Implementing Minimum Error Rate Classifier

Rashedun Nobi Chowdhury
ID: 160204039
Group: A2
Email: 160204039@aust.edu

***Abstract*—One major application of machine learning is classification of data. There are two types of model for data classification, generative approach and discriminant approach. Generative approaches estimate the discriminant function by first estimating the probability distribution of the patterns belonging to each class. The discriminant function separates the two classes of data. Minimum error rate classifier is an example of a generative approach model. In this experiment, I implement the Minimum Error Rate classifier as a binary classifier. I also conduct some basic experiments on the model.**

***Index Terms*—Generative model, Machine Learning, Bayesian Theory, Binary Classification**

I. Introduction

Minimum Error Rate classifier is one of the basic generative approach model. It works with the probability distribution of data. Using the variance and mean of data, the model is able to properly classify a data. In this experiment, I work with two classes and the mean and variance for both classes are given. According to Bayesian Decision Theory, we know that Posterior probability is equal to the multiple of prior probability and likelihood. I would like to mention that for Minimum Error Rate classifier, the discriminant function is the posterior probability. The posterior probability can be written as,

$$P(\omega_i|x) = \frac{P(x|\omega_i)P(\omega_i)}{P(x)} \tag{1}$$

As our dataset is known to show the normal distribution, we are able to calculate the likelihood using the normal distribution formula. The prior probability for both the classes are given. The rest of the report is organized as follows, section 2 contains the experimental design. In section 3 I briefly describe the results. In section 4 I conclude the report with some discussions on the advantages and disadvantages of the algorithm. Finally in section 5, I attach a snapshot of my implemented code.

II. Experimental Design / Methodology

For this experiment, I implemented the Minimum Error Rate Classifier in python. To calculate the likelihood, I use the normal distribution formula,

$$N_k(x_i|\mu_k, \Sigma_k) = \frac{1}{\sqrt{(2\pi)^D|\Sigma_k|}} e^{-\frac{1}{2}(x_i-\mu_k)^T\Sigma_k^-1(x_i-\mu_k)} \tag{2}$$

A brief description of the algorithm is as follows, I have used the **pandas** library to read the dataset. **Numpy** was used to do basic mathematical operations. I also use **MatPlotLib** to plot the data points and decision boundary. **SciPy** package was used to draw the 3D plot which displays the Probability Distribution Function (PDF). I would like to mention that at decision boundary, for a given data point, posterior probability of both the classes are equal. i.e,

1) $P(\omega_1|x) = P(\omega_2|x)$
2) $ln(\frac{P(x|\omega_1)P(\omega_1)}{P(x)}) = ln(\frac{P(x|\omega_2)P(\omega_2)}{P(x)})$
3) $lnP(x|\omega_1) + lnP(\omega_1) - lnP(x|\omega_2) - lnP(\omega_2)$
4) $-\frac{1}{2}(X^T\Sigma_1^-1X - X^T\Sigma_2^-1 - 2\mu_1^T\Sigma_1^-1X + 2\mu_2^T\Sigma_2^-1X + \mu_1^T\Sigma^-1\mu_1 - \mu_2^T\Sigma^-1\mu_2) - ln|\Sigma_1| + ln|\Sigma_2| + lnP(\omega = 1) - lnP(\omega = 2) = 0$
5) $-\frac{1}{2}(x^T(\Sigma_1^-1-\Sigma_2^-1)X) - 2(\mu_1^T\Sigma_1^-1 - \mu_2^T\Sigma_2^-1)x + w_0 = 0$
6) $X_TW_{12}X - W_{12}^t + W_0 = 0$

is the equation of the decision boundary.

