```python
import numpy as np
import pandas as pd

SPAM = 1
NON_SPAM = 0


if __name__ == '__main__':
    train = pd.read_csv("spambase.train", header=
None)
    test = pd.read_csv("spambase.test", header=None
)

    train = pd.concat([train, test], ignore_index=
True, sort=False)

    number_correct = 0
    number_wrong = 0

    # number of rows in Test dataset
    test_rows = test.shape[0]
    # number of columns in Test dataset
    test_columns = test.shape[1]

    # number of rows in Training dataset
    train_rows = train.shape[0]
    # number of columns in Training dataset
    train_columns = train.shape[1]

    # Training dataset with Y = 1
    train_y_1 = train[train[57].isin([1])]
    number_train_y_1 = train_y_1.shape[0]

    # Training dataset with Y = 0
    train_y_0 = train[train[57].isin([0])]
    number_train_y_0 = train_y_0.shape[0]

    # Probability(Y = 1)
    probability_y_1 = number_train_y_1 / train_rows
    # Probability(Y = 0)
    probability_y_0 = number_train_y_0 / train_rows

    medians = []
    for column in range(train_columns - 1):
        medians.append(train[column].median())
        # print(f"column = {column}, median = {train[column].median()}")

    """
    median_condition_probability = np.array([[1, 2, 3]])

    for column in range(train_columns - 1):
        median = medians[column]

        number_match_1 = train_y_1[train_y_1[column] <= median].shape[0]
        theta_1 = number_match_1 / number_train_y_1

        number_match_0 = train_y_0[train_y_0[column] <= median].shape[0]
        theta_0 = number_match_0 / number_train_y_0

        row_median_proba = np.array([[median, theta_1, theta_0]])

        if column == 0:
            median_condition_probability = row_median_proba
        else:

            median_condition_probability = np.concatenate((median_condition_probability, row_median_proba))
    """
    #print(median_condition_probability)
    #number_match_1 = train_y_1[train_y_1[11] < 0.14500000000000002].shape[0]
    #print(f"number_match_1 = {number_match_1}")
    #print(f"number_train_y_1 = {number_train_y_1}")
    #print(f"number_train_y_1 = {number_train_y_0
}")

    for row in range(test_rows):
        predict_value = 0
        real_value = test.iloc[row, test_columns - 1]

        # For spam (Y=1) which indicates label = 1
        probability_spam = 1
        # For Non-spam (Y=0) which indicates label = 0
        probability_non_spam = 1

        for column in range(test_columns - 1):
            new_value = test.iloc[row, column]

            median = medians[column]

            number_match_1 = train_y_1[train_y_1[column] <= median].shape[0]
            theta_1 = number_match_1 / number_train_y_1

            number_match_0 = train_y_0[train_y_0[column] <= median].shape[0]
            theta_0 = number_match_0 / number_train_y_0

            if new_value > median:
                theta_1 = 1 - theta_1
                theta_0 = 1 - theta_0

            probability_spam = probability_spam * theta_1
            probability_non_spam = probability_non_spam * theta_0

        probability_spam = probability_spam * probability_y_1
        probability_non_spam = probability_non_spam * probability_y_0

        #print(f"row = {row}, probability_spam = {probability_spam}, probability_non_spam = {probability_non_spam}")

        if probability_spam >= probability_non_spam:
            predict_value = 1
        else:
            predict_value = 0

        #print(f"row = {row}, real_value = {real_value}, predict_value = {predict_value}")

        if predict_value == real_value:
            number_correct = number_correct + 1
        else:
            number_wrong = number_wrong + 1

    print(f"number_correct = {number_correct}")
    print(f"number_wrong = {number_wrong}")

    print(f"Test Error: {number_wrong / test_rows}")


```