

```
In [8]: import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

```
In [9]: df=pd.read_csv(r"C:\Users\SASIDHAR ROYAL\Downloads\ionosphere.csv")
df
```

Out[9]:

|     | 1   | 0   | 0.99539 | -0.05889 | 0.85243  | 0.02306  | 0.83398  | -0.37708 | 1.1     | 0.03760  | ... | -0.51171 | 0.41078  | -0.46168 | 0.21266  | -0.34090 | 0.42261  |
|-----|-----|-----|---------|----------|----------|----------|----------|----------|---------|----------|-----|----------|----------|----------|----------|----------|----------|
| 0   | 1   | 0   | 1.00000 | -0.18829 | 0.93035  | -0.36156 | -0.10868 | -0.93597 | 1.00000 | -0.04549 | ... | -0.26569 | -0.20468 | -0.18401 | -0.19040 | -0.11593 | -0.16661 |
| 1   | 1   | 0   | 1.00000 | -0.03365 | 1.00000  | 0.00485  | 1.00000  | -0.12062 | 0.88965 | 0.01198  | ... | -0.40220 | 0.58984  | -0.22145 | 0.43100  | -0.17365 | 0.60401  |
| 2   | 1   | 0   | 1.00000 | -0.45161 | 1.00000  | 1.00000  | 0.71216  | -1.00000 | 0.00000 | 0.00000  | ... | 0.90695  | 0.51613  | 1.00000  | 1.00000  | -0.20099 | 0.25661  |
| 3   | 1   | 0   | 1.00000 | -0.02401 | 0.94140  | 0.06531  | 0.92106  | -0.23255 | 0.77152 | -0.16399 | ... | -0.65158 | 0.13290  | -0.53206 | 0.02431  | -0.62197 | -0.05701 |
| 4   | 1   | 0   | 0.02337 | -0.00592 | -0.09924 | -0.11949 | -0.00763 | -0.11824 | 0.14706 | 0.06637  | ... | -0.01535 | -0.03240 | 0.09223  | -0.07859 | 0.00732  | 0.00001  |
| ... | ... | ... | ...     | ...      | ...      | ...      | ...      | ...      | ...     | ...      | ... | ...      | ...      | ...      | ...      | ...      | ...      |
| 345 | 1   | 0   | 0.83508 | 0.08298  | 0.73739  | -0.14706 | 0.84349  | -0.05567 | 0.90441 | -0.04622 | ... | -0.04202 | 0.83479  | 0.00123  | 1.00000  | 0.12815  | 0.86661  |
| 346 | 1   | 0   | 0.95113 | 0.00419  | 0.95183  | -0.02723 | 0.93438  | -0.01920 | 0.94590 | 0.01606  | ... | 0.01361  | 0.93522  | 0.04925  | 0.93159  | 0.08168  | 0.94001  |
| 347 | 1   | 0   | 0.94701 | -0.00034 | 0.93207  | -0.03227 | 0.95177  | -0.03431 | 0.95584 | 0.02446  | ... | 0.03193  | 0.92489  | 0.02542  | 0.92120  | 0.02242  | 0.92401  |
| 348 | 1   | 0   | 0.90608 | -0.01657 | 0.98122  | -0.01989 | 0.95691  | -0.03646 | 0.85746 | 0.00110  | ... | -0.02099 | 0.89147  | -0.07760 | 0.82983  | -0.17238 | 0.96001  |
| 349 | 1   | 0   | 0.84710 | 0.13533  | 0.73638  | -0.06151 | 0.87873  | 0.08260  | 0.88928 | -0.09139 | ... | -0.15114 | 0.81147  | -0.04822 | 0.78207  | -0.00703 | 0.75701  |

350 rows × 35 columns



```
In [10]: pd.set_option('display.max_rows',1000000000)
pd.set_option('display.max_columns',1000000000)
pd.set_option('display.width',93)
```

```
In [11]: print('This DataFrame has %d Rows and %d columns'%(df.shape))
```

This DataFrame has 350 Rows and 35 columns

```
In [12]: df.head()
```

Out[12]:

|   | 1 | 0 | 0.99539 | -0.05889 | 0.85243  | 0.02306  | 0.83398  | -0.37708 |         | 1.1      | 0.03760 | 0.85243.1 | -0.17755 | 0.59755  | -0.44945 | 0.60536  | -0.38223 | 0.8 |
|---|---|---|---------|----------|----------|----------|----------|----------|---------|----------|---------|-----------|----------|----------|----------|----------|----------|-----|
| 0 | 1 | 0 | 1.00000 | -0.18829 | 0.93035  | -0.36156 | -0.10868 | -0.93597 | 1.00000 | -0.04549 | 0.50874 | -0.67743  | 0.34432  | -0.69707 | -0.51685 | -0.97515 | 0.0      |     |
| 1 | 1 | 0 | 1.00000 | -0.03365 | 1.00000  | 0.00485  | 1.00000  | -0.12062 | 0.88965 | 0.01198  | 0.73082 | 0.05346   | 0.85443  | 0.00827  | 0.54591  | 0.00299  | 0.8      |     |
| 2 | 1 | 0 | 1.00000 | -0.45161 | 1.00000  | 1.00000  | 0.71216  | -1.00000 | 0.00000 | 0.00000  | 0.00000 | 0.00000   | 0.00000  | 0.00000  | -1.00000 | 0.14516  | 0.5      |     |
| 3 | 1 | 0 | 1.00000 | -0.02401 | 0.94140  | 0.06531  | 0.92106  | -0.23255 | 0.77152 | -0.16399 | 0.52798 | -0.20275  | 0.56409  | -0.00712 | 0.34395  | -0.27457 | 0.5      |     |
| 4 | 1 | 0 | 0.02337 | -0.00592 | -0.09924 | -0.11949 | -0.00763 | -0.11824 | 0.14706 | 0.06637  | 0.03786 | -0.06302  | 0.00000  | 0.00000  | -0.04572 | -0.15540 | -0.0     |     |

```
In [13]: features_matrix=df.iloc[:,0:34]
```

```
In [14]: target_vector=df.iloc[:, -1]
```

```
In [15]: print('This features 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))
```

This features matrix has 350 Rows and 34 column(s)  
The target matrix has 350 Rows and 1 column(s)

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

```
In [17]: algorithm=LogisticRegression(penalty=None,dual=False,tol=1e-4,C=1.0,fit_intercept=True,intercept_scaling=1,class_weight
```

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

C:\Users\SASIDHAR ROYAL\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\linear\_model\\_logistic.py:  
458: ConvergenceWarning: lbfgs failed to converge (status=1):  
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html> (<https://scikit-learn.org/stable/modules/preprocessing.html>)

Please also refer to the documentation for alternative solver options:

[https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression) ([https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression))

```
n_iter_i = _check_optimize_result(
```

```
In [19]: observation=[[1,0,0.99539,0.05889,0.8524299999999999,0.02306,0.8339799999999999,-0.37708,1.0,0.0376,0.8524299999999999,
```

```
In [20]: 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 [21]: 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 [22]: """The model says the probability of observation we passed belonging to class['b'] is %s"""%(algorithm.predict_proba(ob
"""The model says the probability of observation we passed belonging to class['g'] is %s"""%(algorithm.predict_proba(ob
```

The model says the probability of observation we passed belonging to class['b'] is 6.327727089583401e-05

The model says the probability of observation we passed belonging to class['g'] is 0.9999367227291042

In [26]:

In [ ]:

In [ ]: