In [5]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [6]:
```
df=pd.read_csv(r"C:\Users\Welcome\Downloads\ionosphere.csv")
df
```

Out[6]:

	atr1	atr2	atr3	atr4	atr5	atr6	atr7	atr8	atr9	atr10	...	
0	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.00000	0.03760	...	-0.
1	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	...	-0.
2	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	...	-0.
3	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	...	0.
4	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	...	-0.
...	...	...	...	...	...	...	...	...	...	...	...	
346	1	0	0.83508	0.08298	0.73739	-0.14706	0.84349	-0.05567	0.90441	-0.04622	...	-0.
347	1	0	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.
348	1	0	0.94701	-0.00034	0.93207	-0.03227	0.95177	-0.03431	0.95584	0.02446	...	0.
349	1	0	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	-0.
350	1	0	0.84710	0.13533	0.73638	-0.06151	0.87873	0.08260	0.88928	-0.09139	...	-0.

351 rows × 35 columns

In [7]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [8]:
```
print('This DataFrame ha %d Rows and %d Columns'%(df.shape))
```

```
This DataFrame ha 351 Rows and 35 Columns
```

In [9]:
```
df.head()
```

Out[9]:

	atr1	atr2	atr3	atr4	atr5	atr6	atr7	atr8	atr9	atr10	atr11
0	1	0	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.00000	0.03760	0.85243
1	1	0	1.00000	-0.18829	0.93035	-0.36156	-0.10868	-0.93597	1.00000	-0.04549	0.50874
2	1	0	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	0.73082
3	1	0	1.00000	-0.45161	1.00000	1.00000	0.71216	-1.00000	0.00000	0.00000	0.00000
4	1	0	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	0.52798

In [10]:
```
features_matrix=df.iloc[:,0:34]
```

In [11]:
```
target_vector=df.iloc[:,-1]
```

In [12]:
```
print('The Features Matrix Has %d Rows And %d Columns'%(features_matrix.shape))
print('The Features Matrix Has %d Rows And %d Columns'%(np.array(target_vector)
```

```
The Features Matrix Has 351 Rows And 34 Columns
The Features Matrix Has 351 Rows And 1 Columns
```

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

In [17]:
```
algorithm=LogisticRegression(penalty='l2',dual=False,tol=1e-4,C=1.0,fit_interce
```

In [18]:
```
Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_vec
```

In [19]:
```
Observation=[[1,0,0.99539,-0.05889,0.8542999999999999,0.02306,0.8339799999999999
```

In [20]:
```
predictions=Logistic_Regression_Model.predict(Observation)
print('The Model Predicted The Observations To Belong To Class %s'%(predictions
```

```
The Model Predicted The Observations To Belong To Class ['g']
```

In [21]:
```
print('The Algorithm Was Trained To Predict One Of The Two Classes:%s'%(algorit
```

```
The Algorithm Was Trained To Predict One Of The Two Classes:['b' 'g']
```

In [22]:
```
print("""The Model Says The Probability Of The Observation we Passed Belonging
print("""The Model Says The Probability Of The Observation we Passed Belonging
```

```
The Model Says The Probability Of The Observation we Passed Belonging To clas
s['b']Is 0.007759545690611991
The Model Says The Probability Of The Observation we Passed Belonging To clas
s['g']Is 0.992240454309388
```

In [ ]: