In [2]:

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

In [3]:

```
dc=pd.read_csv(r"C:\Users\91955\Desktop\Data Analysis with Python\ionosphere_data.cs
dc
```

Out[3]:

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

351 rows × 35 columns

In [4]:

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

In [5]:

```
print("This DataFrame has %d rows and %d columns"%(dc.shape))
```

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

In [6]:

```
1 dc.head()
```

Out[6]:

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

In [7]:

```
1 features_matrix=dc.iloc[:,0:34]
```

In [8]:

```
1 target_vector=dc.iloc[:,-1]
2
```

In [9]:

```
1 print("The features matrix has %d rows and %d columns"%(features_matrix.shape))
2 print("The target matrix has %d rows and %d columns"%(np.array(target_vector).reshap
```

```
The features matrix has 351 rows and 34 columns
The target matrix has 351 rows and 1 columns
```

In [10]:

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

In [11]:

```
1 algorithm=LogisticRegression(penalty='l2',dual=False,tol=1e-4,C=1.0,fit_intercept=Tr
```

In [12]:

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

In [20]:

```
1 Observation=[[1,0,0.99539,-0.085889,0.8524299999999999,0.02306,0.8339799999999999,-0
2               -0.17755,0.59755,-0.44945,0.60536,-0.38223,0.8435600000000001,-0.38542
3               0.36946,-0.47357,0.56811,-0.51171,0.4107800000000003,0.461680000000000
4              ]]
```

In [21]:

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

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

In [22]:

```
print('The algorithm was trained to predict one of the two classes:%s'%(algorithm.cl
```

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

In [24]:

```
print("""The model says the probability of the observation we passed belonging to cl
print("""The model says the probability of the observation we passed belonging to cl
```

```
The model says the probability of the observation we passed belonging to c
lass['b'] is 0.008328109136196193
The model says the probability of the observation we passed belonging to c
lass['g'] is 0.9916718908638038
```

In [ ]:

```

```