AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Pattern Recognition Lab

Experiment Number 01
"Designing a minimum distance to class mean classifier"

Submitted by-
MD. SHAMIM TOWHID
ID: 12.02.04.097
Section: B2
Year: 4th Semester: 2nd
Date: 11-06-2016

Designing a minimum distance to class mean classifier

Md. Shamim Towhid
Computer Science and Engineering Department
Ahsanullah University of Science and Technology
Dhaka, Bangladesh
shamim.towhid@gmail.com

***Objectives*—the objectives of this experiment is to know how a simple classifier works. The classifier implemented in this experiment may not work correctly in all situation but the purpose to know how a classifier works can be accomplished. Firstly in the introduction section we will discuss the basic things of a classifier and also we will know what includes in our experiment. Then we will go for the implementation of our experiment. We will use MATLAB tools to implement our classifier. After that we will perform a simple result analysis on the result. Then we will conclude our experiment.**

Keywords*—*classifier; pattern recognition; minimum distance to class mean; MATLAB code;

I. Introduction

Minimum distance to class mean is a very basic classifier which used for classifying some data, based on some training data set. It simply calculates the mean of training data set which are already labeled. Then it takes the test data and calculates the distance between itself and each mean of classes. After calculating it can take decision based on decision rule which class the test data should be.

Thus this classifier can divide the whole feature space with some region which relates to a specific class.
In our experiment we are given the following dataset-
ω1 = {(2,2), (3,1), (3,3), (-1,-3), (4,2), (-2,-2)}
ω2 = {(0,0), (-2,2), (-1,-1), (-4,2), (-4,3), (2,6)}

First we will use the above data set to train our classifier then we will test our classifier by the following test points that whether our classifier works fine or not.

X1 = (-1,-1); X2 = (3, 2); X3 = (-2, 1); X4 = (8, 2)

So we can divide our whole task in the following steps-

1. We have to plot all the training data in MATLAB by using plot function.
2. Then there are some test data are also given which are not labeled we have to classify those test data based on our decision rule using the discriminant function given.
3. Finally we have to draw a decision boundary using linear equation.

II. Implementation

A. Plotting the trainng dataset

Two classes in each dataset were given. Firstly we have to plot all the points. Samples from the same class, were plotted using the same color and marker so that different classes can be distinguished easily. Here green dot '.' represent class 1 and red diamond represent class 2.

B. Test the classifier by the given test points

First I calculate the mean of class 1 then plot it with star "*" sign and did the same for class 2. I use different colors to differentiate between them. Then by using the given discriminant function I was able to classify the test data X1, X2, X3 and X4.

A discriminant function simply gives higher value for desirable class. Thus by comparing the function value of each test data I was able to classify the test data set. I plotted the test data set of the same class with same color but different marker. I used triangle sign for test data set.

C. Defining and drawing the decision boundary

To draw the decision boundary we have to draw a line in which every point is equidistant from both the class mean. Thus the discriminant function value on the line for each of the classes should be same. If $g_i(x)$ is the discriminant function value of class 'i' then we can write-

$$g_i(x) - g_j(x) = 0 \text{ ----------------------- (i)}$$

By substituting the value with the discriminant function value we obtain-

$$w_i^T x - 0.5\, w_i^T w_i - w_j^T x + 0.5\, w_j^T w_j = 0$$

$$\Rightarrow \ (w_i^T - w_j^T)\, x - 0.5\,(w_i^T w_i - w_j^T w_j) = 0$$

Where w_i and w_j are the mean of class i and j resepectively. Now this equation is a form of mx+c -------------------(ii) which is an equation of line. By comparing this two it can be written that c= -0.5*det((mean1' * mean1)- (mean2' * mean2)) and m=mean1-mean2. Now the lowest value of x in our given data set is -4 and the highest value is 8. If we take the lowest and highest value of x and then define a range between this two values then we will obtain a set of x values. By putting these values of c in the above equation we get a set of y values. Then after plotting these values of x and y we will obtain a decision boundary which will divide the whole feature space with two region.

III. RESULT ANALYSIS

To analyze the result we will take two dataset and train the classifier. After that we will test the classifier for some points chosen arbitrarily and then we will observe the result.

By plotting each given point and calculating decision boundary, the following graph is generated-

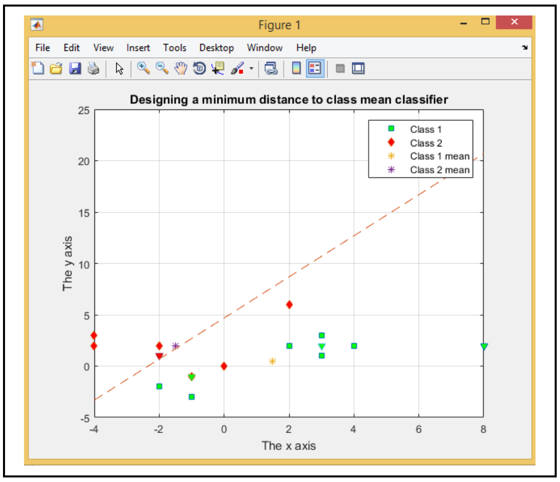

Figure 1: output of the classifier for the given dataset.

