In [1]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing  import StandardScaler
```

In [2]:
```
df=pd.read_csv(r"C:\Users\LENOVO\Downloads\ionosphere (1).csv")
df
```

Out[2]:

| | column_a | column_b | column_c | column_d | column_e | column_f | column_g | column_h | colu |
|---|---|---|---|---|---|---|---|---|---|
| **0** | True | False | 0.99539 | -0.05889 | 0.85243 | 0.02306 | 0.83398 | -0.37708 | 1.( |
| **1** | True | False | 1.00000 | -0.18829 | 0.93035 | -0.36156 | -0.10868 | -0.93597 | 1.( |
| **2** | True | False | 1.00000 | -0.03365 | 1.00000 | 0.00485 | 1.00000 | -0.12062 | 0.8 |
| **3** | True | False | 1.00000 | -0.45161 | 1.00000 | 1.00000 | 0.71216 | -1.00000 | 0.( |
| **4** | True | False | 1.00000 | -0.02401 | 0.94140 | 0.06531 | 0.92106 | -0.23255 | 0.7 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **346** | True | False | 0.83508 | 0.08298 | 0.73739 | -0.14706 | 0.84349 | -0.05567 | 0.5 |
| **347** | True | False | 0.95113 | 0.00419 | 0.95183 | -0.02723 | 0.93438 | -0.01920 | 0.5 |
| **348** | True | False | 0.94701 | -0.00034 | 0.93207 | -0.03227 | 0.95177 | -0.03431 | 0.5 |
| **349** | True | False | 0.90608 | -0.01657 | 0.98122 | -0.01989 | 0.95691 | -0.03646 | 0.8 |
| **350** | True | False | 0.84710 | 0.13533 | 0.73638 | -0.06151 | 0.87873 | 0.08260 | 0.8 |

351 rows × 35 columns

In [3]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [4]:
```
print('This Data Frame has %d Rows and %d Columns'%(df.shape))
```

```
This Data Frame has 351 Rows and 35 Columns
```

In [5]:
```
df.head()
```

Out[5]:

| | column_a | column_b | column_c | column_d | column_e | column_f | column_g | column_h | colum |
|---|---|---|---|---|---|---|---|---|---|
| **0** | True | False | 0.99539 | -0.05889 | 0.85243 | 0.02306 | 0.83398 | -0.37708 | 1.00( |
| **1** | True | False | 1.00000 | -0.18829 | 0.93035 | -0.36156 | -0.10868 | -0.93597 | 1.00( |
| **2** | True | False | 1.00000 | -0.03365 | 1.00000 | 0.00485 | 1.00000 | -0.12062 | 0.88 |
| **3** | True | False | 1.00000 | -0.45161 | 1.00000 | 1.00000 | 0.71216 | -1.00000 | 0.00( |
| **4** | True | False | 1.00000 | -0.02401 | 0.94140 | 0.06531 | 0.92106 | -0.23255 | 0.77 |

In [6]:
```
features_matrix=df.iloc[:,0:34]
target_vector=df.iloc[:,-1]
```

In [7]:
```
print('The feature matrix has %d rows and %d column(s)'%(features_matrix.shape
```

```
The feature matrix has 351 rows and 34 column(s)
```

In [8]:
```
print('The target matrix has %d rows and %d column(S)'%(np.array(target_vector
```

```
The target matrix has 351 rows and 1 column(S)
```

In [9]:
```
features_matrix_Standardized=StandardScaler().fit_transform(features_matrix)
```

In [10]:
```
algorithm = LogisticRegression(penalty=None,dual=False,tol=1e-1,C= 1.0,fit_int
```

In [11]:
```
Logistic_Regression_Model=algorithm.fit(features_matrix_Standardized,target_ve
```

In [27]:
```
observation=[[1,0,0.99539,-0.05889,0.854299999999999,0.02306,0.833979999999999
             -0.38223,0.84356000000000001,-0.36946,-0.4737,0.56811,-0.51171,0
```

In [28]:
```
print('The algoritham was trained to predict one of the two classes:%s'%(algor
```

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

In [29]:
```
print(" " "The Model says the probability of the observation we passed belongi
print()
```

```
 The Model says the probability of the observation we passed belonging to cla
ss['b'] Is 3.356663211084854e-05
```

In [30]:
```
print(" " "The Model says the probability of the observation we passed belongi
```

```
 The Model says the probability of the observation we passed belonging to cla
ss['g'] Is 0.9999664333678892
```

In [ ]:
```
conclusion: The data set we have taken is suitable for this model.The Model sa
Is 0.9999664333678892
```