]
```

In [ ]: