Hill and Valley

Objective

To predict whether a given data point lies in a hill or a valley. This involves training the model on labeled data points that are categorized as either hills or valleys, and then using the trained model to classify new, unseen data points based on their features. The project aims to explore the application of logistic regression in terrain classification and predictive modelling

Data Source

This dataset was taken from the github library which is maintained at YBI Foundation. Each record represents 100 points on a two dimensional graph. When plotted in order (from 1 through 100) as the Y coordinate, the point will create either a Hill (a "bump" in the terrain) or a Valley (a "Dip" in the terrain).

Import libraries

[1]:
```
import pandas as pd
```

[2]:
```
import numpy as np
```

Import data

[3]:
```
hill = pd.read_csv('Hill Valley Dataset.csv')
```

[4]:
```
hill.head()
```

[4]:
```
        V1         V2         V3         V4         V5         V6         V7  \
0      39.02      36.49 38.20 38.85 39.38 39.74 37.02
1      1.83 1.71  1.77  1.77  1.68  1.78  1.80
2      68177.69 66138.42 72981.88 74304.33 67549.66 69369.34 69169.41
3      44889.06 39191.86 40728.46 38576.36 45876.06 47034.00 46611.43
4      5.70 5.40  5.28  5.38  5.27  5.61  6.00

        V8         V9        V10 …        V92        V93        V94        V95 \
0      39.53      38.81 38.79 …     36.62 36.92 38.80 38.52
1      1.70 1.75  1.78 …      1.80  1.79  1.77  1.74
2      73268.61 74465.84 72503.37 … 73438.88 71053.35 71112.62
       74916.48
```

```
3      37668.32 40980.89 38466.15 … 42625.67 40684.20 46960.73
       44546.80
4      5.38 5.34  5.87 …     5.17  5.67  5.60  5.94

        V96      V97      V98      V99     V100 Class
0      38.07      36.73 39.46 37.50 39.10 0
1      1.74 1.80  1.78  1.75  1.69  1
2      72571.58 66348.97 71063.72 67404.27 74920.24     1
3      45410.53 47139.44 43095.68 40888.34 39615.19     0
4      5.73 5.22  5.30  5.73  5.91  0

[5 rows x 101 columns]
```

Describe data

[5]:
```
hill.describe()
```

[5]:
```
V1    V2    V3    V4 \ count 1212.000000      1212.000000
1212.000000      1212.000000 mean 8169.091881
8144.306262      8192.653738      8176.868738 std
17974.950461     17881.049734     18087.938901
17991.903982
min        0.920000      0.900000      0.850000      0.890000
25%       19.602500     19.595000     18.925000     19.277500
50%      301.425000    295.205000    297.260000    299.720000
75%     5358.795000   5417.847500   5393.367500   5388.482500
max   117807.870000 108896.480000 119031.350000 110212.590000

                 V5             V6             V7             V8 \
count   1212.000000   1212.000000   1212.000000   1212.000000
mean    8128.297211   8173.030008   8188.582748   8183.641543
std    17846.757963  17927.114105  18029.562695  18048.582159
min        0.880000      0.860000      0.870000      0.650000
25%       19.210000     19.582500     18.690000     19.062500
50%      295.115000    294.380000    295.935000    290.850000
75%     5321.987500   5328.040000   5443.977500   5283.655000
max   113000.470000 116848.390000 115609.240000 118522.320000

                 V9            V10 …           V92            V93 \
count   1212.000000    1212.000000 …   1212.000000   1212.000000
mean    8154.670066    8120.767574 …   8120.056815   8125.917409
std    17982.390713   17900.798206 …  17773.190621  17758.182403
min        0.650000       0.620000 …      0.870000      0.900000
25%       19.532500      19.285000 …     19.197500     18.895000
50%      294.565000     295.160000 …    297.845000    295.420000
75%     5378.180000    5319.097500 …   5355.355000   5386.037500
```

```
max    112895.900000 117798.300000 … 113858.680000
       112948.830000

                V94           V95           V96           V97  \
count   1212.000000   1212.000000   1212.000000   1212.000000
mean    8158.793812   8140.885421   8213.480611   8185.594002
std    17919.510371  17817.945646  18016.445265  17956.084223
min        0.870000      0.880000      0.890000      0.890000
25%       19.237500     19.385000     19.027500     19.135000
50%      299.155000    293.355000    301.370000    296.960000
75%     5286.385000   5345.797500   5300.890000   5361.047500
max    112409.570000 112933.730000 112037.220000 115110.420000

                V98           V99          V100        Class
count   1212.000000   1212.000000  1212.000000 1212.000000
mean    8140.195355   8192.960891   8156.197376    0.500000
std    17768.356106  18064.781479  17829.310973    0.500206
min        0.860000      0.910000      0.890000    0.000000
25%       19.205000     18.812500     19.145000    0.000000
50%      300.925000    299.200000    302.275000    0.500000
75%     5390.850000   5288.712500   5357.847500    1.000000
max    116431.960000 113291.960000 114533.7600001.000000
[8 rows x 101 columns]
```

