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\sudheer\Downloads\ionosphere.csv")
df
```

Out[2]:

	column_a	column_b	column_c	column_d	column_e	column_f	column_g	column_h	col
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
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
...	...	...	...	...	...	...	...	...	
346	True	False	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0
347	True	False	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0
348	True	False	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0
349	True	False	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0
350	True	False	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0

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 DataFrame has %d Rows and %d columns'%(df.shape))
```

```
This DataFrame 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	colun
0	True	False	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.0(
1	True	False	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.0(
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	0.0(
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]
```

In [7]:
```
target_vector=df.iloc[:,-1]
```

In [8]:
```
print('The Features Matric Has %d Rows And %d column(s)'%(features_matrix.shap
print('The Target Matrix Has %d Rows and %d columns(s)'%(np.array(target_vectc
```

```
The Features Matric Has 351 Rows And 34 column(s)
The Target Matrix Has 351 Rows and 1 columns(s)
```

In [16]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [24]:
```
algorithm=LogisticRegression(penalty=None,dual=False,tol=1e-4,C=1.0,fit_interc
                             random_state=None,solver='lbfgs',max_iter=1000,mul
                              n_jobs=None,l1_ratio=None)
```

In [26]:
```
logistic_Regression_Model = algorithm.fit(features_matrix_standardized,target_
```

In [27]:
```
observation=[[1,0,0.99539,-0.5889,0.8524299999999999,0.02306,0.83397999999999,
             0.59755,-0.44945,0.60536,-0.38223,0.84356000000000001,-0.38542,0.
             0.56811,-0.51171,0.41078000000000003,-0.46168000000000003,0.21256
```

In [28]:
```
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 [29]:
```
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 [30]:
```
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 [ ]:
```

```