EECE5644 HW1

CONTENT

Question 1

Part A

The mean and covariance matrix values given in problem 1 were used to first generate 10,000 samples pictured in Figure 1 below.

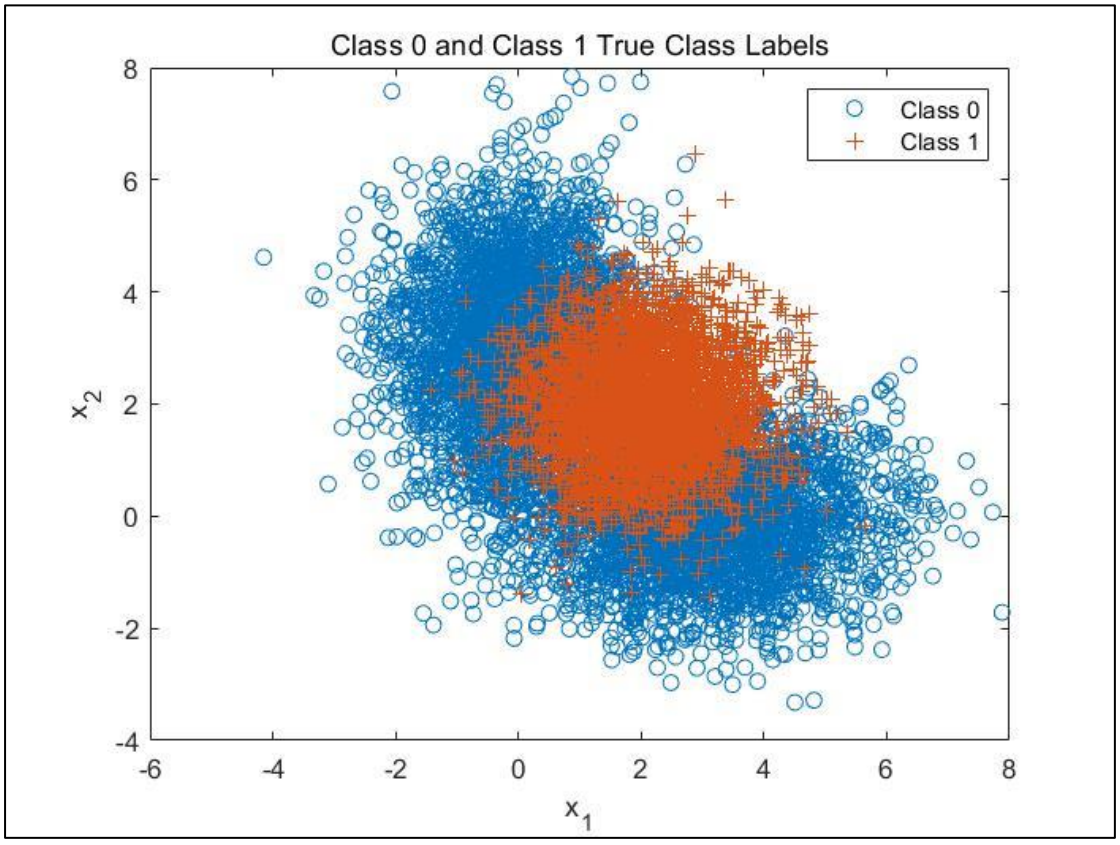

Figure 1: Question 1 class distributions and true labels of data points

1. The minimum expected risk classification rule:

$$g(x \mid m_0, c_0) = w_1 g(x \mid m_{01}, c_{01}) + w_2 g(x \mid m_{02}, c_{02})$$

$$(D=1) \quad \frac{g(x \mid m_0, c_0)}{g(x \mid m_1, c_1)} \gtrless \frac{P(L=0)}{P(L=1)}\left(\frac{\lambda_{10}-\lambda_{00}}{\lambda_{01}-\lambda_{11}}\right) = \frac{0.65}{0.35}\left(\frac{\lambda_{10}-\lambda_{00}}{\lambda_{01}-\lambda_{11}}\right) \quad (D=0)$$

$$\text{That is} \quad \frac{w_1 g(x|m_{01}, c_{01}) + w_2 g(x|m_{02}, c_{02})}{g(x|m_1, c_1)} \gtrless 1.86\left(\frac{\lambda_{10}-\lambda_{00}}{\lambda_{01}-\lambda_{11}}\right)$$

2. The classifier was implemented for multiple values of gamma and the ROC curve is shown in Figure 2 below. The locations of the theoretical minimum error (orange plus) as well as the minimum error determined by a parametric sweep of gamma (red circle) are marked on the plot.

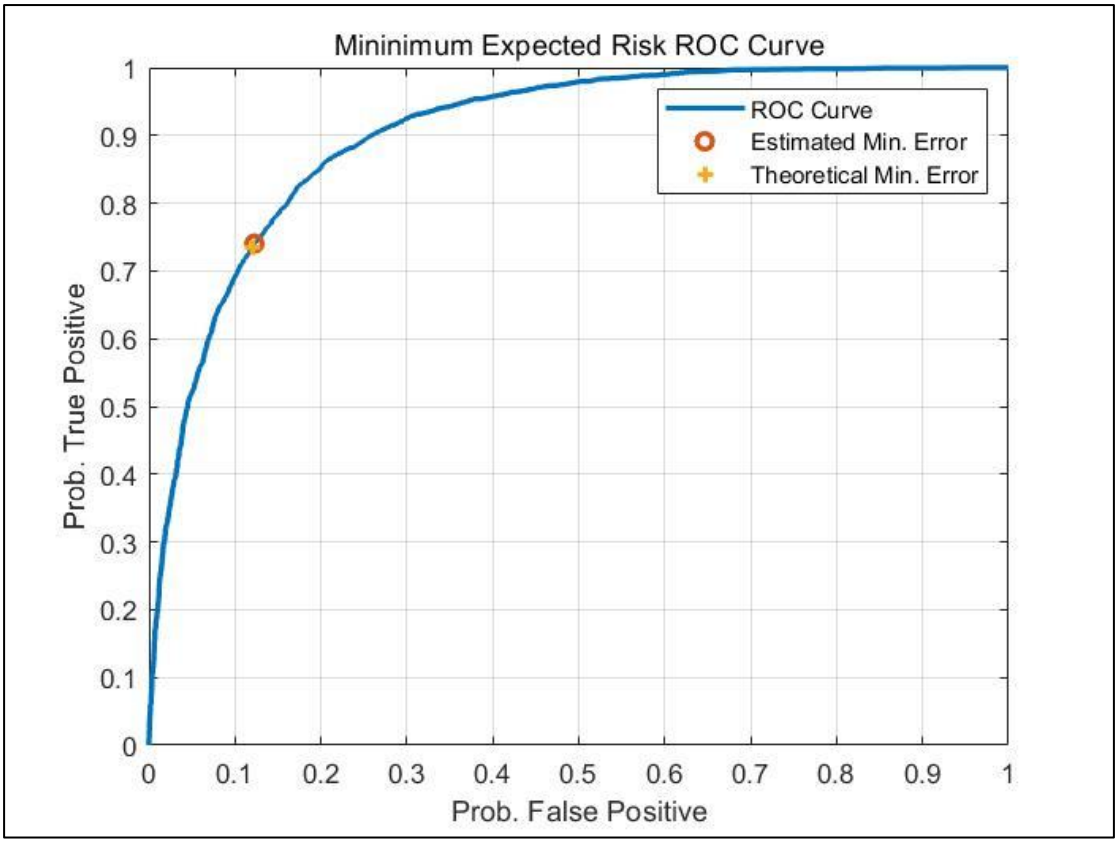

Figure 2: ROC Curve for ERM Classification

3. Table 1 contains the theoretical and estimated gamma values that result in the minimum probability of error. As can be seen the two values closely align providing confidence in the estimated value.

$$P_e = 1 - \mathrm{P(D = 0|L = 0)P(L = 0)} - \mathrm{P(D = 1|L = 1)P(L = 1)} = 0.1719$$

Table 1 compares the theoretical and the calculated minimum probability of error.

Table 1: Comparison of Gammas to Produce Minimum Errors

	γ	min P_e
Theoretical	1.86	0.1719

Calculated from data	1.82	0.1711

Figure 3 shows a plot of the probability of errors versus the gamma parameters. The location of the minimum error is marked. Additionally, as the gamma parameter approached its limits at 0 and +∞ the probability of error asymptotes to the priors for the two distributions. That is when the gamma is set to its minimum value all of the data points will be classified as class 1 so the overall error will be the proportion of data in class 0 which is equivalent to its prior. Similarly, when gamma is at +∞ all of the data points will be classified as class 0 and so all of the class 1 data will be misclassified, and the probability of error is equivalent to the prior for class 1.

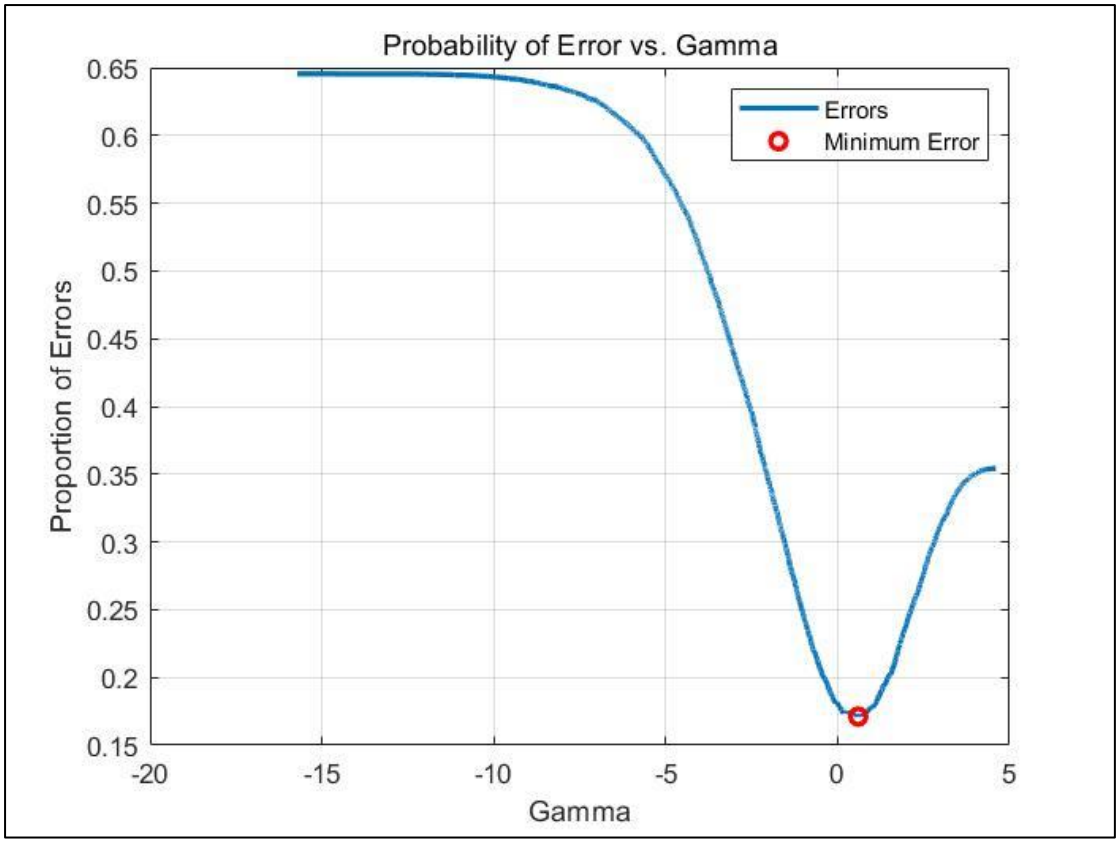

Figure 3: Probability of Error vs. ln(Gamma) for ERM Classification

Part B

In part B of Question 1, Fisher Linear Discriminant Analysis (LDA) was used to create a classifier and plot

1. The fisher LDA classification rule:

$$(D = 1) \quad w_{LDA}x \geq \tau \quad (D = 0)$$

Where W_{LDA} is the generalized eigenvector of the scatter matrices S_B and S_w with the largest eigenvalue as described by the following equations.

$$S_w^{-1} S_B W = \lambda W$$
$$S_W = \Sigma_0 + \Sigma_1$$
$$S_B = (\mu_0 - \mu_1)(\mu_0 - \mu_1)^T$$

Figure 4 below shows the resulting projection of the data.

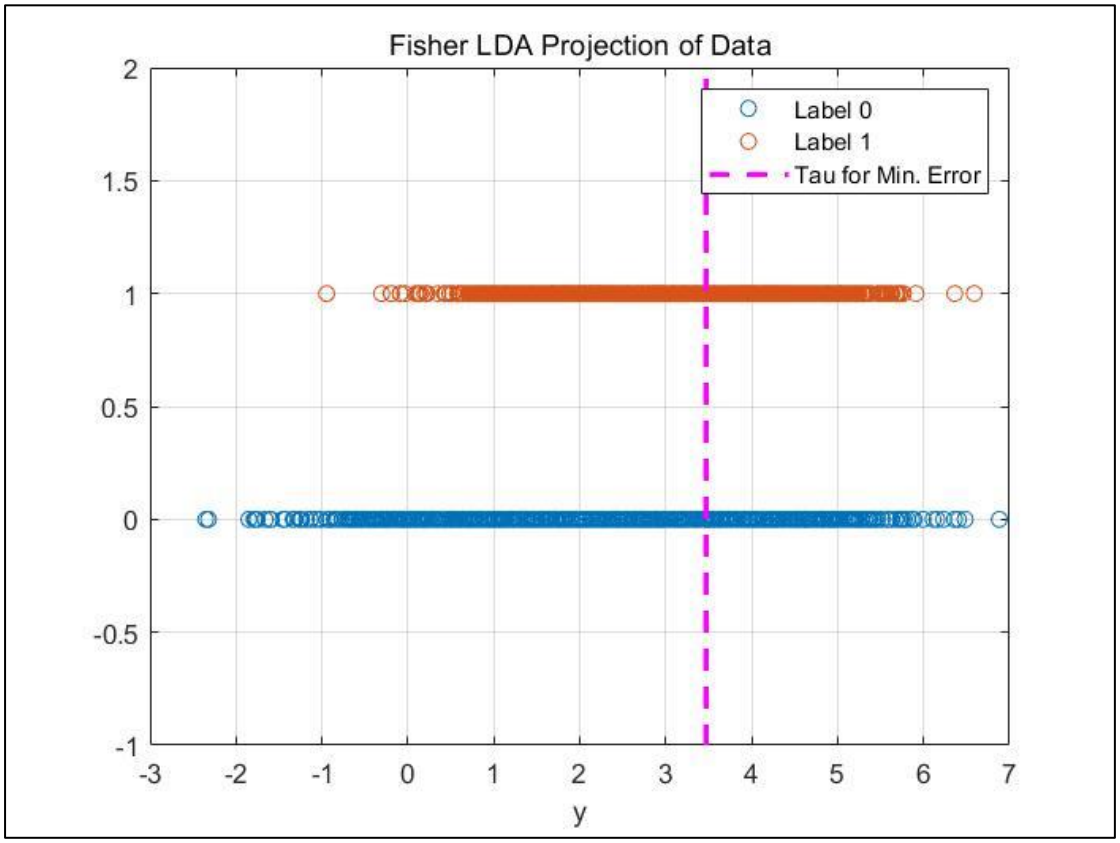

Figure 4: Fisher LDA Projection

2. Figure 5 below displays the ROC curve generated after implementing the LDA classifier from above, applying it to the 10,000 generated samples, and varying the threshold τ from -∞ to ∞.

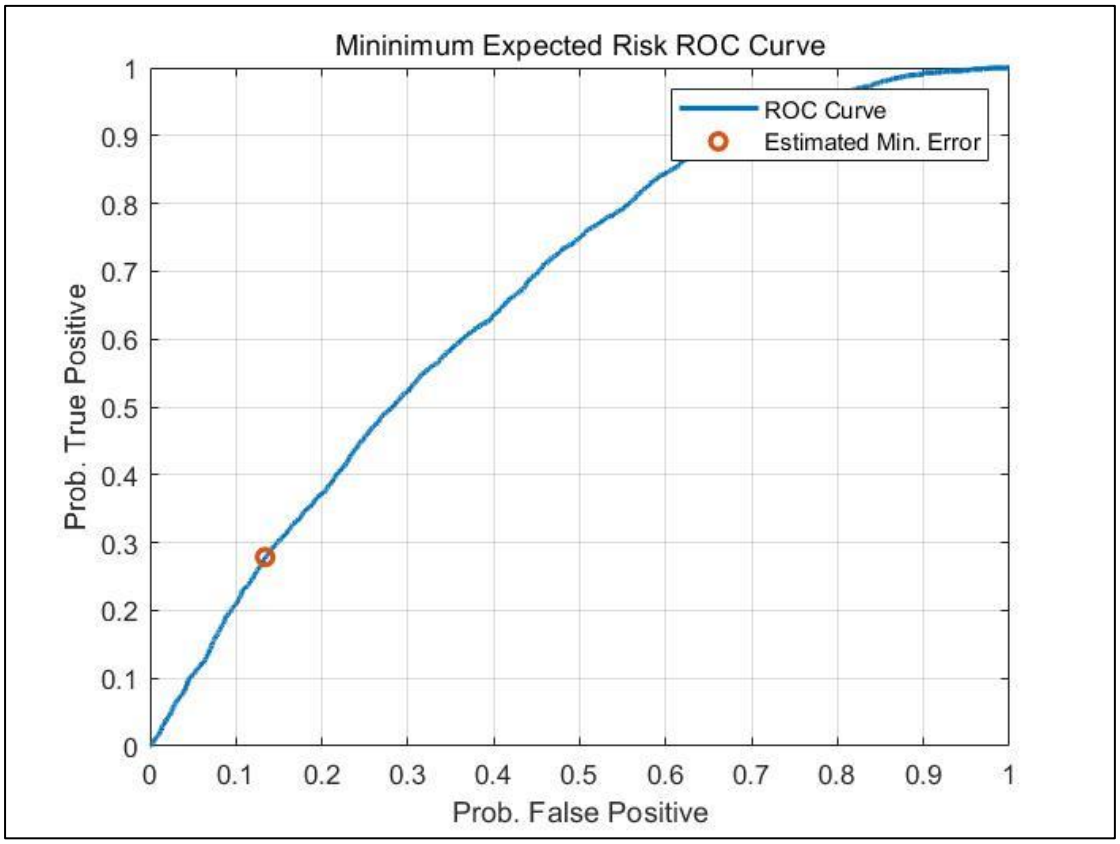

Figure 5: ROC Curve for Fisher LDA

Based on the 10,000 samples generated, the minimum probability of error was calculated to be P_e = 0.3421 at the threshold value of τ = 3.47. This error and threshold value were calculated by finding the minimum error as the value of τ was changed from $-\infty$ to ∞. The red circle in Figure 5 above marks this point.

Table 2 shows the comparison of fisher LDA with ERM classification. ERM performs better than Fisher LDA.

Table 2: Min Error comparison

	ERM	Fisher LDA
Theoretical value	0.1719	0.35
Calculated value	0.1711	0.3421

Figure 6 shows a plot of probability of error versus the tau parameter. The shape of this curve is similar to the curves for both of the ERM based approaches.

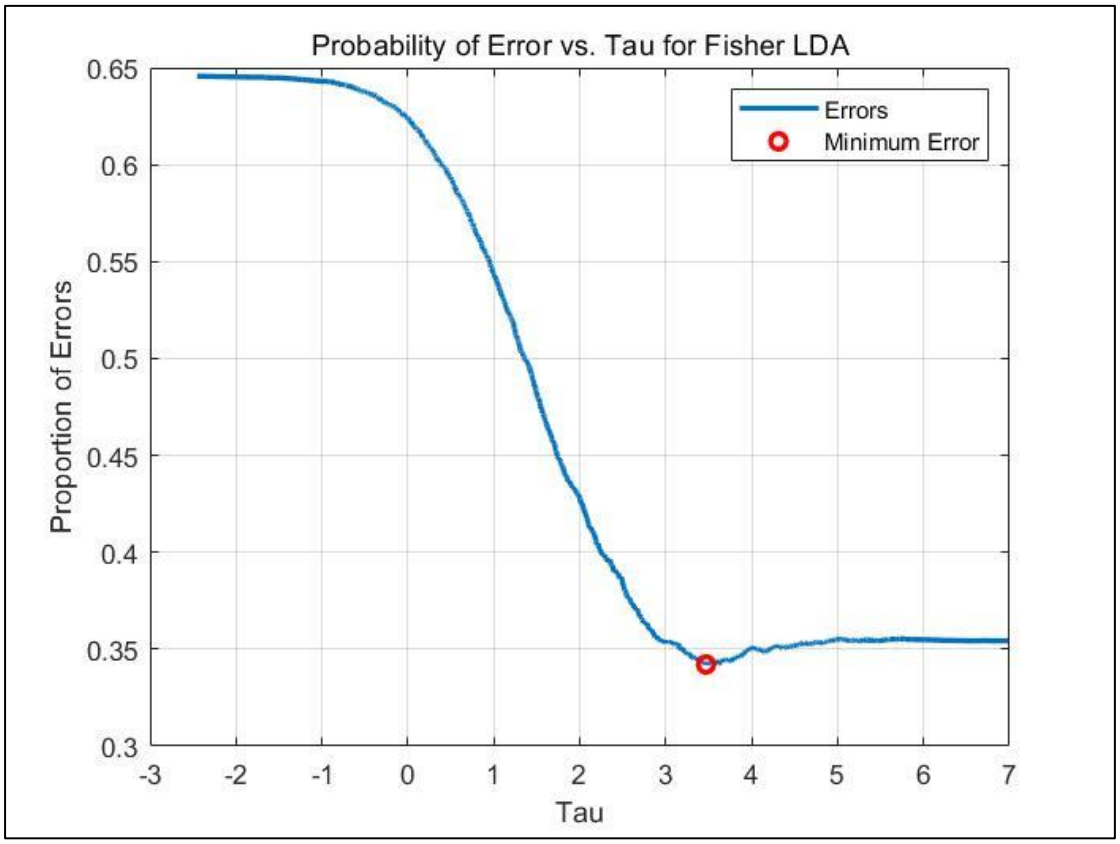

Figure 6: Fisher LDA Probability of Error vs. Tau

Question 2

Part A

The class priors and parameters are as follows.

$$P(X \mid L = 1): P(L = 1) = 0.3, \mu = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \sigma^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

$$P(X \mid L = 2): P(L = 2) = 0.3, \mu = \begin{bmatrix} 4 \\ 4 \\ 4 \end{bmatrix} \sigma^2 = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P(x \mid L = 3): P(\mathrm{L} = 3) = 0.4, \mu = \begin{bmatrix} 7 \\ 7 \\ 7 \end{bmatrix} \sigma^2 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

$$\text{Or } \mu = \begin{bmatrix} 11 \\ 11 \\ 11 \end{bmatrix} \sigma^2 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

1. The mean and covariance matrix values given in Question 2 were used to first generate 10,000 samples pictured in Figure 7 below.

