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

Project 3: Feature Selection

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 Feature Selection technique 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 and 4 features names Sepal Length, Sepal Width, Petal Length, Petal Width.

2. Technical Description:

Divergence is a measure of separability between two classes and it is defined as the difference between expected values of the likelihood ratio for the two classes under consideration. For better separability between two classes, divergence must be high.

For the computation of divergence, transformed divergence, or the Bhattacharyya distance and the probability of error for two or three feature(s) out of four taken at a time the 150 observations for the three IRIS data classes are used.

3. Mathematical Formulation:

For normal distribution, the divergence is estimated as

$$D = 0.5 * tr[(c_1 - c_2)(c_2^{-1} - c_1^{-1})] + 0.5 * tr[(c_1^{-1} - c_2^{-1})(m_1 - m_2)(m_1 - m_2)']$$

Where, c_1 and c_2 are covariance matrices for the two classes and m_1 and m_2 are the mean vectors.

The average divergence for three classes is:

$$D_{av} = \frac{D_{12} + D_{13} + D_{23}}{3}$$

Transformed divergence is expressed as

$$D_T = 2 * \left(1 - exp(-D/8)\right)$$

Where, D is the divergence.

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

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

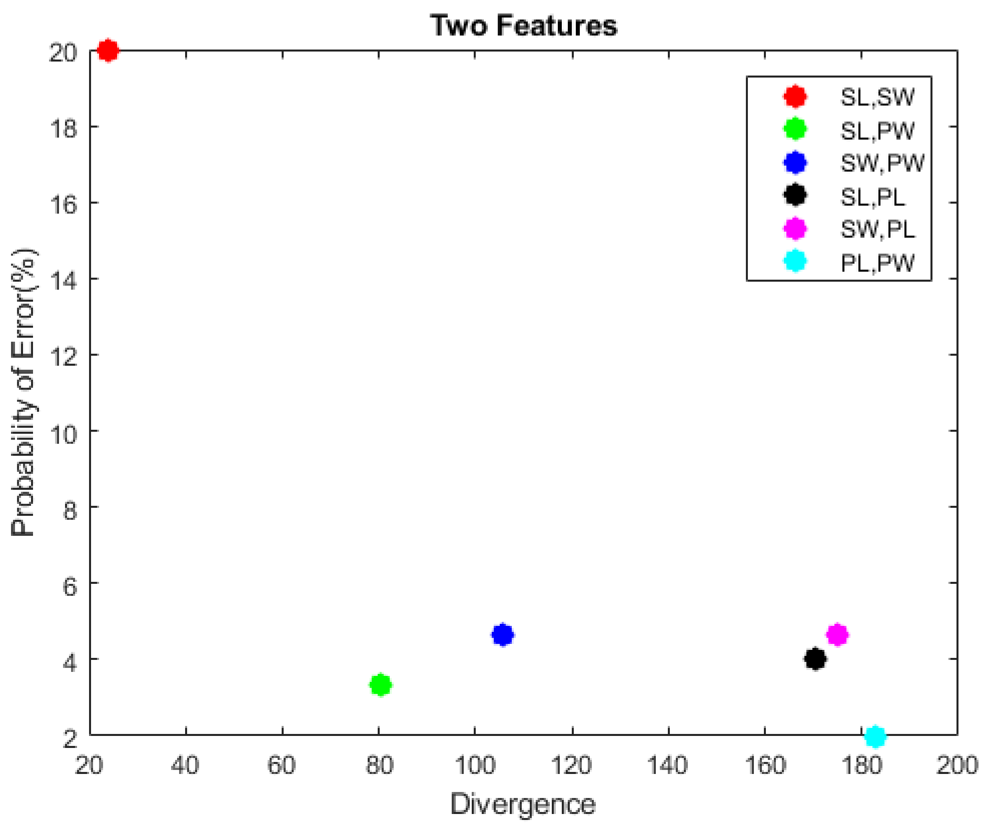
For quadratic Bayesian classifier the decision function is given as below:

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

4. Results:

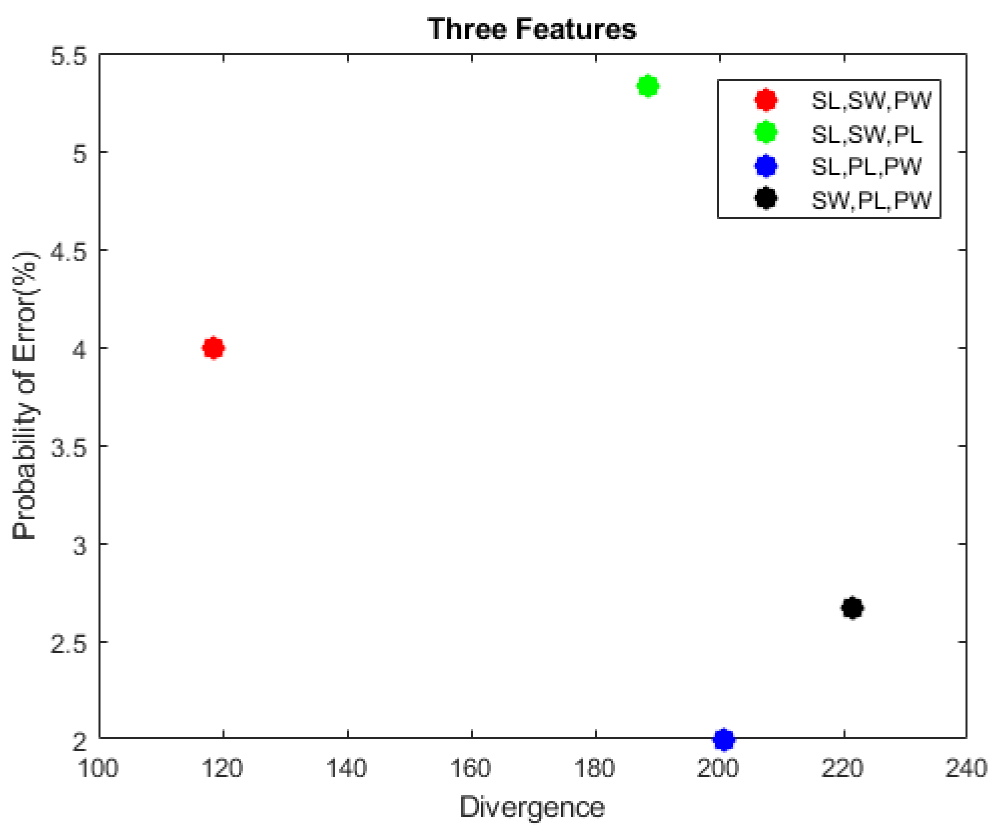
Two features using divergence measure:

Feature Combination	SL,SW	SL,PW	SW,PW	SL,PL	SW,PL	PL,PW
Divergence	23.8078	80.3513	105.6774	170.2532	174.9820	182.7527
Probability of Error(%)	20.0000	3.3333	4.6667	4.0000	4.6667	2.0000



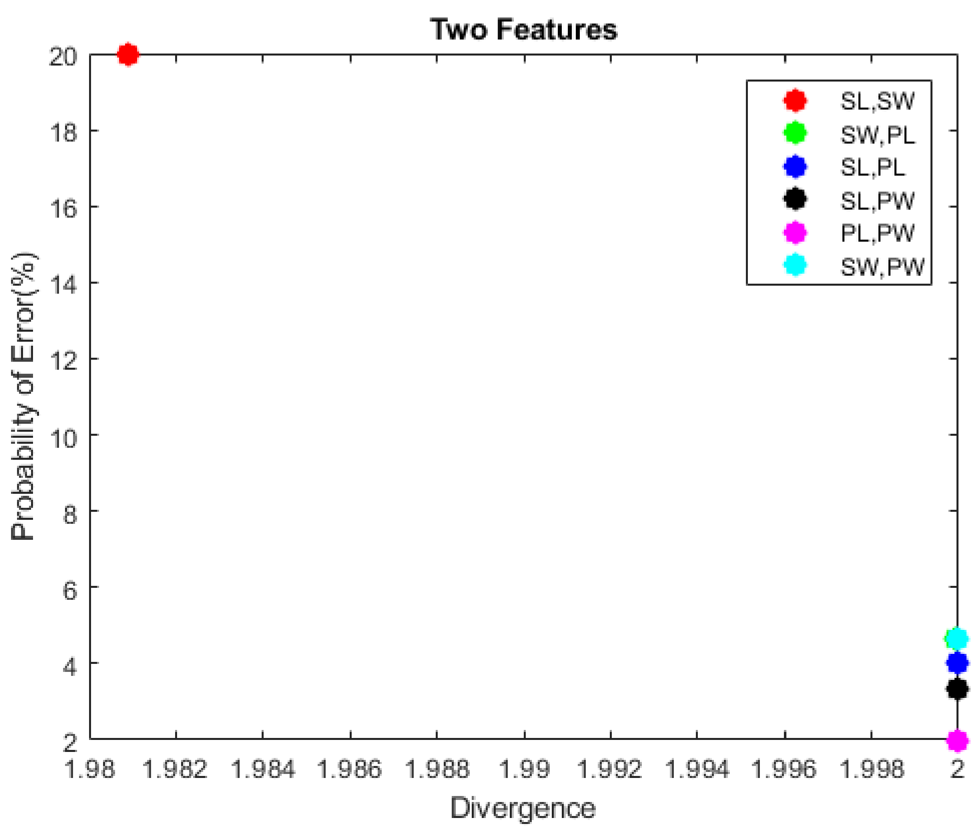
Three features using divergence measure:

Feature Combination	SL,SW,PW	SL,SW,PL	SL,PL,PW	SW,PL,PW
Divergence	118.5123	188.6182	200.7230	221.3827
Probability of Error(%)	4.0000	5.3333	2.0000	2.6667



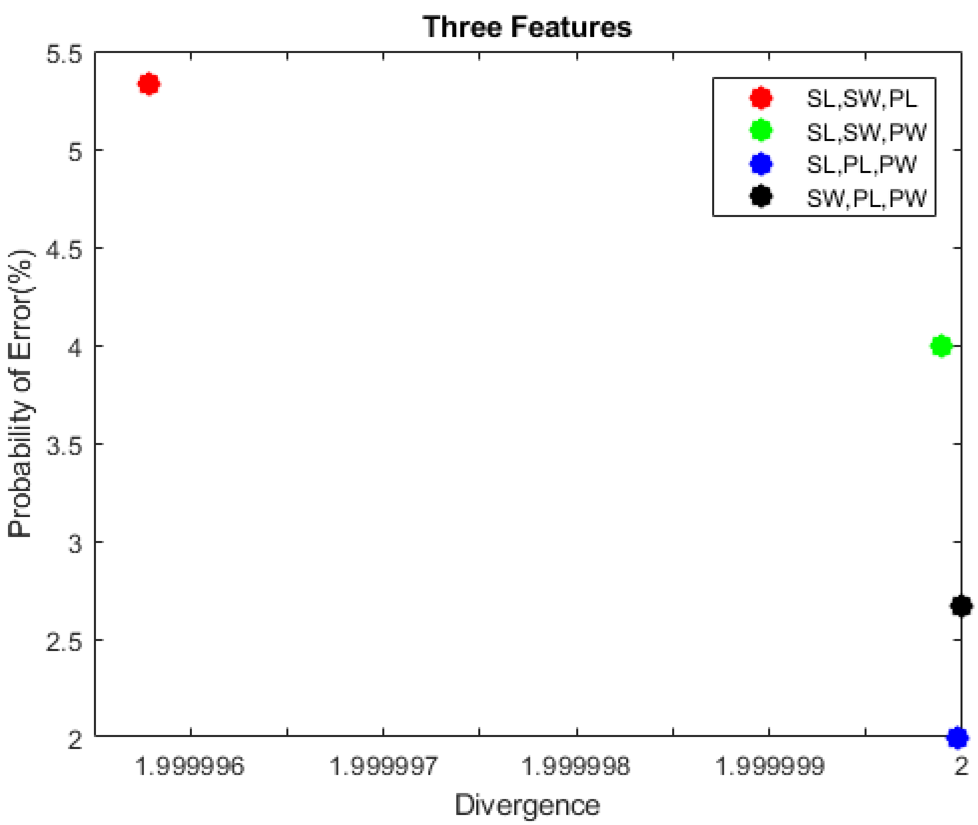
Two features using transformed divergence measure:

Feature Combination	SL,SW	SW,PL	SL,PL	SL,PW	PL,PW	SW,PW
Transformed Divergence	1.9809	1.9999	2.0000	2.0000	2.0000	2.0000
Probability of Error(%)	20.0000	4.6667	4.0000	3.3333	2.0000	4.6667



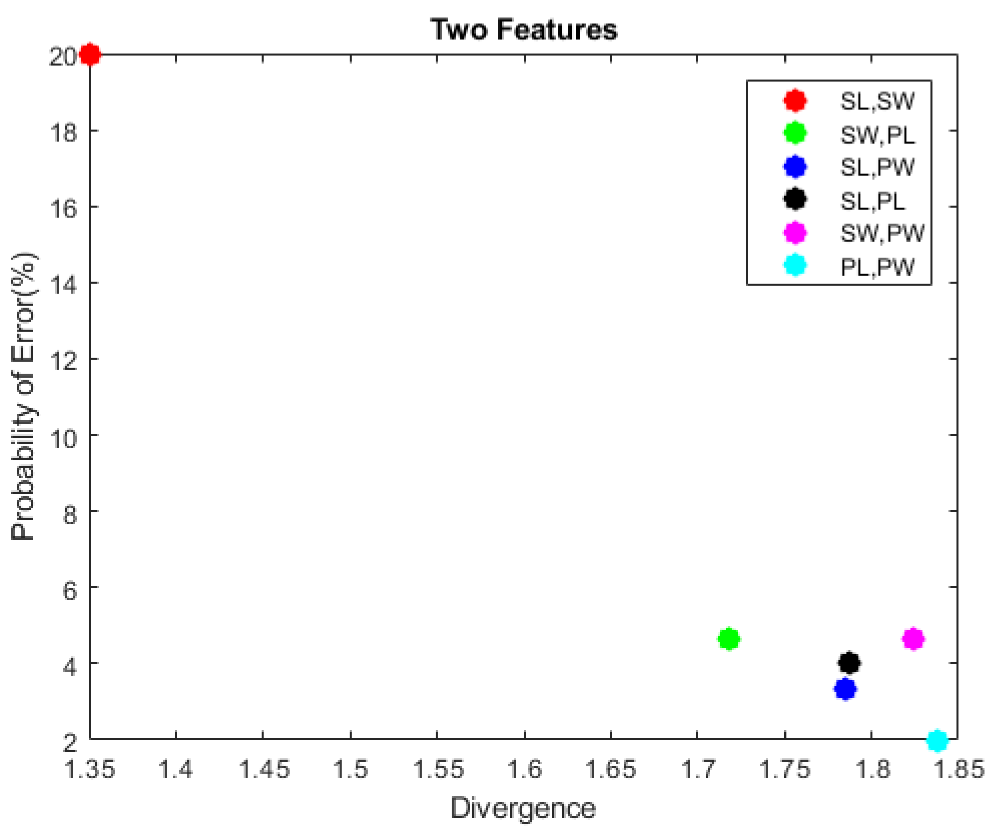
Three features using transformed divergence measure:

Feature Combination	SL,SW,PL	SL,SW,PW	SL,PL,PW	SW,PL,PW
Transformed Divergence	2.0000	2.0000	2.0000	2.0000
Probability of Error(%)	5.3333	4.0000	2.0000	2.6667



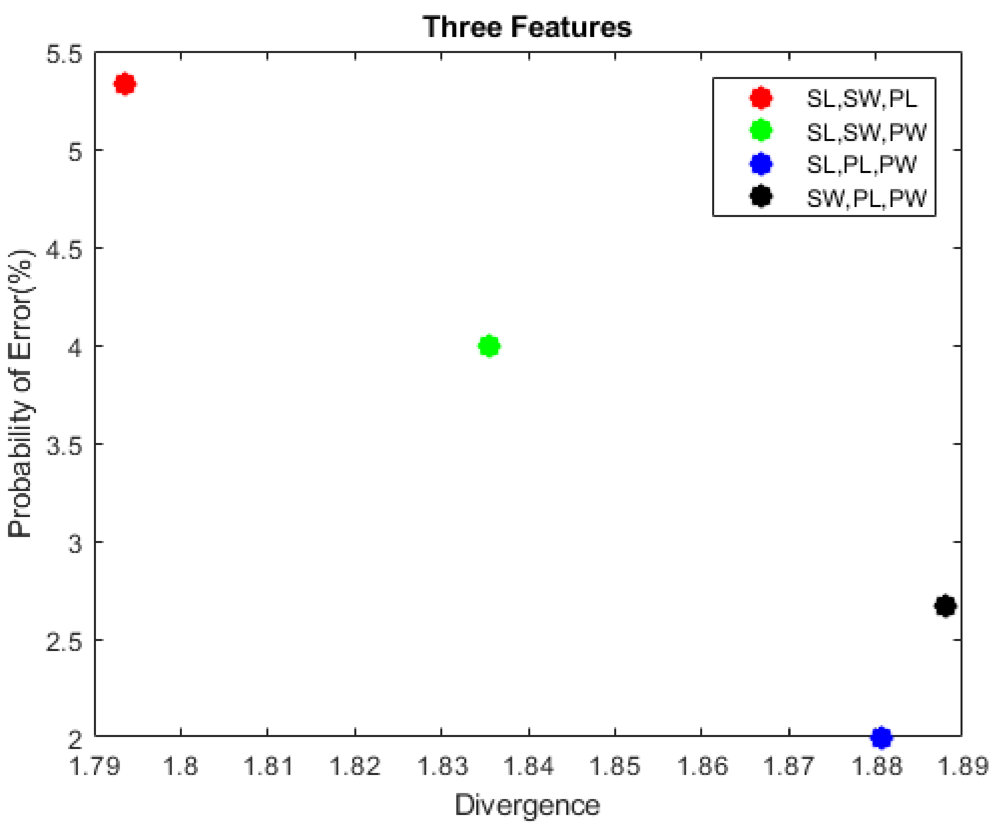
Two features using Bhattacharyya distance measure:

Feature Combination	SL,SW	SW,PL	SL,PW	SL,PL	SW,PW	PL,PW
Bhattacharyya distance	1.3509	1.7183	1.7850	1.7876	1.8243	1.8384
Probability of Error(%)	20.0000	4.6667	3.3333	4.0000	4.6667	2.0000



Three features using Bhattacharyya distance measure:

Feature Combination	SL,SW,PL	SL,SW,PW	SL,PL,PW	SW,PL,PW
Bhattacharyya distance	1.7937	1.8356	1.8806	1.8881
Probability of Error(%)	5.3333	4.0000	2.0000	2.6667



5. Summary:

- Quadratic classifier performs better when using three features than two features.
- The transformed divergence performs so close to the divergence, it slightly has better analysis advantage.
- Bhattacharyya distance gives an analysis close to that of transformed divergence.
- Feature selection using divergence analysis is a useful technique for making better and well-informed feature selection.

6. Appendix:

MATLAB Code:

```
%% file name project4.m
% author: Mrinmoy Sarkar
% email: msarkar@aggies.ncat.edu
% date: 12/1/2017

%%
clear all;
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

irisSetosad = zeros(50,4);
irisVersicolord = zeros(50,4);
irisVerginicad = 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
            irisSetosad(indxSeto,j) =
str2double(cell2mat(x(j)));
        end
        indxSeto = indxSeto + 1;
    elseif strcmp(x(5), 'Iris-versicolor')
        for j=1:4
            irisVersicolord(indxVers,j) =
str2double(cell2mat(x(j)));
        end
        indxVers = indxVers + 1;
    elseif strcmp(x(5), 'Iris-virginica')
        for j=1:4
            irisVerginicad(indxVerg,j) =
str2double(cell2mat(x(j)));
        end
        indxVerg = indxVerg + 1;
    end
end

featureName = {'SL','SW','PL','PW'};
color = 'rgbkmc';

%% divergence calculation and quadratic Bayesian
classifier implementation
for opt = 1:3
    option = opt;
    for sf=2:3
        noOfclass = 3;
        classes={};
        classes{1} = irisSetosad;
        classes{2} = irisVersicolord;
        classes{3} = irisVerginicad;
```

```
        noOfselectedFeature = sf;
        featurevector = 1:4;
        combOffeature =
nchoosek(featurevector,noOfselectedFeature);
        combOfclass = nchoosek(1:3,2);
        Di = zeros(size(combOffeature,1),1);
        Pe = Di;
        for ii=1:size(combOffeature,1)
            mu = {};
            co = {};
            for j=1:noOfclass
                cl = classes{j};
                mu{j} =
(mean(cl(:,combOffeature(ii,:))))';
                co{j} =
(cov(cl(:,combOffeature(ii,:))));
            end
            D = zeros(noOfclass,1);
            a = D;
            for j=1:size(combOfclass,1)
                D(j) =
0.5*trace((co{combOfclass(j,1)}-
co{combOfclass(j,2)})*(pinv(co{combOfclass(j,2)})-
pinv(co{combOfclass(j,1)})))...
                    +
0.5*trace((pinv(co{combOfclass(j,1)})+pinv(co{combOfcl
ass(j,2)}))*(mu{combOfclass(j,1)}-
mu{combOfclass(j,2)})...
                    *(mu{combOfclass(j,1)} -
mu{combOfclass(j,2)})');

                A =
0.5*(co{combOfclass(j,1)}+co{combOfclass(j,2)});
                a(j) = 0.125*(mu{combOfclass(j,1)} -
mu{combOfclass(j,2)})'...
                    *pinv(A)*(mu{combOfclass(j,1)} -
mu{combOfclass(j,2)})...

+0.5*log(det(A)/sqrt(det(co{combOfclass(j,1)})*det(co{
combOfclass(j,2)})));
```

```
            end
            if option == 1 %divergence
                Di(ii) = mean(D);
            elseif option == 2 %Transformed divergence
                Di(ii) = mean(2*(1-exp(-D)/8));
            elseif option == 3 %Bhattacharyya distance
                Di(ii) = mean(2*(1-exp(-a)));
            end




            % quadratic Bayesian classifier
            irisSetosa =
irisSetosad(:,combOffeature(ii,:));
            irisVersicolor =
irisVersicolord(:,combOffeature(ii,:));
            irisVerginica =
irisVerginicad(:,combOffeature(ii,:));

            no_of_train_sample = 50;
            no_of_test_sample = 50;
            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(1:no_of_test_sample,:);
irisVersicolor(1:no_of_test_sample,:);
irisVerginica(1:no_of_test_sample,:)];

            % mean vector calculation
            m_i =
[mean(trainSet(1:no_of_train_sample,:));mean(trainSet(
no_of_train_sample+1:2*no_of_train_sample,:));mean(tra
```

```
inSet(2*no_of_train_sample+1:3*no_of_train_sample,:))]
;

            % 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(trainS
et(2*no_of_train_sample+1:3*no_of_train_sample,:))];

            confussionMat = zeros(3,3);
            % test each sample
            for i = 1:no_of_test_sample*3
                testX = testSet(i,:);
                ln_ci =
[log(det(cv(1:noOfselectedFeature,:)))
log(det(cv(noOfselectedFeature+1:2*noOfselectedFeature
,:)))
log(det(cv(2*noOfselectedFeature+1:3*noOfselectedFeatu
re,:)))];
                ln_ci = -0.5 * ln_ci;
                xm_i = [testX;testX;testX] - m_i;
                xm_i_das_Cinv_xm_i =
[xm_i(1,:)*pinv(cv(1:noOfselectedFeature,:))*(xm_i(1,:
))'
xm_i(2,:)*pinv(cv(noOfselectedFeature+1:2*noOfselected
Feature,:))*(xm_i(2,:))'
xm_i(3,:)*pinv(cv(2*noOfselectedFeature+1:3*noOfselect
edFeature,:))*(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_te
st_sample));
            Pe(ii) = 100 -
100*sum(diag(confussionMat))/(3*no_of_test_sample);
        end

        % sort divergence
        [Di,indx] = sort(Di);
        tpe = Pe;
        tcombOffeature = combOffeature;
        for jj=1:length(Di)
```

```
            Pe(jj) = tpe(indx(jj));
            combOffeature(jj,:) =
tcombOffeature(indx(jj),:);
        end
        figure
        for jj=1:length(Di)
            if length(Di)==6
                lgnd =
strcat(featureName{combOffeature(jj,1)},',',featureNam
e{combOffeature(jj,2)});
            else
                lgnd =
strcat(featureName{combOffeature(jj,1)},',',featureNam
e{combOffeature(jj,2)},',',featureName{combOffeature(j
j,3)});
            end
            disp(lgnd)

plot(Di(jj),Pe(jj),strcat('*',color(jj)),'LineWidth',7
,'DisplayName', lgnd);
            legend('-DynamicLegend');
            hold on;
        end
        if length(Di)==6
            title('Two Features');
        else
            title('Three Features');
        end
        xlabel('Divergence');
        ylabel('Probability of Error(%)');
        disp('Divergence:')
        disp(Di')
        disp('Probability of Error(%):')
        disp(Pe')
    end
end
```