Observations:

- All class 1 training data are classified correctly and 2 from class 2 is misclassified.
- All test data are correctly classified.
- Accuracy 81.25%

For another dataset chosen arbitrarily the output is as the following-

class1 = [2 -1;3 0;3 2;-1 -3;4 1;-2 -4;0 -1;-2 2;-1 -4;-4 1];
class2 = [0 0;-2 1;-1 -1;-4 4;-4 1;2 6;2 2;3 1;3 -1;-1 -3];

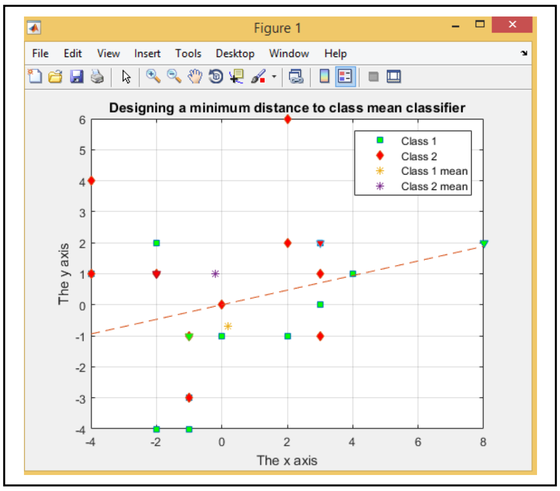

Figure 2: output of the classifier for the chosen dataset.

Observations:

- One out of 10 training data is misclassified of class 1 and 2 out of 10 training data is misclassified of class 2.
- One out of 4 test data is misclassified of class 1 and 1 out of 4 test data is misclassified of class 2.
- Accuracy 79.16%

MATLAB CODE:

```
% Given sets
class1 = [2 2; 3 1; 3 3; -1 -3; 4 2; -2 -2];
class2 = [0 0; -2 2; -1 -1; -4 2; -4 3; 2 6];

% Plotting all samples in a graph
plot(class1(:,1),class1(:,2),'s','MarkerFaceColor','g');
hold on;
plot(class2(:,1),class2(:,2),'d','MarkerFaceColor','r');
 % Calculating means of class 1 and 2

mean1 = [ mean(class1(:,1)) mean(class1(:,2))];
plot(mean1(1),mean1(2),'*','MarkerFaceColor','g');
mean2 = [ mean(class2(:,1)) mean(class2(:,2))];
plot(mean2(1),mean2(2),'*','MarkerFaceColor','r');
% Sample points
x = [-1 -1; 3 2; -2 1; 8 2];
% Plotting the sample points


for n = 1:4
 %dist1=sqrt( ( mean1(1)-x(n) )^2 + ( mean1(2)-x(n+4) )^2 );
 %dist2=sqrt( ( mean2(1)-x(n) )^2 + ( mean2(2)-x(n+4) )^2 );

 %if dist1 < dist2
  % plot(x(n,1),x(n,2),'v','MarkerFaceColor','g');
 %else
  % plot(x(n,1),x(n,2),'v','MarkerFaceColor','r');
 %end

 val1 = x(n,:) * mean1' - .5 * (mean1 * mean1');
 val2 = x(n,:) * mean2' - .5 * (mean2 * mean2');
```

```
 if val1 > val2

plot(x(n,1),x(n,2),'v','MarkerFaceColor',
'g');
 else

plot(x(n,1),x(n,2),'v','MarkerFaceColor',
'r');
 end
end

% Calculating decision boundary between
two classes

coeff=mean1-mean2;
const= -0.5*det((mean1' * mean1)- (mean2'
* mean2));
xvalue = -4:8;
for i=1:length(xvalue)
    yvalue(:,i) = -((coeff(1,1)*
xvalue(:,i)+const)/coeff(1,2));
end
final_xy_matrix=[xvalue',yvalue'];

plot(final_xy_matrix(:,1),final_xy_matrix
(:,2),'--','MarkerFaceColor','y');

legend('Class 1','Class 2','Class 1
mean','Class 2 mean');

title('Designing a minimum distance to
class mean classifier');
grid;
ylabel('The y axis');
xlabel('The x axis');
hold off;
```

IV. Conclusion

As this is a very simple classifier I tried to keep it simple in the code also. As the data set will grow this classifier however won't give good result because this is done by using linear equation. The error rate will be high. Still this is a good start and to understand the basic work principle of a classifier this is a great experiment. This also helps to become familiar with MATLAB.