Data preprocessing

[6]:
```python
hill.columns
```

[6]:
```
Index(['V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',
      …
      'V92', 'V93', 'V94', 'V95', 'V96', 'V97', 'V98', 'V99',
      'V100',
      'Class'],
     dtype='object',
     length=101)
```

[7]:
```python
hill['Class'].value_counts()
```

[7]:
```
0    606
1    606
Name: Class, dtype: int64
```

Define target(y) and feature(X)

[8]:
```python
y=hill['Class']
```

[9]:
```python
y.shape
```

[9]: (1212,)

[10]:

```
y
```

[10]: 0 0

1 1

2 1

3 0

4 0

..

1207 1

1208 0

1209 1

1210 1

1211 0

Name: Class, Length: 1212, dtype: int64

[11]:
```
X=hill.drop('Class',axis=1)
```

[12]:
```
X.shape
```

[12]: (1212, 100)

[13]:

```
X
```

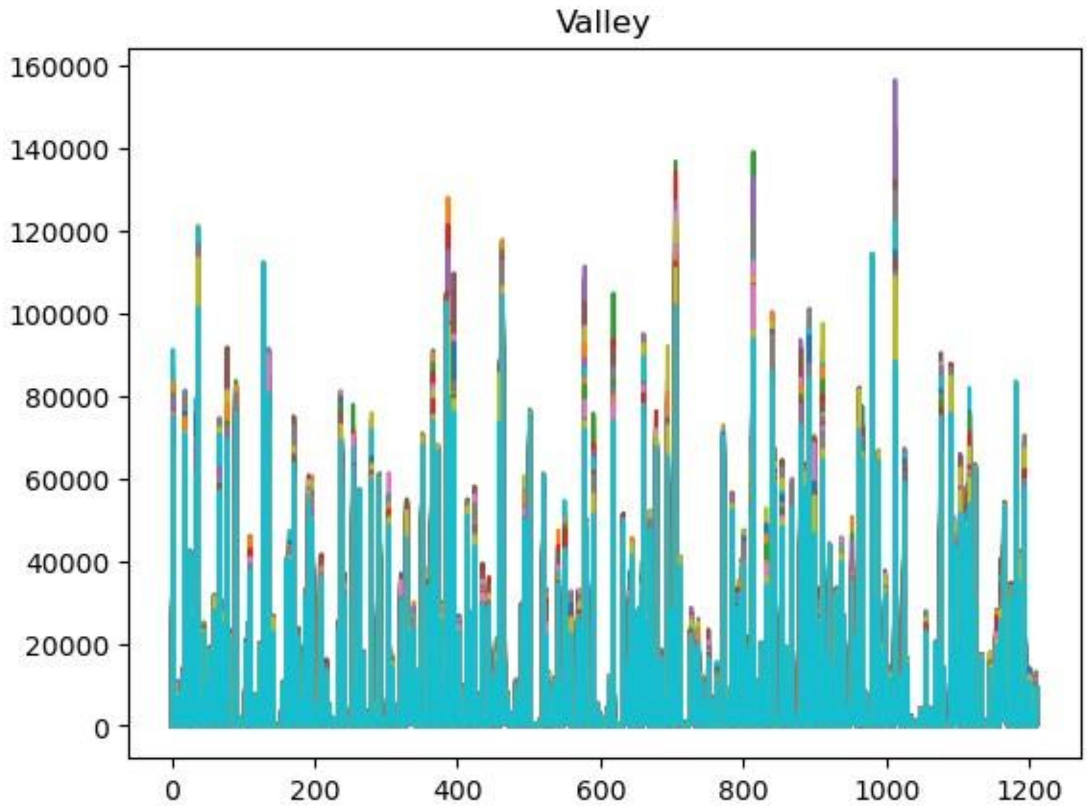
[13]: V1 V2 V3 V4 V5 V6 V7 \
0 39.02 36.49 38.20 38.85 39.38 39.74 37.02 1 1.83 1.71 1.77 1.77 1.68 1.78 1.80
2 68177.69 66138.42 72981.88 74304.33 67549.66 69367.34 69169.41
3 44889.06 39191.86 40728.46 38576.36 45876.06 47034.00 46611.43
4 5.70 5.40 5.28 5.38 5.27 5.61 6.00
… … … … … … … …
1207 13.00 12.87 13.27 13.04 13.19 12.53 14.31
1208 48.66 50.11 48.55 50.43 50.09 49.67 48.95
1209 10160.65 9048.63 8994.94 9514.39 9814.74 10195.24 10031.47
1210 34.81 35.07 34.98 32.37 34.16 34.03 33.31

```
1211    8489.43  7672.98    9132.14    7985.73    8226.85    8554.28
           8838.87

          V8       V9       V10 …       V91      V92      V93 \
0 39.53 38.81 38.79 … 37.57 36.62 36.92 1 1.70 1.75 1.78 …
1.71 1.80 1.79
2        73268.61 74465.84 72503.37 … 69384.71 73438.88 71053.35
3        37668.32 40980.89 38466.15 … 47653.60 42625.67 40684.20
4        5.38    5.34  5.87 …     5.52  5.17  5.67
…       …      …      … …       …      …      …
1207    13.33    13.63 14.55 …    12.89 12.48 12.15
1208    48.65    48.63 48.61 …    47.45 46.93 49.61
1209    10202.28 9152.99    9591.75 … 10413.41    9068.11    9191.80
1210    32.48    35.63 32.48 …    33.18 32.76 35.03
1211    8967.24  8635.14    8544.37 …  7747.70    8609.73    9209.48

         V94      V95      V96      V97      V98      V99     V100
0 38.80 38.52 38.07 36.73 39.46 37.50 39.10 1 1.77 1.74 1.74 1.80
1.78 1.75 1.69
2        71112.62 74916.48 72571.58 66348.97 71063.72 67404.27
         74920.24
3        46960.73 44546.80 45410.53 47139.44 43095.68 40888.34
         39615.19
4        5.60    5.94  5.73  5.22  5.30  5.73  5.91
…       …      …      …      …      …      …      …
1207    13.15    12.35 13.58 13.86 12.88 13.87 13.51
1208    47.16    48.17 47.94 49.81 49.89 47.43 47.77
1209    9275.04  9848.18    9074.17    9601.74 10366.24 8997.60
           9305.77
1210    32.89    31.91 33.85 35.28 32.49 32.83 34.82
1211    8496.33  8724.01    8219.99    8550.86    8679.43    8389.31
           8712.80

[1212 rows x 100 columns]
```

