Course No.: **ELEN-857**
Course Title: **Advanced Pattern Recognition Method**
Department: **Electrical and Computer Engineering**

Project 3: Statistical Classifier

Submitted To:
Dr. Robert Y.Li, Professor
Department of Electrical Engineering
Telephone: (336) 285-3716; E-mail: eeli@ncat.edu

Prepared By:
Name: Mrinmoy Sarkar
Banner Id: 950-363-260
E-mail: msarkar@aggies.ncat.edu

Contents:

1. Abstract:

The main purpose of the project is to apply minimum-distance classifier and quadratic Bayesian classifier to Fisher's Iris data. Fisher's Iris data contains a set of measurements related to 3 species of the Iris plant. The three species are Iris Setosa, Iris Versicolor, and Iris Virginica.

2. Technical Description:

The dataset contains 50 plants from each of the 3 species. There are 4 features in the dataset named sepal length, sepal width, petal length and petal width. MATLAB programming language is used to implement the minimum-distance classifier and quadratic Bayesian classifier. Two different training set (10, 25) and two different test set (40, 25) are used to verify the algorithm and corresponding confusion matrix is calculated.

3. Mathematical Formulation:

For minimum-distance classifier the decision function is given as below:

$$d_i(X) = X'm_i - \frac{1}{2}m_i'm_i\,, \quad i = 1,2,\dots,M$$

For quadratic Bayesian classifier the decision function is given as below:

$$d_i(X) = \ln\big(p(w_i)\big) - \frac{1}{2}\ln|C_i| - \frac{1}{2}[(X - m_i)'C_i^{-1}(X - m_i)]\,, \quad i = 1,2,\dots,M$$

4. Results:

minimum-distance classifier for no. of training sample: 10 and no. of test sample: 40

mean vector:

4.8600	3.3100	1.4500	0.2200
6.1000	2.8700	4.3700	1.3800
6.5700	2.9400	5.7700	2.0400

Confusion Matrix:

	irisSetosa(True)	irisVersicolor(True)	irisVerginica(True)
irisSetosa(Predicted)	40	0	0
irisVersicolor(Predicted)	0	39	11

irisVerginica(Predicted)	0	1	29

Correct = 90.00%

minimum-distance classifier for no. of training sample: 25 and no. of test sample: 25

mean vector:

```
5.0280    3.4800    1.4600    0.2480
6.0120    2.7760    4.3120    1.3440
6.5760    2.9280    5.6400    2.0440
```

Confusion Matrix:

	irisSetosa(True)	irisVersicolor(True)	irisVerginica(True)
irisSetosa(Predicted)	25	0	0
irisVersicolor(Predicted)	0	24	3
irisVerginica(Predicted)	0	1	22

Correct = 94.67%

quadratic Bayesian classifier for no. of training sample: 10 and no. of test sample: 40

mean vector:

```
4.8600    3.3100    1.4500    0.2200
6.1000    2.8700    4.3700    1.3800
6.5700    2.9400    5.7700    2.0400
```

Co-variance Matrix:

CV1 =

```
0.0849    0.0704    0.0189    0.0087
0.0704    0.0943    0.0172    0.0164
0.0189    0.0172    0.0117    0.0044
0.0087    0.0164    0.0044    0.0062
```

CV2 =

0.5289	0.1933	0.3233	0.0800
0.1933	0.1157	0.1323	0.0427
0.3233	0.1323	0.2379	0.0649
0.0800	0.0427	0.0649	0.0284

CV3 =

0.6468	0.1391	0.4601	0.0869
0.1391	0.1138	0.1169	0.0882
0.4601	0.1169	0.3601	0.0836
0.0869	0.0882	0.0836	0.0849

Confusion Matrix:

	irisSetosa(True)	irisVersicolor(True)	irisVerginica(True)
irisSetosa(Predicted)	40	0	0
irisVersicolor(Predicted)	0	39	5
irisVerginica(Predicted)	0	1	35

Correct = 95.00%

quadratic Bayesian classifier for no. of training sample: 25 and no. of test sample: 25

mean vector:

5.0280	3.4800	1.4600	0.2480
6.0120	2.7760	4.3120	1.3440
6.5760	2.9280	5.6400	2.0440

Co-variance Matrix:

CV1 =

0.1604	0.1181	0.0241	0.0194
0.1181	0.1358	0.0062	0.0223
0.0241	0.0062	0.0392	0.0066
0.0194	0.0223	0.0066	0.0109

CV2 =

0.3003	0.1095	0.1865	0.0520
0.1095	0.1244	0.0886	0.0465
0.1865	0.0886	0.1969	0.0640
0.0520	0.0465	0.0640	0.0426

CV3 =

0.5244	0.1215	0.4323	0.0619
0.1215	0.1304	0.0988	0.0604
0.4323	0.0988	0.4175	0.0673
0.0619	0.0604	0.0673	0.0651

Confusion Matrix:

	irisSetosa(True)	irisVersicolor(True)	irisVerginica(True)
irisSetosa(Predicted)	25	0	0
irisVersicolor(Predicted)	0	24	1
irisVerginica(Predicted)	0	1	24

Correct = 97.33%

5. Summary:

- The correct results using the Bayesian quadratic were more than the correct samples using Minimum distance function with the same training samples. Bayesian decision function gives less error and more efficient.

- When the number of training samples increases, using Bayesian function or Minimum distance function, the percentage of the correct samples increases for the same function. Therefore, it is better to use more training samples for higher percentage of correct samples.

6. Appendix:

MATLAB Code:

```
%% file name project3.m
% author: Mrinmoy Sarkar
```

```
% email: msarkar@aggies.ncat.edu
% date: 10/24/2017


clear;
close all;

% load data to a veriable
data = importdata('iris.txt');

% no. of class is 3 named Iris-setosa, Iris-versicolor and Iris-
verginica
% there are 4 attributes named sepal-length, sepal-width, petal-
length,
% petal-width
% there are 50 plants for each species

irisSetosa = zeros(50,4);
irisVersicolor = zeros(50,4);
irisVerginica = zeros(50,4);

n = size(data,1);

indxSeto = 1;
indxVers = 1;
indxVerg = 1;

for i=2:n
    x = strsplit(cell2mat(data(i)));
    if strcmp(x(5), 'Iris-setosa')
        for j=1:4
            irisSetosa(indxSeto,j) = str2double(cell2mat(x(j)));
        end
        indxSeto = indxSeto + 1;
    elseif strcmp(x(5), 'Iris-versicolor')
        for j=1:4
            irisVersicolor(indxVers,j) =
str2double(cell2mat(x(j)));
        end
        indxVers = indxVers + 1;
    elseif strcmp(x(5), 'Iris-virginica')
        for j=1:4
            irisVerginica(indxVerg,j) =
str2double(cell2mat(x(j)));
        end
        indxVerg = indxVerg + 1;
    end
```

```
end

%% minimum-distance classifier
trs = [10,25];
tns = [40,25];
for l=1:2
    no_of_train_sample = trs(l);
    no_of_test_sample = tns(l);
    fprintf('minimum-distance classifier for no. of training
sample : %d and no. of test sample : %d\n',no_of_train_sample,
no_of_test_sample);
    % dataset is partisioned as train:test =
no_of_train_sample:no_of_test_sample
    trainSet = [irisSetosa(1:no_of_train_sample,:);
irisVersicolor(1:no_of_train_sample,:);
irisVerginica(1:no_of_train_sample,:)];
    testSet = [irisSetosa(no_of_train_sample+1:50,:);
irisVersicolor(no_of_train_sample+1:50,:);
irisVerginica(no_of_train_sample+1:50,:)];

    % mean vector calculation
    m_i =
[mean(trainSet(1:no_of_train_sample,:));mean(trainSet(no_of_trai
n_sample+1:2*no_of_train_sample,:));mean(trainSet(2*no_of_train_
sample+1:3*no_of_train_sample,:))];
    disp('mean vector:');
    disp(m_i)
    mi_das_mi = 0.5 * diag(m_i*m_i');
    confussionMat = zeros(3,3);
    % test each sample
    for i = 1:no_of_test_sample*3
        testX = testSet(i,:);
        di = testX*m_i' - mi_das_mi';
        [m, m_index] = max(di);
        if i<=no_of_test_sample
            confussionMat(m_index, 1) = confussionMat(m_index,
1) + 1;
        elseif i>no_of_test_sample && i<=2*no_of_test_sample
            confussionMat(m_index, 2) = confussionMat(m_index,
2) + 1;
        else
            confussionMat(m_index, 3) = confussionMat(m_index,
3) + 1;
        end
    end
```

```
    fprintf('\nConfusion Matrix:        |irisSetosa(True) |
irisVersicolor(True) | irisVerginica(True)\n');
    fprintf('---------------------------------------------------
-------------------------------------\n');
    fprintf('irisSetosa(Predicted)     | %2d              | %2d
| %2d\n', confussionMat(1,1), confussionMat(1,2),
confussionMat(1,3));
    fprintf('---------------------------------------------------
