# IONOSPHERE_USING_LOGISTICREGRESSION

In [15]:

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [16]:

```
df=pd.read_csv(r"C:\Users\prajapath Arjun\Downloads\ionosphere.csv")
df
```

Out[16]:

| | column_a | column_b | column_c | column_d | column_e | column_f | column_g | column_h | column_i | column_j | column_k | column_l | column_m | column_n | column_o |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | True | False | 0.99539 | -0.05889 | 0.85243 | 0.02306 | 0.83398 | -0.37708 | 1.00000 | 0.03760 | 0.85243 | -0.17755 | 0.59755 | -0.44945 | 0.6053 |
| 1 | True | False | 1.00000 | -0.18829 | 0.93035 | -0.36156 | -0.10868 | -0.93597 | 1.00000 | -0.04549 | 0.50874 | -0.67743 | 0.34432 | -0.69707 | -0.5168 |
| 2 | True | False | 1.00000 | -0.03365 | 1.00000 | 0.00485 | 1.00000 | -0.12062 | 0.88965 | 0.01198 | 0.73082 | 0.05346 | 0.85443 | 0.00827 | 0.5459 |
| 3 | True | False | 1.00000 | -0.45161 | 1.00000 | 1.00000 | 0.71216 | -1.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | -1.0000 |
| 4 | True | False | 1.00000 | -0.02401 | 0.94140 | 0.06531 | 0.92106 | -0.23255 | 0.77152 | -0.16399 | 0.52798 | -0.20275 | 0.56409 | -0.00712 | 0.3439 |
| 5 | True | False | 0.02337 | -0.00592 | -0.09924 | -0.11949 | -0.00763 | -0.11824 | 0.14706 | 0.06637 | 0.03786 | -0.06302 | 0.00000 | 0.00000 | -0.0457 |
| 6 | True | False | 0.97588 | -0.10602 | 0.94601 | -0.20800 | 0.92806 | -0.28350 | 0.85996 | -0.27342 | 0.79766 | -0.47929 | 0.78225 | -0.50764 | 0.7462 |
| 7 | False | False | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 1.00000 | -1.00000 | 0.00000 | 0.00000 | -1.00000 | -1.00000 | 0.00000 | 0.00000 | 0.0000 |
| 8 | True | False | 0.96355 | -0.07198 | 1.00000 | -0.14333 | 1.00000 | -0.21313 | 1.00000 | -0.36174 | 0.92570 | -0.43569 | 0.94510 | -0.40668 | 0.9039 |

In [17]:

```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [18]:

```
print('This DataFrame has %d rows and %d columns'%(df.shape))
```

```
This DataFrame has 351 rows and 35 columns
```

In [19]:

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

In [20]:

```
print('The Feature Matrix has %d Rows and %d columns(s)'%(features_matrix.shape))
print('The Target Matrix has %d Rows and %d columns(s)'%(np.array(target_vector).reshape(-1,1).shape))
```

```
The Feature Matrix has 351 Rows and 34 columns(s)
The Target Matrix has 351 Rows and 1 columns(s)
```

In [21]:

```
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [22]:

```
,class_weight=None,random_state=None,solver='lbfgs',max_iter=100,multi_class='auto',verbose=0,warm_start=False,n_jobs=None,l1_ratio=None)
```

In [23]:

```
Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_vector)
```

In [24]:

```
observation=[[1,0,0.99539,-0.05889,0.854299999999999,0.02306,
              0.8339799999999999,-0.37708,1.0,0.0376,
              0.8524299999999999,-0.17788,0.59755,-0.44945,0.60536,
              -0.38223,0.8435600000000001,-0.38542,0.58212,-0.32192,
              0.56971,-0.29674,0.36946,-0.47357,0.56811,-0.51171,
              0.41078000000000003,-0.46168000000000003,0.21266,-0.3409,
              0.42267,-0.54487,0.18641,-0.453]]
```

In [25]:

```
predictions=Logistic_Regression_Model.predict(observation)
print('The model predicted the observation to belong to class %s'%(predictions))
print('The algoritham was trained to predict one of the two classes:%s'%(algorithm.classes_))
```

```
The model predicted the observation to belong to class ['g']
The algoritham was trained to predict one of the two classes:['b' 'g']
```

In [27]:

```
e Model says the probability of the observation we passed belonging to class['b'] Is %s" ""%(algorithm.predict_proba(observation)[0][0]))
```

```
The Model says the probability of the observation we passed belonging to class['b'] Is 0.007759086630364176
```

In [28]:

```
Model says the probability of the observation we passed belonging to class['g'] Is %s" " "%(algorithm.predict_proba(observation)[0][0]))
```

```
The Model says the probability of the observation we passed belonging to class['g'] Is 0.007759086630364176
```

In [ ]: