EECE-5644
INTRO TO MACHINE LEARNING AND PATTERN RECOGNITION
ASSIGNMENT 1
SAI PRASASTH KOUNDINYA GANDRAKOTA

NUID: 002772719

Question 1

Code (MATLAB)

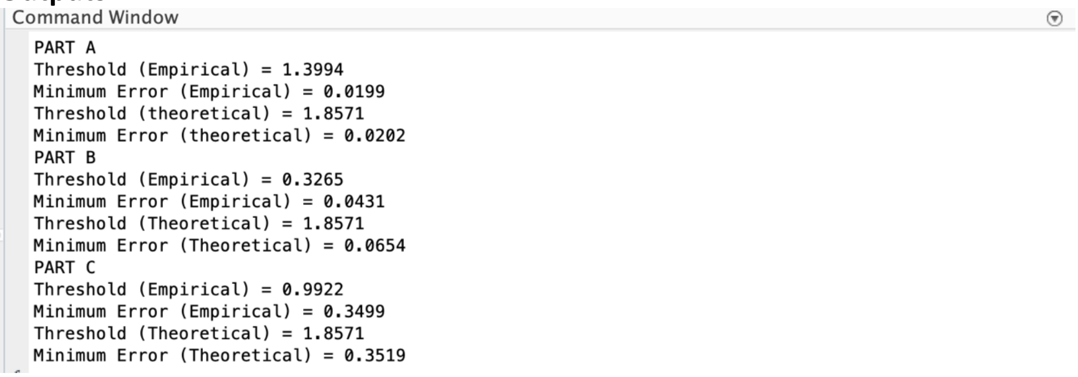
```
clear all; close all; clc


%
rng(10);

% INPUT PARAMETERS
mean1 = [-0.5 -0.5 -0.5 -0.5];
SD1 = (1/4)*[2 -0.5 0.3 0;-0.5 1 -0.5 0;0.3 -0.5 1 0;0 0 0 2];
mean2 = [1 1 1 1];
SD2 = [1 0.3 -0.2 0;0.3 2 0.3 0;-0.2 0.3 1 0;0 0 0 3];
priori = [0.35, 0.65];
n = 4; %4D Gaussian
sample_size = 10000;



%PART A

%Class Label Generation
[C_L,smple] = input_and_class_labels(mean1,mean2,SD1,SD2,sample_size, priori,n);

%Input Data Graphical Plot
plot_input_data(C_L,smple);

%Discriminant Score, Threshold Calculation
[DS,sorted_DS,threshold] = calc_threshold(mean1,mean2,SD1,SD2,smple);

%Theoretical Threshold Calculation
theoretical_threshold = log(priori(2) / priori(1));
theoretical_decision = DS >= theoretical_threshold;
theoretical_true_p = numel(find((theoretical_decision==1) & (C_L==1))) /
numel(find(C_L==1));
theoretical_false_p = numel(find((theoretical_decision==1) & (C_L==0))) /
numel(find(C_L==0));
theoretical_error = priori(2)*theoretical_false_p + priori(1)*(1-theoretical_true_p);

%Data Classification
for i = 1:length(threshold)
    decision = DS >= threshold(i);
    true_p(i) = numel(find((decision==1) & (C_L==1))) / numel(find(C_L==1));
    false_p(i) = numel(find((decision==1) & (C_L==0))) / numel(find(C_L==0));
    error(i) = priori(2)*false_p(i) + priori(1)*(1-true_p(i));
end
```

```
%ROC Curve Graphical Plot
plot_ROC_Curve(true_p,false_p,error);

%OUTPUT DATA
disp('PART A');
fprintf('Threshold (Empirical) = %.4f\n', exp(threshold(find(error==min(error)))))
fprintf('Minimum Error (Empirical) = %.4f\n', min(error))
fprintf('Threshold (theoretical) = %.4f\n', exp(theoretical_threshold))
fprintf('Minimum Error (theoretical) = %.4f\n', theoretical_error)


%PART B

%Class Label Generation
[C_L,smple] = input_and_class_labels(mean1,mean2,SD1,SD2,sample_size, priori,n);

%Input Data Graphical Plot
plot_input_data(C_L,smple);

%Discriminant Score, Threshold Calculation
[DS,sorted_DS,threshold] = calc_threshold(mean1,mean2,eye(n,n),eye(n,n),smple);

%Theoretical Threshold Calculation
theoretical_threshold = log(priori(2) / priori(1));
theoretical_decision = DS >= theoretical_threshold;
theoretical_true_p = numel(find((theoretical_decision==1) & (C_L==1))) /
numel(find(C_L==1));
theoretical_false_p = numel(find((theoretical_decision==1) & (C_L==0))) /
numel(find(C_L==0));
theoretical_error = priori(2)*theoretical_false_p + priori(1)*(1-theoretical_true_p);

%Data Classification
for i = 1:length(threshold)
    decision = DS >= threshold(i);