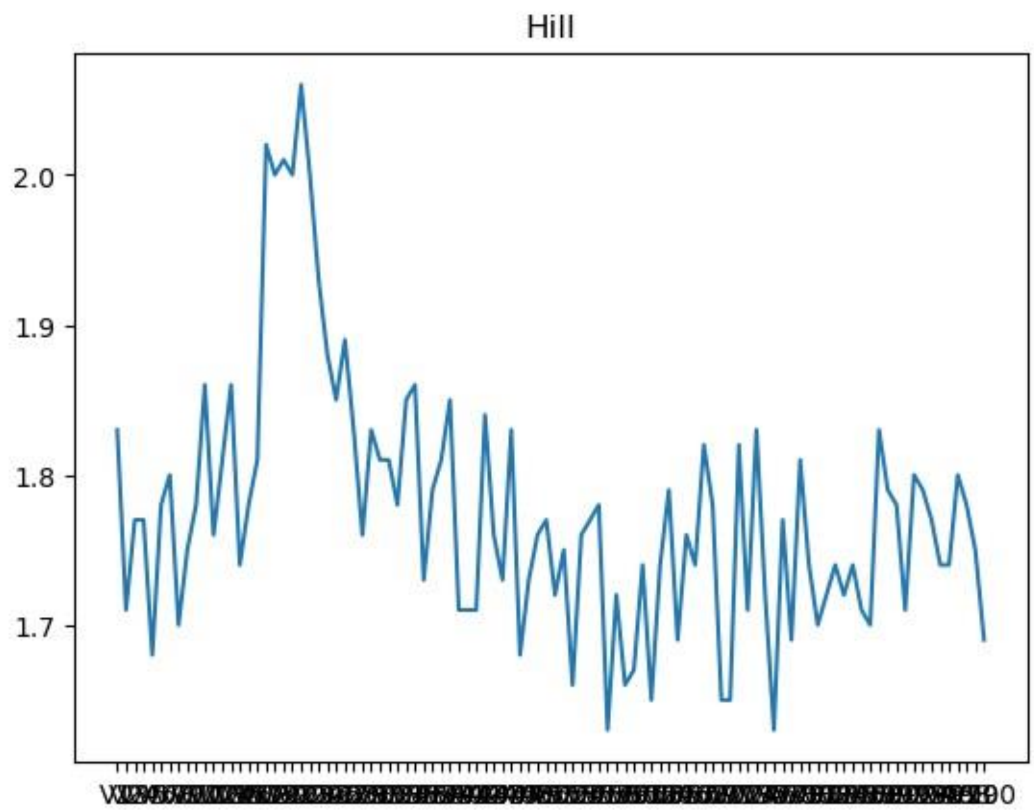
Data Visualization

[14]:
```python
import matplotlib.pyplot as plt
```

[17]:
```python
plt.plot(X.iloc[0:])
plt.title('Valley');
```



[18]:
```
plt.plot(X.iloc[1,:])
plt.title('Hill');
```



Train test split

```
[19]: from sklearn.model_selection import train_test_split
```

```
[20]: X_train,X_test,y_train,y_test=train_test_split(X,y, test_size=0.3,
       stratify=y,random_state=2529)
```

```
[21]: X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
[21]: ((848, 100), (364, 100), (848,), (364,))
```

Modelling

```
[22]: from sklearn.linear_model import LogisticRegression
```

```
[23]: LR=LogisticRegression()
```

```
[24]: LR.fit(X_train,y_train)
```

```
C:\Users\nfps2\anaconda3\Lib\sitepackages\sklearn\linear_model\_logis
tic.py:460: 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
Please also refer to the documentation for alternative solver
options:
                                              https://scikit-
        learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[24]: LogisticRegression()

Model prediction

[25]:
```
y_pred=LR.predict(X_test)
```

[26]:
```
y_pred
```

[26]:
```
array([0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1,
1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0,
                                                    1, 0, 1, 1,
       0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       1,
       0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0,
       0,
       1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1,
       1,
       1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1,
       0,
       0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       0,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       1,
       0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0,
       1,
       1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1,
       1,
       0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1,
       1,
       0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1,
       0,
       1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0,
       0,
```

```
       1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       0,
       1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0,
       0,
       0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1], dtype=int64)
```

[27]:
```
y_pred.shape
```

[27]:
```
(364,)
```

[28]:
```
LR.predict_proba(X_test)
```

[28]:
```
array([[1.00000000e+000, 1.45527801e-039],
       [3.27055781e-005, 9.99967294e-001],
       [1.00000000e+000, 5.28846506e-067],
       [1.94738258e-002, 9.80526174e-001],
       [1.58892426e-001, 8.41107574e-001],
       [5.85050347e-001, 4.14949653e-001],
       [3.44887077e-001, 6.55112923e-001],
       [1.00000000e+000, 1.75541727e-180],
       [8.97351886e-001, 1.02648114e-001],
       [9.68989178e-001, 3.10108218e-002],
       [1.44591103e-001, 8.55408897e-001],
       [9.99855611e-001, 1.44388628e-004],
       [1.00000000e+000, 4.10091161e-283],
       [9.62074285e-001, 3.79257154e-002],
       [1.00000000e+000, 5.11380981e-018],
       [1.00000000e+000, 2.55440072e-293],
       [0.00000000e+000, 1.00000000e+000],
       [1.30991286e-001, 8.69008714e-001],
       [4.74479838e-001, 5.25520162e-001],
       [5.16883390e-001, 4.83116610e-001],
       [0.00000000e+000, 1.00000000e+000],
       [0.00000000e+000, 1.00000000e+000],
       [5.16475850e-001, 4.83524150e-001],
       [0.00000000e+000, 1.00000000e+000],
       [6.02618174e-001, 3.97381826e-001],
       [3.74841582e-002, 9.62515842e-001],
       [5.85861125e-001, 4.14138875e-001],
       [1.45004935e-004, 9.99854995e-001],
       [4.87993323e-001, 5.12006677e-001],
       [8.25295482e-002, 9.17470452e-001],
       [0.00000000e+000, 1.00000000e+000],
       [9.99999928e-001, 7.15632211e-008],
       [0.00000000e+000, 1.00000000e+000],
       [1.00000000e+000, 0.00000000e+000],
```