-------------------------------------\n');
    fprintf('irisVersicolor(Predicted) | %2d              | %2d
| %2d\n', confussionMat(2,1), confussionMat(2,2),
confussionMat(2,3));
    fprintf('---------------------------------------------------
-------------------------------------\n');
    fprintf('irisVerginica(Predicted)  | %2d              | %2d
| %2d\n', confussionMat(3,1), confussionMat(3,2),
confussionMat(3,3));
    fprintf('---------------------------------------------------
-------------------------------------\n');
    fprintf('Correct =
%.2f%%\n\n\n',100*sum(diag(confussionMat))/(3*no_of_test_sample)
);
end

%% quadratic Bayesian classifier
trs = [10,25];
tns = [40,25];
for l=1:2
    no_of_train_sample = trs(l);
    no_of_test_sample = tns(l);
     fprintf('quadratic Bayesian classifier for no. of training
sample : %d and no. of test sample : %d\n',no_of_train_sample,
no_of_test_sample);
    % dataset is partisioned as train:test =
no_of_train_sample:no_of_test_sample
    trainSet = [irisSetosa(1:no_of_train_sample,:);
irisVersicolor(1:no_of_train_sample,:);
irisVerginica(1:no_of_train_sample,:)];
    testSet = [irisSetosa(no_of_train_sample+1:50,:);
irisVersicolor(no_of_train_sample+1:50,:);
irisVerginica(no_of_train_sample+1:50,:)];

    % mean vector calculation
    m_i =
[mean(trainSet(1:no_of_train_sample,:));mean(trainSet(no_of_trai
n_sample+1:2*no_of_train_sample,:));mean(trainSet(2*no_of_train_
sample+1:3*no_of_train_sample,:))];
```

```
    disp('mean vector:');
    disp(m_i)
    % co-varience matrix calculation
    cv =
[cov(trainSet(1:no_of_train_sample,:));cov(trainSet(no_of_train_
sample+1:2*no_of_train_sample,:));cov(trainSet(2*no_of_train_sam
ple+1:3*no_of_train_sample,:))];
    disp('Co-variance Matrix:');
    disp('CV1 = ');
    disp(cv(1:4,:));
    disp('CV2 = ');
    disp(cv(5:8,:));
    disp('CV3 = ');
    disp(cv(9:12,:));

    confussionMat = zeros(3,3);
    % test each sample
    for i = 1:no_of_test_sample*3
        testX = testSet(i,:);
        ln_ci = [log(det(cv(1:4,:))) log(det(cv(5:8,:)))
log(det(cv(9:12,:)))];
        ln_ci = -0.5 * ln_ci;
        xm_i = [testX;testX;testX] - m_i;
        xm_i_das_Cinv_xm_i =
[xm_i(1,:)*inv(cv(1:4,:))*(xm_i(1,:))'
xm_i(2,:)*inv(cv(5:8,:))*(xm_i(2,:))'
xm_i(3,:)*inv(cv(9:12,:))*(xm_i(3,:))'];
        xm_i_das_Cinv_xm_i = -0.5 * xm_i_das_Cinv_xm_i;
        di = ln_ci + xm_i_das_Cinv_xm_i;
        [m, m_index] = max(di);
        if i<=no_of_test_sample
            confussionMat(m_index, 1) = confussionMat(m_index,
1) + 1;
        elseif i>no_of_test_sample && i<=2*no_of_test_sample
            confussionMat(m_index, 2) = confussionMat(m_index,
2) + 1;
        else
            confussionMat(m_index, 3) = confussionMat(m_index,
3) + 1;
        end
    end

    fprintf('\nConfusion Matrix:        |irisSetosa(True) |
irisVersicolor(True) | irisVerginica(True)\n');
    fprintf('----------------------------------------------------
-------------------------------------\n');
```

```
    fprintf('irisSetosa(Predicted)     | %2d                  | %2d
| %2d\n', confussionMat(1,1), confussionMat(1,2),
confussionMat(1,3));
    fprintf('-----------------------------------------------------
-------------------------------------\n');
    fprintf('irisVersicolor(Predicted) | %2d                  | %2d
| %2d\n', confussionMat(2,1), confussionMat(2,2),
confussionMat(2,3));
    fprintf('-----------------------------------------------------
-------------------------------------\n');
    fprintf('irisVerginica(Predicted)  | %2d                  | %2d
| %2d\n', confussionMat(3,1), confussionMat(3,2),
confussionMat(3,3));
    fprintf('-----------------------------------------------------
-------------------------------------\n');
    fprintf('Correct =
%.2f%%\n\n\n',100*sum(diag(confussionMat))/(3*no_of_test_sample)
);
end
```