    true_p(i) = numel(find((decision==1) & (C_L==1))) / numel(find(C_L==1));
    false_p(i) = numel(find((decision==1) & (C_L==0))) / numel(find(C_L==0));
    error(i) = priori(2)*false_p(i) + priori(1)*(1-true_p(i));
end

%ROC Curve Graphical Plot
plot_ROC_Curve(true_p,false_p,error);

%OUTPUT DATA
disp('PART B');
fprintf('Threshold (Empirical) = %.4f\n', exp(threshold(find(error==min(error)))))
fprintf('Minimum Error (Empirical) = %.4f\n', min(error))
fprintf('Threshold (Theoretical) = %.4f\n', exp(theoretical_threshold))
fprintf('Minimum Error (Theoretical) = %.4f\n', theoretical_error)

%PART C

%Scatter Matrices Calculation
SB_4 = transpose(mean1-mean2)*(mean1-mean2);
SB = SB_4(1,:);
SW = SD1 + SD2;
% disp(SW);
% disp(SB);
% disp(inv(SW));

%Class Label Generation
[C_L,smple] = input_and_class_labels(mean1,mean2,SD1,SD2,sample_size, priori,n);
```

```matlab
%Input Data Graphical Plot
plot_input_data(C_L,smple);

S_E = inv(SW)*SB_4;
% disp(S_E);
[weight,e_v_M] = eig(S_E);
e_v = eig(S_E);
% disp(weight);
% disp(e_v);
w_max = weight(find(e_v==max(e_v)));
%disp (w_max);


C_L_0 = smple(find(C_L == 0));
C_L_1 = smple(find(C_L == 1));

w_max_T = w_max.';
C_L_0_T = C_L_0.';
C_L_1_T = C_L_1.';
Y_0 = w_max_T*C_L_0_T;
Y_1 = w_max_T*C_L_1_T;
Y = [Y_0 Y_1];
Y_S = sort(Y.');


threshold = (Y_S(1:end-1) + Y_S(2:end)) / 2;
%Theoretical Threshold Calculation
theoretical_threshold = log(priori(2) / priori(1));
theoretical_decision = Y.' >= theoretical_threshold;
theoretical_true_p = numel(find((theoretical_decision==1) & (C_L==1))) /
numel(find(C_L==1));
theoretical_false_p = numel(find((theoretical_decision==1) & (C_L==0))) /
numel(find(C_L==0));
theoretical_error = priori(2)*theoretical_false_p + priori(1)*(1-theoretical_true_p);

%Data Classification
for i = 1:length(threshold)
    decision = Y.' >= threshold(i);
    true_p(i) = numel(find((decision==1) & (C_L==1))) / numel(find(C_L==1));
    false_p(i) = numel(find((decision==1) & (C_L==0))) / numel(find(C_L==0));
    error(i) = priori(2)*false_p(i) + priori(1)*(1-true_p(i));
end

%ROC Curve Graphical Plot
plot_ROC_Curve(true_p,false_p,error);

%OUTPUT DATA
disp('PART C');
fprintf('Threshold (Empirical) = %.4f\n', threshold(find(error==min(error))))
fprintf('Minimum Error (Empirical) = %.4f\n', min(error))
fprintf('Threshold (Theoretical) = %.4f\n', exp(theoretical_threshold))
fprintf('Minimum Error (Theoretical) = %.4f\n', theoretical_error)

%FUNCTIONS
function [C_L,smple] = input_and_class_labels (mean1,mean2,SD1,SD2,sample_size, priori,n)
    C_L = (rand(sample_size, 1) >= priori(1));
    C_L = double(C_L);
    smple = zeros(sample_size, n);
    for i = 1:sample_size
        if C_L(i) == 0
```

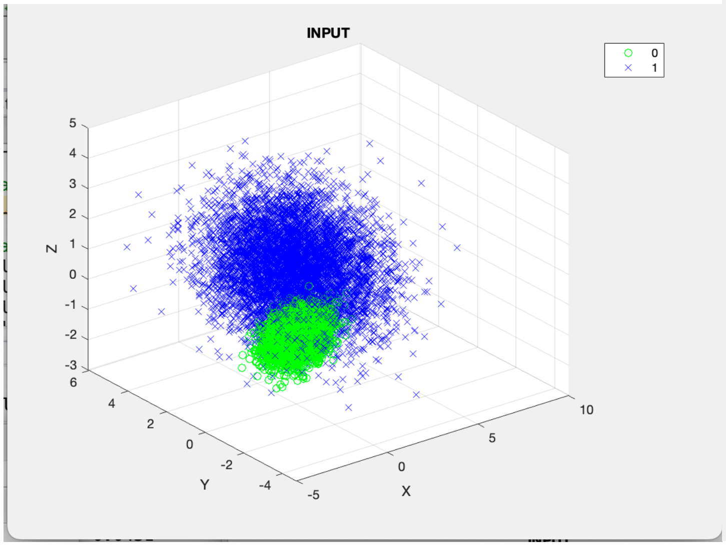
```
            smple(i,:) = mvnrnd(mean1, SD1);
        elseif C_L(i) == 1
            smple(i,:) = mvnrnd(mean2, SD2);
        end
    end
end

function plot_input_data(C_L,smple)
    figure
    scatter3(smple(C_L==0,4), smple(C_L==0,2), smple(C_L==0,3), 'o', 'g')
    hold on
    scatter3(smple(C_L==1,4), smple(C_L==1,2), smple(C_L==1,3), 'X', 'b')
    xlabel('X')
    ylabel('Y')
    zlabel('Z')
    legend('0','1')
    title('INPUT')
end

function [DS,sorted_DS,threshold] = calc_threshold(mean1,mean2,SD1,SD2,smple)
    DS = log(mvnpdf(smple, mean2, SD2)) - log(mvnpdf(smple, mean1, SD1));
    sorted_DS = sort(DS);
    threshold = (sorted_DS(1:end-1) + sorted_DS(2:end)) / 2;
end

function plot_ROC_Curve(true_p,false_p,error)
    figure
    plot(false_p, true_p, 'r')
    hold on
    plot(false_p(find(error==min(error))), true_p(find(error==min(error))), 'square',
'color', 'k')
    xlabel('False Positive')
    ylabel('True Positive')
    title('ROC Curve')
end
```

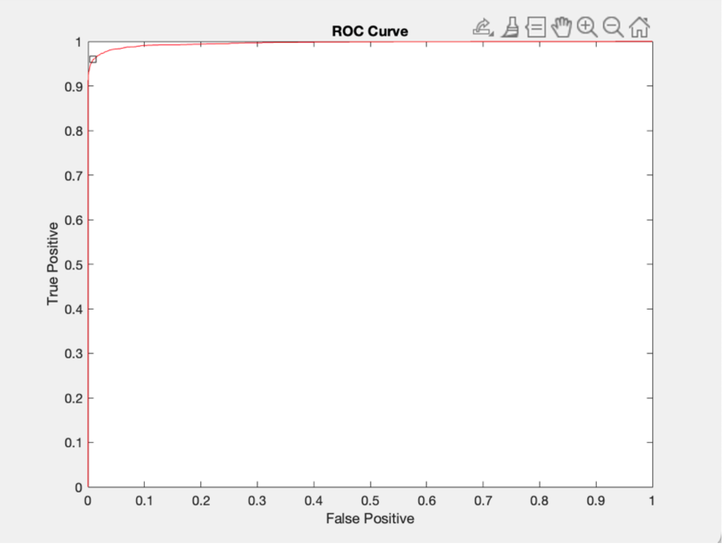
Outputs



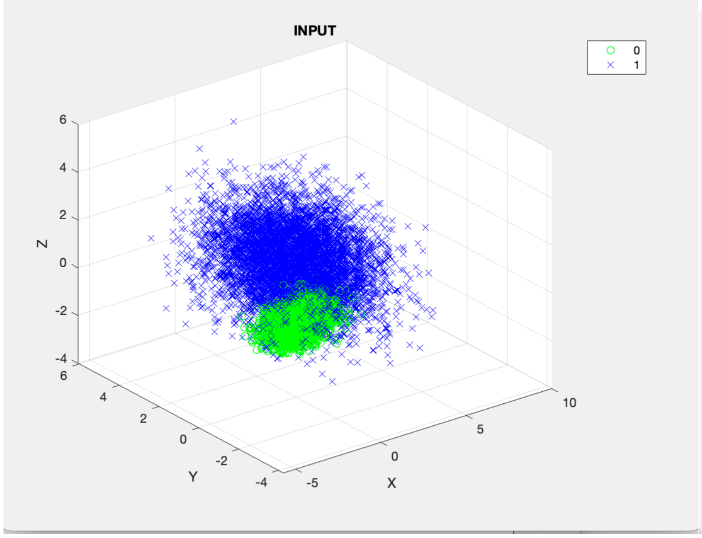
```
PART A
Threshold (Empirical) = 1.3994
Minimum Error (Empirical) = 0.0199
Threshold (theoretical) = 1.8571
Minimum Error (theoretical) = 0.0202
PART B
Threshold (Empirical) = 0.3265
Minimum Error (Empirical) = 0.0431
Threshold (Theoretical) = 1.8571
Minimum Error (Theoretical) = 0.0654
PART C
Threshold (Empirical) = 0.9922
Minimum Error (Empirical) = 0.3499
Threshold (Theoretical) = 1.8571
Minimum Error (Theoretical) = 0.3519
```

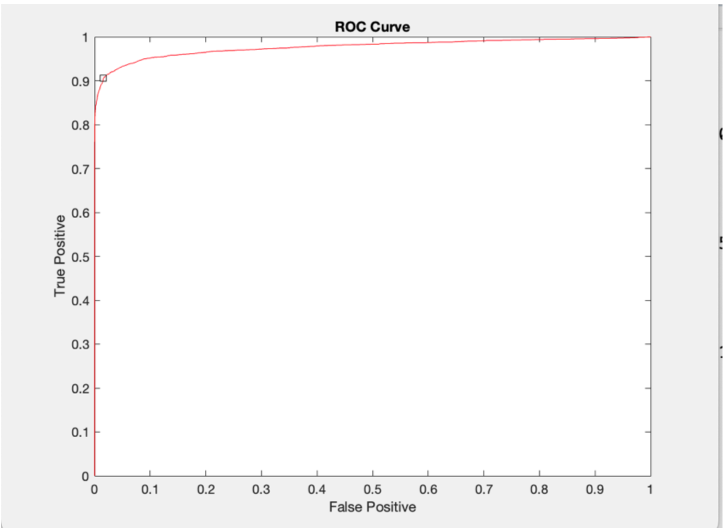
Part A Graphs



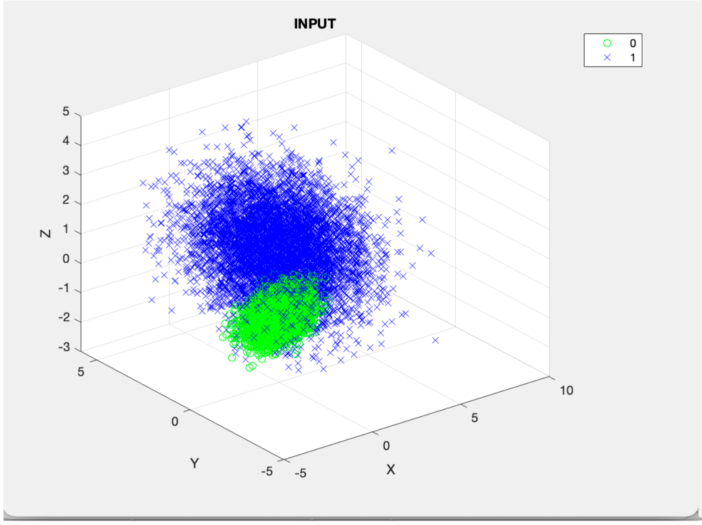


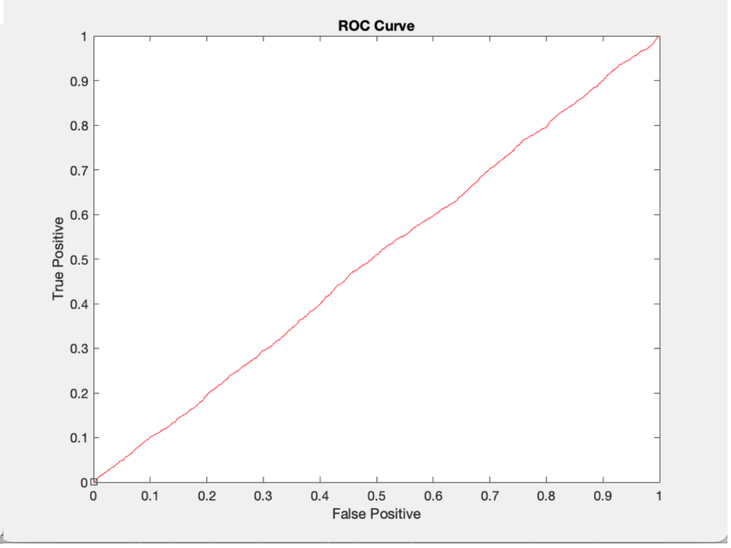
Part B Graphs





Part C Graphs





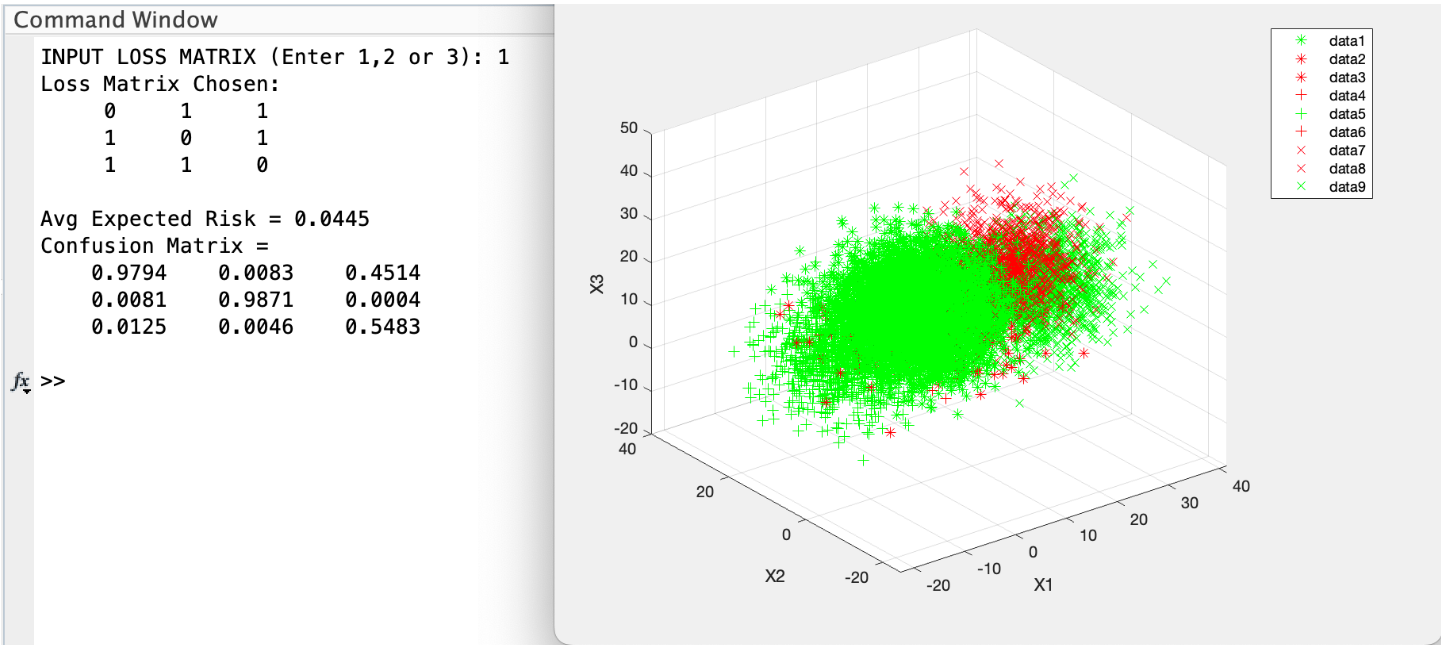
Question 2

Code (MATLAB)

```
clear all; close all; clc;

% INPUT PARAMETERS
n = 3; %3D Gaussian
m = 4; %4 mixtures
sample_size = 10000;
N_C = 3;
priori = [0.3, 0.3, 0.4];

% Mean vectors
m1 = [0, 0, 20];
m2 = [0, 20, 0];
m3 = [20, 0, 0];
m4 = [20, 0, 20];

% Covariance matrices
cvm1 = 36 * (eye(n) + (0.01 * randn(n, n))).^2;
cvm2 = 36 * (eye(n) + (0.02 * randn(n, n))).^2;
cvm3 = 36 * (eye(n) + (0.03 * randn(n, n))).^2;
cvm4 = 36 * (eye(n) + (0.04 * randn(n, n))).^2;

% Prior weights for Class 2
weights = [0.5, 0.5];

%Class Label and Input Generation
[C_L,smple,mean,cvm] =
input_and_class_labels(m,weights,priori,sample_size,m1,m2,m3,m4,cvm1,cvm2,cvm3,cvm4,n);


LMA = ones(N_C) - eye(N_C);
LMB = [0, 1, 10; 1, 0, 10; 1, 1, 0];
LMC = [0, 1, 100; 1, 0, 100; 1, 1, 0];

LM = input_loss_m(LMA,LMB,LMC);
disp('Loss Matrix Chosen:')
disp(LM);

%PDF of each conditional class
prob_X_given_L = zeros(N_C, sample_size);
for i = 1:N_C
        prob_X_given_L(i,:) = mvnpdf(smple, mean(i,:), squeeze(cvm(:,:,i)));
end

%Posterior Calculation
prob_X = priori * prob_X_given_L;
class_Pos = (prob_X_given_L .* repmat(priori', 1, sample_size))./repmat(prob_X, N_C, 1);

[conf_M, avg_exp_risk,dec] =
calc_risk_and_confusion_matrix(LM,class_Pos,sample_size,N_C,C_L);

%PLOTTING OUTPUT
plot_output(smple,C_L,dec)

%Printing OUTPUT
fprintf('Avg Expected Risk = %.4f\n', avg_exp_risk);
disp('Confusion Matrix = ');
disp(conf_M);


%Functions
function [C_L,smple,mean,cvm] =
input_and_class_labels(m,weights,priori,sample_size,m1,m2,m3,m4,cvm1,cvm2,cvm3,cvm4,n)
    mean = [m1;m2;m3;m4];
    cvm = zeros(n, n, m);
    cvm (:, :, 1) = cvm1;
    cvm (:, :, 2) = cvm2;
    cvm (:, :, 3) = cvm3;
    cvm (:, :, 4) = cvm4;
    CP_Cumulative = cumsum(priori);
    rand = randn(sample_size, 1);
    C_L = zeros(size(rand));

    %Generating Class Labels
    for i = 1:sample_size
        if rand(i) <= CP_Cumulative(1)
            C_L(i) = 0;
        elseif rand(i) <= CP_Cumulative(2)
            C_L(i) = 1;
        else
            C_L(i) = 2;
        end
    end

    %sample generation
    smple = zeros(10000, n);

    cvm(:,:,1) = (cvm(:,:,1) + cvm(:,:,1)')/2;
    cvm(:,:,2) = (cvm(:,:,2) + cvm(:,:,2)')/2;
    cvm(:,:,3) = (cvm(:,:,3) + cvm(:,:,3)')/2;
    cvm(:,:,4) = (cvm(:,:,4) + cvm(:,:,4)')/2;

    for i = 1:sample_size
        if C_L(i) == 0
            smple(i, :) = mvnrnd(mean(1,:), squeeze(cvm(:,:,1)));
        elseif C_L(i) == 1
            smple(i, :) = mvnrnd(mean(2,:), squeeze(cvm(:,:,2)));
        elseif C_L(i) == 2
            if rand(1,1) >= weights(1)
                smple(i,:) = mvnrnd(mean(3,:), squeeze(cvm(:,:,3)));
            else
                smple(i,:) = mvnrnd(mean(4,:), squeeze(cvm(:,:,4)));
            end
        end
    end

end

function LM = input_loss_m(LMA,LMB,LMC)
    ip = input("INPUT LOSS MATRIX (Enter 1,2 or 3): ");
    if ip == 1
        LM = LMA;
    elseif ip == 2
        LM = LMB;
    elseif ip == 3
        LM = LMC;
    end
end
```

```
function [conf_M, avg_exp_risk,dec] =
calc_risk_and_confusion_matrix(LM,class_Pos,sample_size,N_C,C_L)
    exp_risk = LM * class_Pos;
    [~, dec] = min(exp_risk, [], 1);
    dec = dec-1;
    dec = dec';
    avg_exp_risk = sum(min(exp_risk, [], 1))/sample_size;

    conf_M = zeros(N_C);
    for i = 1:N_C
        for j = 1:N_C
            conf_M(i,j) = numel(find((i-1 == dec) & (j-1 == C_L)))/numel(find(C_L == j-
1));
        end
    end
end

function plot_output(smple,C_L,dec)
    smple1 = smple(:,1);
    smple2 = smple(:,2);
    smple3 = smple(:,3);
    fig = figure;
    ax = axes('Parent',fig,'Projection','Perspective');
    figure(1)
    scatter3(ax, smple1(C_L==0 & dec==0), smple2(C_L==0 & dec==0), smple3(C_L==0 &
dec==0),'*','g');
    hold on;
    scatter3(ax, smple1(C_L==0 & dec==1), smple2(C_L==0 & dec==1), smple3(C_L==0 &
dec==1),'*','r');
    scatter3(ax, smple1(C_L==0 & dec==2), smple2(C_L==0 & dec==2), smple3(C_L==0 &
dec==2),'*','r');
    scatter3(ax, smple1(C_L==1 & dec==0), smple2(C_L==1 & dec==0), smple3(C_L==1 &
dec==0),'+','r');
    scatter3(ax, smple1(C_L==1 & dec==1), smple2(C_L==1 & dec==1), smple3(C_L==1 &
dec==1),'+','g');
    scatter3(ax, smple1(C_L==1 & dec==2), smple2(C_L==1 & dec==2), smple3(C_L==1 &
dec==2),'+','r');
    scatter3(ax, smple1(C_L==2 & dec==0), smple2(C_L==2 & dec==0), smple3(C_L==2 &
dec==0),'X','r');
    scatter3(ax, smple1(C_L==2 & dec==1), smple2(C_L==2 & dec==1), smple3(C_L==2 &
dec==1),'X','r');
    scatter3(ax, smple1(C_L==2 & dec==2), smple2(C_L==2 & dec==2), smple3(C_L==2 &
dec==2),'X','g');
    xlabel('X1');
    ylabel('X2');
    zlabel('X3');
    grid on;
    legend show;
end
```

Outputs



```
INPUT LOSS MATRIX (Enter 1,2 or 3): 1
Loss Matrix Chosen:
     0     1     1
     1     0     1
     1     1     0

Avg Expected Risk = 0.0445
Confusion Matrix =
    0.9794    0.0083    0.4514
    0.0081    0.9871    0.0004
    0.0125    0.0046    0.5483

fx >>
```

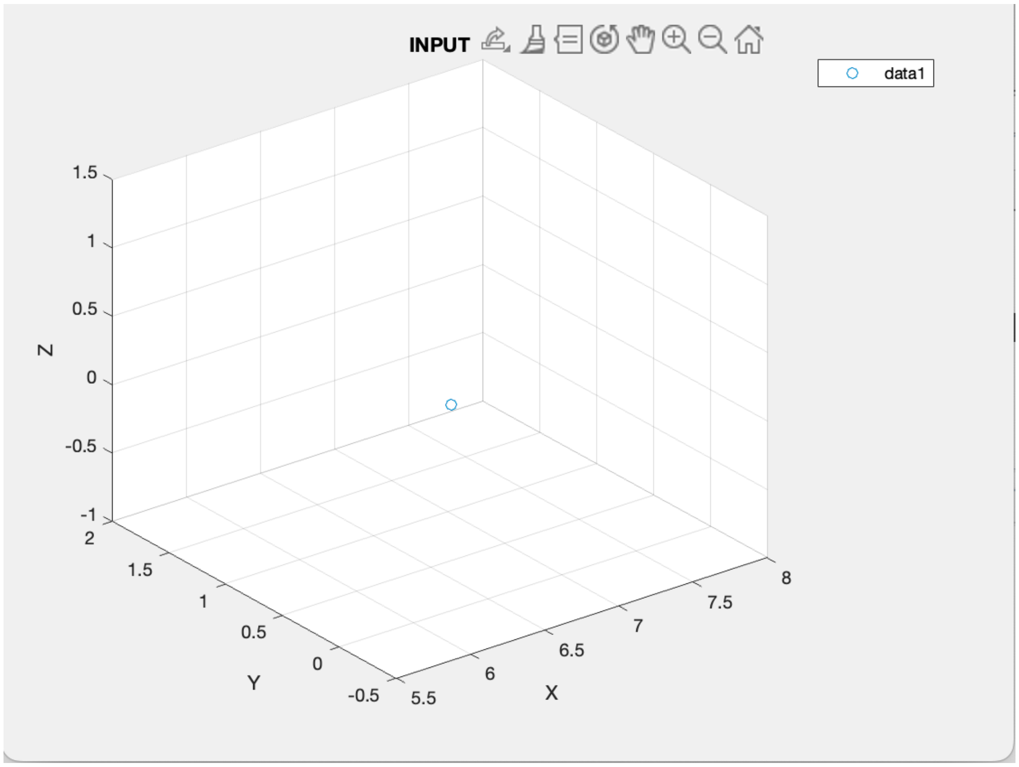
Question 3

Code (MATLAB)

```
%Read Data Sets
D_S_R = readmatrix('/Users/prasasth/Downloads/winequality-red.csv');
D_S_W = readmatrix('/Users/prasasth/Downloads/winequality-white.csv');
D_S = [D_S_R;D_S_W];

labels = 11;
features = 11;

%Input variables (based on physicochemical tests):
%    1 - fixed acidity
%    2 - volatile acidity
%    3 - citric acid
%    4 - residual sugar
%    5 - chlorides
%    6 - free sulfur dioxide
%    7 - total sulfur dioxide
%    8 - density
%    9 - pH
%    10 - sulphates
%    11 - alcohol
%    Output variable (based on sensory data):
%    12 - quality (score between 0 and 10)
D_Var = 1:12;
D_Lab = 1:11;

figure
for i = 1:labels
    scatter3(D_S((D_Lab==i),1),D_S((D_Lab==i),2),D_S((D_Lab==i),3));
end
xlabel('X')
ylabel('Y')
```

```
zlabel('Z')
legend show;
title('INPUT')

D_S = normalize(D_S,'zscore');

[coeff,D_S,latent] = pca(D_S);

mean_M = zeros(labels,features);
cv_M = zeros(labels,features,features);

for i = 1:labels
    mean_M(i, :) = mean(D_S(find(D_Lab==i),:));
    cv_M(i,:,:) = cov(D_S(find(D_Lab==i),:));
    cv_M(i,:,:) = squeeze(cv_M(i,:,:)) +
(0.00001)*(trace(squeeze(cv_M(i,:,:)))/rank(squeeze(cv_M(i,:,:))))*eye(11);
end

loss_matrix = ones(labels,labels) - eye(labels);

Prob_X_given_L = zeros(labels,size(D_S,1));

for i = 1:labels
    Prob_X_given_L(i, :) = mvnpdf(D_S(:,1:11),mean_M(i, :),squeeze(cv_M(i, :,:)));
end

priori = zeros(labels,1);
for i = 1:labels
    priori(i,1) = (size(D_Lab(find(D_Lab==i)),1)) / size(D_S,1);
end

P_X = priori'*Prob_X_given_L;
CP_rep1 = repmat(priori,1,size(D_S,1));
CP_rep2 = repmat(P_X,labels,1);
posteriori = (Prob_X_given_L' * CP_rep1)/CP_rep2;

exp_risk = loss_matrix*posteriori';
dec = min(exp_risk);
avg_exp_risk = sum(min(exp_risk', [], 1))/6497;


D_Lab = D_S(:,12);
conf_M = zeros(labels,labels);
for i = 1:labels
    for j = 1:labels
        conf_M(i,j) = numel(find((i-1==dec) & (j-1==D_Lab))) / numel(find(D_Lab==j-1));
    end
end
```

Outputs



CITATIONS:

- P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis.
  Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.