```
[5.11575699e-001, 4.88424301e-001],
[5.01066055e-001, 4.98933945e-001],
[5.05715706e-001, 4.94284294e-001],
[4.43168738e-001, 5.56831262e-001],
[9.99962283e-001, 3.77168713e-005],
[1.00000000e+000, 1.81499684e-054],
[4.86845230e-001, 5.13154770e-001],
[1.00000000e+000, 4.94774494e-014],
[0.00000000e+000, 1.00000000e+000],
[3.03278716e-003, 9.96967213e-001],
[1.00000000e+000, 9.04593353e-012],
[0.00000000e+000, 1.00000000e+000],
[3.32677064e-001, 6.67322936e-001],
[9.27532513e-006, 9.99990725e-001],
[6.73174313e-007, 9.99999327e-001],
[5.77002801e-001, 4.22997199e-001],
[5.06516081e-001, 4.93483919e-001],
[1.00000000e+000, 1.49897393e-028],
[4.55831144e-001, 5.44168856e-001],
[4.59833225e-002, 9.54016677e-001],
[9.99998397e-001, 1.60275884e-006],
[4.68178466e-001, 5.31821534e-001],
[4.98146445e-001, 5.01853555e-001],
[4.38895012e-001, 5.61104988e-001], [1.00000000e+000,
3.63011475e-076],
[0.00000000e+000, 1.00000000e+000],
[3.15402369e-004, 9.99684598e-001],
[4.81209327e-001, 5.18790673e-001],
[4.91646114e-001, 5.08353886e-001],
[0.00000000e+000, 1.00000000e+000],
[8.20251298e-001, 1.79748702e-001],
[4.68067901e-001, 5.31932099e-001],
[1.00000000e+000, 2.10082628e-044],
[0.00000000e+000, 1.00000000e+000],
[5.29065002e-001, 4.70934998e-001],
[1.39999123e-012, 1.00000000e+000],
[6.61035397e-001, 3.38964603e-001],
[4.97759513e-001, 5.02240487e-001],
[9.94040003e-001, 5.95999719e-003],
[3.57146544e-001, 6.42853456e-001],
[7.85881291e-001, 2.14118709e-001],
[1.00000000e+000, 3.85011931e-101],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 1.06999270e-285],
```

```
[1.00000000e+000, 8.24488840e-030],
[4.77964229e-001, 5.22035771e-001],
[5.04876707e-001, 4.95123293e-001],
[2.31971008e-001, 7.68028992e-001],
[0.00000000e+000, 1.00000000e+000],
[4.73775708e-001, 5.26224292e-001],
[2.42510129e-004, 9.99757490e-001],
[5.07014558e-001, 4.92985442e-001],
[7.59975209e-001, 2.40024791e-001],
[4.86600420e-001, 5.13399580e-001],
[8.48177103e-002, 9.15182290e-001],
[1.00000000e+000, 1.18229364e-275],
[5.03752658e-001, 4.96247342e-001],
[9.73292515e-001, 2.67074847e-002],
[1.00000000e+000, 4.36606778e-019],
[4.93401912e-001, 5.06598088e-001],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 1.39125702e-116],
[5.23937799e-001, 4.76062201e-001],
[1.00000000e+000, 8.72883096e-058],
[1.00000000e+000, 6.25571943e-012],
[1.63923226e-008, 9.99999984e-001],
[1.00000000e+000, 2.93719854e-056],
[7.13602273e-001, 2.86397727e-001],
[5.79291056e-004, 9.99420709e-001], [1.00000000e+000,
1.06516138e-206],
[4.19048746e-001, 5.80951254e-001],
[3.53043668e-005, 9.99964696e-001],
[4.93977308e-001, 5.06022692e-001],
[2.11214258e-001, 7.88785742e-001],
[0.00000000e+000, 1.00000000e+000],
[3.94367747e-001, 6.05632253e-001],
[7.85683902e-001, 2.14316098e-001],
[5.30747642e-001, 4.69252358e-001],
[1.00000000e+000, 2.06106863e-101],
[0.00000000e+000, 1.00000000e+000],
[3.47145821e-001, 6.52854179e-001],
[3.31909587e-005, 9.99966809e-001],
[1.00000000e+000, 0.00000000e+000],
[9.99999990e-001, 1.02242401e-008],
[9.68308623e-001, 3.16913769e-002],
[0.00000000e+000, 1.00000000e+000],
[5.28494251e-001, 4.71505749e-001],
[0.00000000e+000, 1.00000000e+000],
```

