In [7]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [8]:
```
df=pd.read_csv(r"C:\Users\ubinl\OneDrive\Documents\jupyter\ionosphere.csv")
df
```

Out[8]:

	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	colu
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	0
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	-0
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	0
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	1
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	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
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	1
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	0
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	0
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	0
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	0

351 rows × 35 columns

In [9]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [16]:
```
print("This DataFrame has %d Rows and %d Columns"%(df.shape))
```

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

In [17]:
```
df.head()
```

Out[17]:

	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	c
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	
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	
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	
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	
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	

In [28]:
```
features_matrix=df.iloc[:,0:34]
target_vector=df.iloc[:,-1]
print('The features 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 features matrix has 351 Rows and 34 Columns(s)
The Target matrix has 351 Rows and 1 Columns(s)
```

In [30]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [46]:
```
eight=None,random_state=None,solver='lbfgs',max_iter=100,multi_class='auto',verbose=0,warm_start=False,n_jobs=None,l1_ratio=Non
```

In [47]:
```
Logistic_Regression_model=algorithm.fit(features_matrix_standardized,target_vector)
```

In [49]:
```
-0.29674,0.36946,-0.47357,0.56811,0.51171,0.4107800000000003,-0.461680000000000,0.21266,-0.3409,0.42267,-0.54487,0.18641,-0.453
```

In [52]:
```
predictions=Logistic_Regression_model.predict(observation)
print('The model perdict the observation to belong To class %s'%(predictions))
```

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

In [57]:
```
print('The algorithm was Trained to perdict one of the Two classes: %s'%(algorithm.classes_))
```

```
The algorithm was Trained to perdict one of the Two classes: ['b' 'g']
```

In [58]:
```
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 0.02295488727503503
```

In [55]:
```
says the probability of the observation we passed Belonging To class['g'] is %s"""%(algorithm.predict_proba(observation)[0][1]
```

```
The model says the probability of the observation we passed Belonging To class['g'] is 0.977045112724965
```