

```
1 import numpy as np
2 import pandas as pd
3
4 SPAM = 1
5 NON_SPAM = 0
6
7
8 if __name__ == '__main__':
9     train = pd.read_csv("spambase.train", header=
None)
10    test = pd.read_csv("spambase.test", header=None
)
11
12    train = pd.concat([train, test], ignore_index=
True, sort=False)
13
14    number_correct = 0
15    number_wrong = 0
16
17    # number of rows in Test dataset
18    test_rows = test.shape[0]
19    # number of columns in Test dataset
20    test_columns = test.shape[1]
21
22    # number of rows in Training dataset
23    train_rows = train.shape[0]
24    # number of columns in Training dataset
25    train_columns = train.shape[1]
26
27    # Training dataset with Y = 1
28    train_y_1 = train[train[57].isin([1])]
29    number_train_y_1 = train_y_1.shape[0]
30
31    # Training dataset with Y = 0
32    train_y_0 = train[train[57].isin([0])]
33    number_train_y_0 = train_y_0.shape[0]
34
35    # Probability(Y = 1)
36    probability_y_1 = number_train_y_1 / train_rows
37    # Probability(Y = 0)
38    probability_y_0 = number_train_y_0 / train_rows
```

```

39
40     medians = []
41     for column in range(train_columns - 1):
42         medians.append(train[column].median())
43         # print(f"column = {column}, median = {
train[column].median()}")
44
45     """
46     median_condition_probability = np.array([[1, 2
, 3]])
47
48     for column in range(train_columns - 1):
49         median = medians[column]
50
51         number_match_1 = train_y_1[train_y_1[column
] <= median].shape[0]
52         theta_1 = number_match_1 / number_train_y_1
53
54         number_match_0 = train_y_0[train_y_0[column
] <= median].shape[0]
55         theta_0 = number_match_0 / number_train_y_0
56
57         row_median_proba = np.array([[median,
theta_1, theta_0]])
58
59         if column == 0:
60             median_condition_probability =
row_median_proba
61         else:
62
63             median_condition_probability = np.
concatenate((median_condition_probability,
row_median_proba))
64     """
65     #print(median_condition_probability)
66     #number_match_1 = train_y_1[train_y_1[11] < 0.
145000000000000002].shape[0]
67     #print(f"number_match_1 = {number_match_1}")
68     #print(f"number_train_y_1 = {number_train_y_1
}")
69     #print(f"number_train_y_1 = {number_train_y_0

```

```
69 }")
70
71     for row in range(test_rows):
72         predict_value = 0
73         real_value = test.iloc[row, test_columns
74             - 1]
75         # For spam (Y=1) which indicates label = 1
76         probability_spam = 1
77         # For Non-spam (Y=0) which indicates label
78         = 0
79         probability_non_spam = 1
80
81         for column in range(test_columns - 1):
82             new_value = test.iloc[row, column]
83             median = medians[column]
84
85             number_match_1 = train_y_1[train_y_1[
86 column] <= median].shape[0]
87             theta_1 = number_match_1 /
88 number_train_y_1
89             number_match_0 = train_y_0[train_y_0[
90 column] <= median].shape[0]
91             theta_0 = number_match_0 /
92 number_train_y_0
93
94             if new_value > median:
95                 theta_1 = 1 - theta_1
96                 theta_0 = 1 - theta_0
97
98                 probability_spam = probability_spam *
99                 theta_1
100                 probability_non_spam =
101                 probability_non_spam * theta_0
102
103                 probability_spam = probability_spam *
104                 probability_y_1
105                 probability_non_spam =
106                 probability_non_spam * probability_y_0
```

```
100
101         #print(f"row = {row}, probability_spam = {
    probability_spam}, probability_non_spam = {
    probability_non_spam}")
102
103         if probability_spam >=
    probability_non_spam:
104             predict_value = 1
105         else:
106             predict_value = 0
107
108         #print(f"row = {row}, real_value = {
    real_value}, predict_value = {predict_value}")
109
110         if predict_value == real_value:
111             number_correct = number_correct + 1
112         else:
113             number_wrong = number_wrong + 1
114
115         print(f"number_correct = {number_correct}")
116         print(f"number_wrong = {number_wrong}")
117
118         print(f"Test Error: {number_wrong / test_rows}
    ")
119
120
```