```
[5.40224580e-001, 4.59775420e-001],
[5.29916176e-001, 4.70083824e-001],
[1.48015731e-001, 8.51984269e-001],
[9.98493269e-001, 1.50673111e-003],
[8.59152952e-001, 1.40847048e-001],
[6.17089810e-002, 9.38291019e-001],
[3.14521222e-003, 9.96854788e-001],
[1.00000000e+000, 1.10779520e-035],
[6.81525463e-001, 3.18474537e-001],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 1.93890429e-041],
[4.37814727e-001, 5.62185273e-001],
[9.99990829e-001, 9.17098210e-006],
[4.66287843e-001, 5.33712157e-001],
[1.27658417e-009, 9.99999999e-001],
[1.08293027e-001, 8.91706973e-001],
[4.86030914e-001, 5.13969086e-001],
[1.00000000e+000, 2.42623716e-145],
[5.65119683e-001, 4.34880317e-001],
[9.64451866e-001, 3.55481338e-002],
[6.18200054e-001, 3.81799946e-001],
[1.00000000e+000, 8.02669835e-073],
[4.98157217e-001, 5.01842783e-001],
[1.00000000e+000, 1.08750820e-205],
[5.54828616e-001, 4.45171384e-001],
[5.16823055e-001, 4.83176945e-001],
[4.67903753e-001, 5.32096247e-001],
[5.14367581e-001, 4.85632419e-001], [1.00000000e+000,
0.00000000e+000],
[5.15569917e-001, 4.84430083e-001],
[5.08965842e-001, 4.91034158e-001],
[0.00000000e+000, 1.00000000e+000],
[5.09368266e-001, 4.90631734e-001],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 2.92257383e-034],
[9.99995949e-001, 4.05110408e-006],
[5.23003962e-001, 4.76996038e-001],
[1.00000000e+000, 3.66872651e-173],
[9.95901442e-001, 4.09855751e-003],
[7.70855342e-001, 2.29144658e-001],
[4.27426239e-001, 5.72573761e-001],
[1.00000000e+000, 6.49429848e-116],
[1.00000000e+000, 7.24039995e-056],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 3.14067150e-010],
```

```
[9.32569147e-001, 6.74308531e-002],
[9.99451296e-001, 5.48704458e-004],
[5.47562987e-001, 4.52437013e-001],
[1.00000000e+000, 7.99872532e-142],
[7.73705800e-001, 2.26294200e-001],
[1.00000000e+000, 1.05573785e-031],
[2.16341162e-006, 9.99997837e-001],
[1.00000000e+000, 0.00000000e+000],
[5.09599227e-001, 4.90400773e-001],
[1.00000000e+000, 0.00000000e+000],
[1.00000000e+000, 5.49020373e-042],
[7.11454961e-001, 2.88545039e-001],
[5.64173428e-001, 4.35826572e-001],
[4.72495149e-001, 5.27504851e-001],
[1.00000000e+000, 0.00000000e+000],
[4.98017430e-001, 5.01982570e-001],
[6.47784417e-001, 3.52215583e-001],
[0.00000000e+000, 1.00000000e+000],
[9.99993653e-001, 6.34682260e-006],
[1.40293166e-002, 9.85970683e-001],
[0.00000000e+000, 1.00000000e+000],
[5.04985932e-001, 4.95014068e-001],
[0.00000000e+000, 1.00000000e+000],
[2.90702999e-007, 9.99999709e-001],
[0.00000000e+000, 1.00000000e+000],
[4.74946586e-001, 5.25053414e-001],
[7.06857505e-001, 2.93142495e-001],
[9.99999925e-001, 7.50819613e-008],
[1.85380730e-001, 8.14619270e-001],
[8.18397350e-010, 9.99999999e-001], [2.72946551e-003,
9.97270534e-001],
[5.10534583e-001, 4.89465417e-001],
[1.00000000e+000, 3.10725461e-019],
[1.00000000e+000, 1.35538339e-012],
[0.00000000e+000, 1.00000000e+000],
[4.68232564e-010, 1.00000000e+000],
[2.51654276e-001, 7.48345724e-001],
[4.94055882e-001, 5.05944118e-001],
[3.94517834e-001, 6.05482166e-001],
[4.81042365e-001, 5.18957635e-001],
[1.00000000e+000, 1.68297930e-033],
[1.00000000e+000, 8.37362865e-075],
[4.42501059e-001, 5.57498941e-001],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 2.35969865e-108],
```

