```
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\pavan\Downloads\ionosphere.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	colun
0	True	False	0.99539	-0.05889	0.85243	0.02306	0.83398	-0.37708	1.00000	0.03760	...	-0.51171	0.41078	-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.
2	True	False	1.00000	-0.03365	1.00000	0.00485	1.00000	-0.12062	0.88965	0.01198	...	-0.40220	0.58984	-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.
4	True	False	1.00000	-0.02401	0.94140	0.06531	0.92106	-0.23255	0.77152	-0.16399	...	-0.65158	0.13290	-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.
347	True	False	0.95113	0.00419	0.95183	-0.02723	0.93438	-0.01920	0.94590	0.01606	...	0.01361	0.93522	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.
349	True	False	0.90608	-0.01657	0.98122	-0.01989	0.95691	-0.03646	0.85746	0.00110	...	-0.02099	0.89147	-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.

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	column_i	column_j	column_k	column_l	column_m
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
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
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
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
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

```
In [6]: features_matrix=df.iloc[:,0:34]

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

In [8]: print('The Feature 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 Feature Matrix has 351 Rows and 34 columns(s)
        The Target Matrix has 351 Rows and 1 columns(s)

In [9]: features_matrix_standardized=StandardScaler().fit_transform(features_matrix)

In [11]: algorithm=LogisticRegression(penalty='l2',dual=False,tol=1e-4,C=1.0,fit_intercept=True,intercept_scaling=1,class_weigh

In [12]: Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_vector)
```

```
In [14]: observation=[[1,0,0.99539,-0.05889,0.854299999999999,0.02306,0.833979999999999,-0.37708,1.0,0.0376,0.85242999999999
```

```
In [15]: 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 [16]: print('The algoritham was trained to predict one of the two classes:%s'%(algorithm.classes_))
```

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

```
In [17]: print(" " "The Model says the probability of the observation we passed belonging to class['b'] Is %s" " "%(algorithm.p
         print()
```

 The Model says the probability of the observation we passed belonging to class['b'] Is 0.007759545690611991

```
In [18]: print(" " "The Model says the probability of the observation we passed belonging to class['g'] Is %s" " "%(algorithm.p
```

 The Model says the probability of the observation we passed belonging to class['g'] Is 0.992240454309388

```
In [ ]:
```