Algorithm 1 Minimum Error Rate Classifier

```
1) Read the train dataset
2) For each data, calculate likelihood using 2
3) Calculate posterior probability for both class $\omega_1$ and $\omega_2$ using 1
4) If $P(\omega_1|x) > P(\omega_2|x)$ data in class 1 else class 2
5) Plot data points using appropriate markers
6) Plot a 3D contour graph showing PDF and Decision Boundary
```

III. Result Analysis

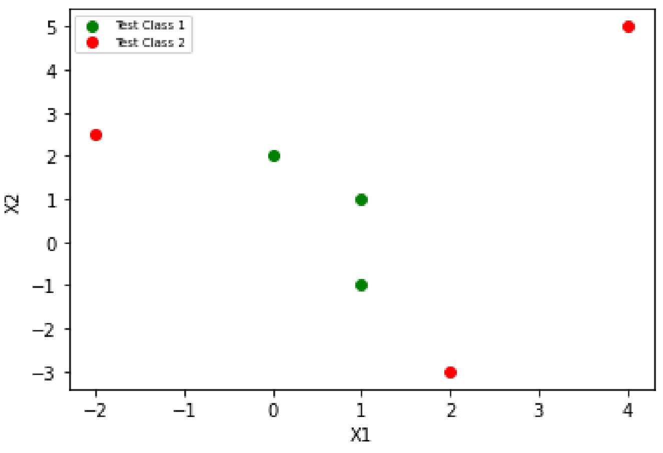

Fig. 1. Distribution of Data Points

After classifying all the data, I first plot them in a 2D graph in 1. It can be seen that, three data fall in class 1 while other three fall in class 2. Next I draw a 3D contour graph that

.

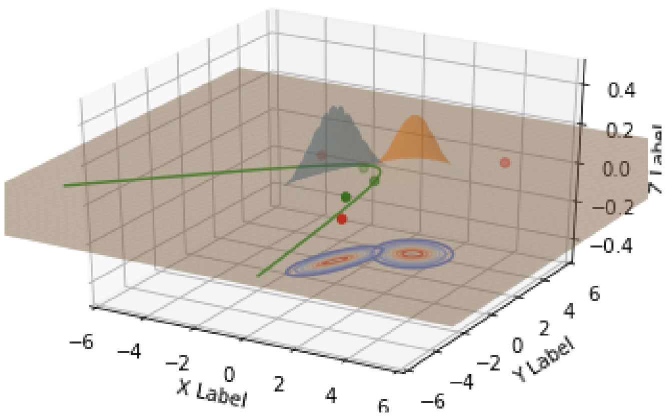

Fig. 2. PDF and its contour

include the corresponding probability distribution function of both the classes in 2.

IV. Conclusion

In this experiment I designed a minimum error rate classifier for binary classification. The dataset I used showed normal distribution. I found that the algorithm is easy to implement and requires less resources for computation. I also found that the algorithm is capable classifying data that are not linearly separable without taking them to a higher dimensional space.

V. ALGORITHM IMPLEMENTATION / CODE

A snapshot of my implementation of the algorithm in python is given below,

```
#declaring constants
miu_1 = np.array([0,0])
miu_2 = np.array([2,2])

sigma_1 = np.array([[0.25,0.3],
                    [0.3,1]])
sigma_2 = np.array([[0.5,0],
                    [0,0.5]])

prior_1 = 0.5
prior_2 = 0.5

class_1 = []
class_2 = []
#predicting class
for x in range(len(X_test)):
    #class 1
    pst_1 = prior_1* 1/math.sqrt((2*math.pi)**2 * np.linalg.det(sigma_1))
        * math.exp(-0.5*np.matmul((miu_1-X_test[x].reshape(1,2)),np.matmul(np.linalg.inv(sigma
        np.transpose(miu_1-X_test[x].reshape(1,2)))))
    pst_2 = prior_2* 1/math.sqrt((2*math.pi)**2 * np.linalg.det(sigma_2))
        * math.exp(-0.5*np.matmul((miu_2-X_test[x].reshape(1,2)),np.matmul(np.linalg.inv(sigma
        np.transpose(miu_2-X_test[x].reshape(1,2)))))
    if(pst_1>pst_2):
        class_1.append(X_test[x])
    else:
        class_2.append(X_test[x])
```