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\Gowthami\Downloads\ionosphere_data.csv")
df
```

Out[2]:

	column_a	column_b	column_c	column_d	column_e	column_f	column_g	column_h	column_i	column_j	...	column_z	column_aa	column_ab	c
0	True	False	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.00000	0.03760	...	-0.51171	0.41078	-0.46168	
1	True	False	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	...	-0.26569	-0.20468	-0.18401	
2	True	False	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	...	-0.40220	0.58984	-0.22145	
3	True	False	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	...	0.90695	0.51613	1.00000	
4	True	False	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	...	-0.65158	0.13290	-0.53206	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
346	True	False	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	...	-0.04202	0.83479	0.00123	
347	True	False	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.01361	0.93522	0.04925	
348	True	False	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	...	0.03193	0.92489	0.02542	
349	True	False	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	-0.02099	0.89147	-0.07760	
350	True	False	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	...	-0.15114	0.81147	-0.04822	

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 [6]:

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

```
The DataFrame has 351 Rows and 35 columns
```

In [7]:

```
df.head()
```

Out[7]:

	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
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	-0.4494
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	-0.6970
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	0.0082
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	0.0000
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	-0.0071

In [8]:

```
df.tail()
```

Out[8]:

	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	columr
346	True	False	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	0.89391	0.13130	0.81197	0.067
347	True	False	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	0.96510	0.03281	0.94171	0.07
348	True	False	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	0.94124	0.01766	0.92595	0.046
349	True	False	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	0.89724	-0.03315	0.89061	-0.01
350	True	False	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	0.78735	0.06678	0.80668	-0.00

In [9]:

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

In [10]:

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

In [11]:

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

```
The Feature Matrix Has 351 Rows and 34 Column(s)
The Target Matrix Has 351 Rows and 1 Column(s)
```

In [12]:

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

In [18]:

```
algorithm=LogisticRegression(penalty='l2',dual=False,tol=1e-4,C=1.0,fit_intercept=True,intercept_scaling=1,class_weight=None,random_state=
```

In [19]:

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

In [20]:

```
observation=[[1,0,0.99539,-0.085889,0.8524299999999999,0.02306,0.8339799999999999,-0.37708,1.0,0.0376,0.8524299999999999,-0.17755,
              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,-0.51171,0.4107800000000003,-0.4616800000000003,0.21260,-0.3409,0.42267,-0.54487,0.18641,-0.453]]
```

In [24]:

```
predictions=Logistic_Regression_Model.predict(observation)
print("The model predicted the observation to belong to class %s"%(predictions))
```

```
The model predicted the observation to belong to class ['g']
```

In [25]:

```
print('The algorithm was Trained to predict one of the Two Classes %s'%(algorithm.classes_))
```

```
The algorithm was Trained to predict one of the Two Classes ['b' 'g']
```

In [26]:

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

The Model says The probabilty of the observation we passed Belonging to class['g'] Is %s"""%(algorithm.predict_proba(observation)[0][1]))
```

```
 The Model says The probabilty of the observation we passed Belonging to class['b'] Is 0.00777308403249255

 The Model says The probabilty of the observation we passed Belonging to class['g'] Is 0.9922269159675075
```

In [ ]:

```

```