```
[0.00000000e+000, 1.00000000e+000],
[5.10866537e-001, 4.89133463e-001],
[3.06788593e-001, 6.93211407e-001],
[5.07777455e-001, 4.92222545e-001],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[5.26159602e-001, 4.73840398e-001],
[4.94116597e-005, 9.99950588e-001],
[1.00000000e+000, 1.37994050e-135],
[0.00000000e+000, 1.00000000e+000],
[5.02705122e-001, 4.97294878e-001],
[0.00000000e+000, 1.00000000e+000],
[4.85053688e-001, 5.14946312e-001],
[2.75130303e-001, 7.24869697e-001],
[0.00000000e+000, 1.00000000e+000],
[4.91942265e-001, 5.08057735e-001],
[0.00000000e+000, 1.00000000e+000],
[4.65651867e-001, 5.34348133e-001],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[7.45759996e-001, 2.54240004e-001],
[1.00000000e+000, 6.09380084e-015],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[3.90420581e-001, 6.09579419e-001],
[6.47776984e-001, 3.52223016e-001],
[0.00000000e+000, 1.00000000e+000],
[2.36334763e-002, 9.76366524e-001],
[1.00000000e+000, 0.00000000e+000],
[4.55362215e-009, 9.99999995e-001],
[4.59222505e-001, 5.40777495e-001],
[9.86340679e-001, 1.36593205e-002], [1.00000000e+000,
1.65491470e-157],
[2.18815416e-002, 9.78118458e-001],
[2.82292968e-005, 9.99971771e-001],
[5.08852491e-001, 4.91147509e-001],
[5.22694794e-001, 4.77305206e-001],
[0.00000000e+000, 1.00000000e+000],
[4.44390571e-001, 5.55609429e-001],
[0.00000000e+000, 1.00000000e+000],
[5.04877561e-001, 4.95122439e-001],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 1.68644646e-034],
[5.03471851e-001, 4.96528149e-001],
[4.87671439e-001, 5.12328561e-001],
```

```
[6.69039004e-001, 3.30960996e-001],
[1.21570820e-009, 9.99999999e-001],
[1.00000000e+000, 1.59266807e-024],
[5.41660270e-008, 9.99999946e-001],
[5.10309153e-001, 4.89690847e-001],
[5.69737592e-002, 9.43026241e-001],
[1.40230190e-001, 8.59769810e-001],
[4.64813064e-001, 5.35186936e-001],
[5.02108071e-001, 4.97891929e-001],
[1.00000000e+000, 4.05332607e-110],
[4.95849286e-001, 5.04150714e-001],
[2.86604859e-001, 7.13395141e-001],
[4.30648039e-001, 5.69351961e-001],
[9.97122840e-001, 2.87716035e-003],
[5.30083710e-001, 4.69916290e-001],
[5.98650907e-002, 9.40134909e-001],
[0.00000000e+000, 1.00000000e+000],
[4.69978448e-001, 5.30021552e-001],
[1.00000000e+000, 0.00000000e+000],
[4.91444383e-001, 5.08555617e-001],
[4.75846998e-001, 5.24153002e-001],
[4.80284733e-001, 5.19715267e-001],
[4.87970387e-001, 5.12029613e-001],
[9.99999750e-001, 2.50397119e-007],
[8.35144322e-001, 1.64855678e-001],
[7.88120469e-001, 2.11879531e-001],
[9.99095074e-001, 9.04925541e-004],
[4.93915545e-001, 5.06084455e-001],
[4.90688058e-001, 5.09311942e-001],
[0.00000000e+000, 1.00000000e+000],
[5.08427646e-001, 4.91572354e-001],
[9.78474334e-001, 2.15256660e-002],
[7.98971040e-001, 2.01028960e-001],
[0.00000000e+000, 1.00000000e+000], [5.01851832e-001,
4.98148168e-001],
[0.00000000e+000, 1.00000000e+000],
[1.00000000e+000, 5.41314025e-020],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[4.63055862e-001, 5.36944138e-001],
[0.00000000e+000, 1.00000000e+000],
[3.80269999e-001, 6.19730001e-001],
[5.00257268e-001, 4.99742732e-001],
[4.96091013e-001, 5.03908987e-001],
```

[6.80660871e-005, 9.99931934e-001],
[6.59835884e-002, 9.34016412e-001],
[0.00000000e+000, 1.00000000e+000],
[5.07097198e-001, 4.92902802e-001],
[0.00000000e+000, 1.00000000e+000],
[1.97687825e-001, 8.02312175e-001],
[5.03205071e-001, 4.96794929e-001],
[5.71595642e-001, 4.28404358e-001],
[2.05873610e-003, 9.97941264e-001],
[0.00000000e+000, 1.00000000e+000],
[5.28252997e-011, 1.00000000e+000],
[4.97518614e-001, 5.02481386e-001],
[9.68689231e-001, 3.13107692e-002],
[1.00000000e+000, 4.03231154e-185],
[5.52089092e-001, 4.47910908e-001],
[6.50789456e-001, 3.49210544e-001],
[5.85547999e-001, 4.14452001e-001],
[0.00000000e+000, 1.00000000e+000],
[4.92293012e-001, 5.07706988e-001],
[9.54791801e-015, 1.00000000e+000],
[6.56513426e-001, 3.43486574e-001],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[7.40163153e-009, 9.99999993e-001],
[1.00000000e+000, 3.60501895e-268],
[6.24382025e-001, 3.75617975e-001],
[1.00000000e+000, 5.48961926e-174],
[6.74766807e-001, 3.25233193e-001],
[0.00000000e+000, 1.00000000e+000],
[0.00000000e+000, 1.00000000e+000],
[9.85072301e-001, 1.49276988e-002],
[0.00000000e+000, 1.00000000e+000],
[3.76206589e-001, 6.23793411e-001],
[9.99999996e-001, 3.63618517e-009],
[8.42308055e-001, 1.57691945e-001],
[1.00000000e+000, 0.00000000e+000], [5.43896884e-001,
4.56103116e-001],
[1.00000000e+000, 5.51476638e-015],
[1.00000000e+000, 1.75386932e-124],
[4.19341790e-001, 5.80658210e-001],
[9.14512910e-012, 1.00000000e+000],
[1.00000000e+000, 3.80042574e-163],
[9.77687965e-001, 2.23120345e-002],
[9.78968819e-001, 2.10311814e-002],
[1.43616364e-001, 8.56383636e-001],

```
       [6.63861274e-001, 3.36138726e-001],
       [5.82098500e-001, 4.17901500e-001],
       [6.85095846e-001, 3.14904154e-001],
       [1.00000000e+000, 1.49168954e-032],
       [6.37242259e-010, 9.99999999e-001],
       [5.34915117e-001, 4.65084883e-001],
       [0.00000000e+000, 1.00000000e+000],
       [9.94722280e-003, 9.90052777e-001],
       [5.09272228e-001, 4.90727772e-001],
       [1.00000000e+000, 1.68806630e-215],
       [5.29338174e-001, 4.70661826e-001],
       [4.73405345e-001, 5.26594655e-001],
       [0.00000000e+000, 1.00000000e+000],
       [3.38446918e-001, 6.61553082e-001],
       [0.00000000e+000, 1.00000000e+000]])
```

Model Evaluation

[29]:
```python
from sklearn.metrics import confusion_matrix,classification_report
```

[30]:
```python
print(confusion_matrix(y_test,y_pred))
```

```
[[175   7]
 [  3 179]]
```

[31]:
```python
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.96      0.97       182
           1       0.96      0.98      0.97       182

    accuracy                           0.97       364
   macro avg       0.97      0.97      0.97       364
weighted
avg                0.97      0.97      0.97       364
```

[ ]:

Explanation

Accuracy in machine leaning model is used for Classification. Accuracy score in Machine Learning model means number of correct predictions. It is the ratio of number of correct predictions to the total number of predictions. In machine learning model accuracy score above 0.7 is treated as good-to-go-model; here, our accuracy score is 0.97 therefore our Machine learning model is 97% accurate in correct predictions.