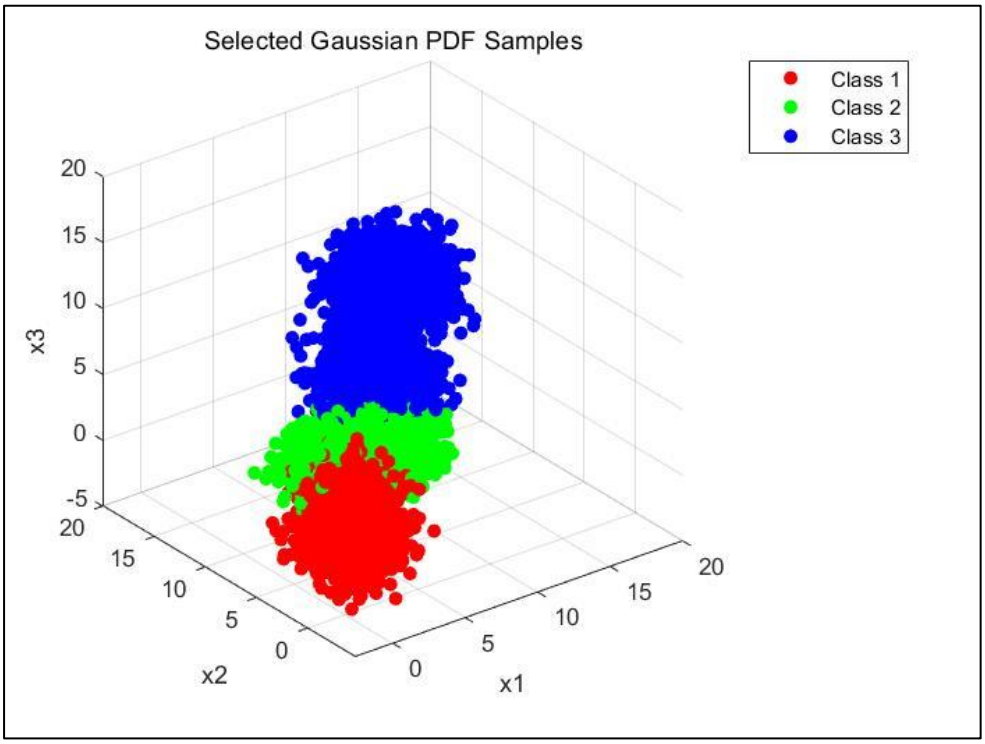

Figure 7: Question 2 class distributions

2. A decision rule that achieves the min probability of error is the 0-1 loss, which is a special case decision rule as:

$$R(\alpha_i \mid x) = \sum_{j=1}^{c} \lambda(\alpha_i \mid w_j) \cdot \mathrm{P}(\mathrm{w_j} \mid x)$$
$$= \sum_{j \neq i} P(w_j \mid x)$$

Figure 8 shows the the confusion matrix of how the samples were classified according to this decision rule.

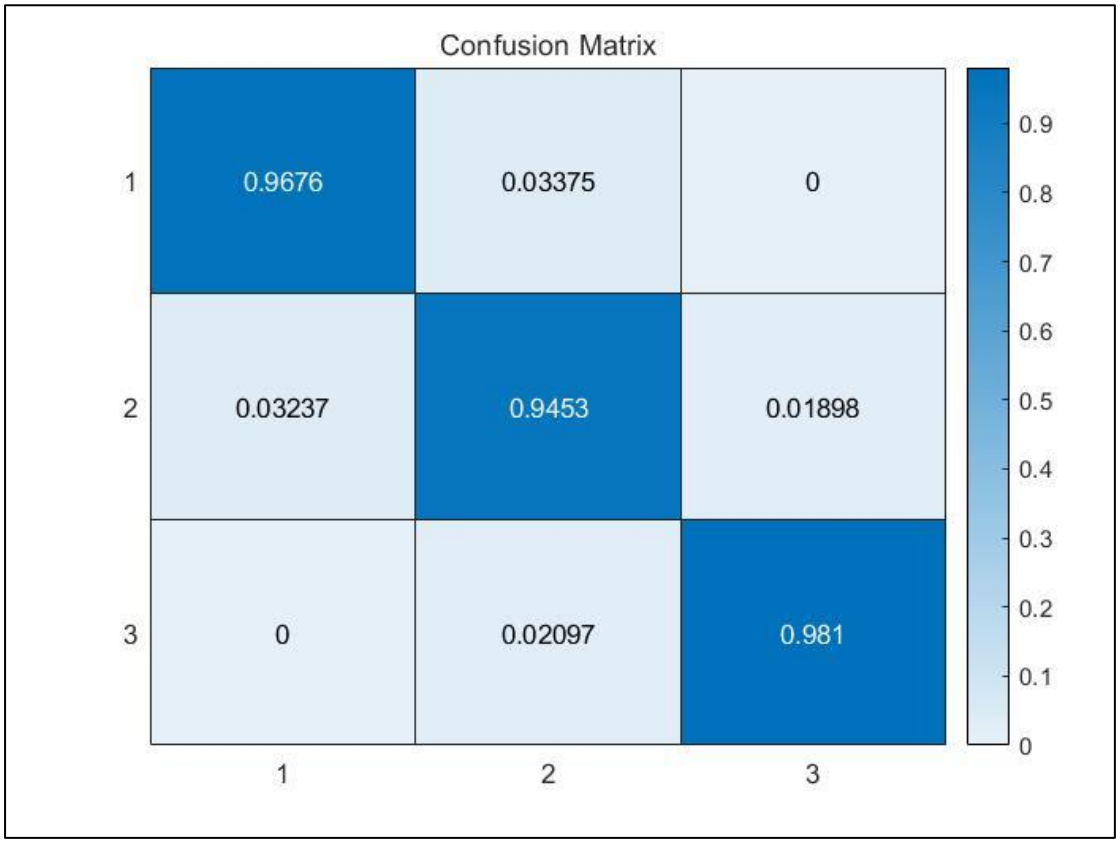

Figure 8: Confusion matrix over the true classification with 0-1 loss

Figure 9 shows the visualization of the data

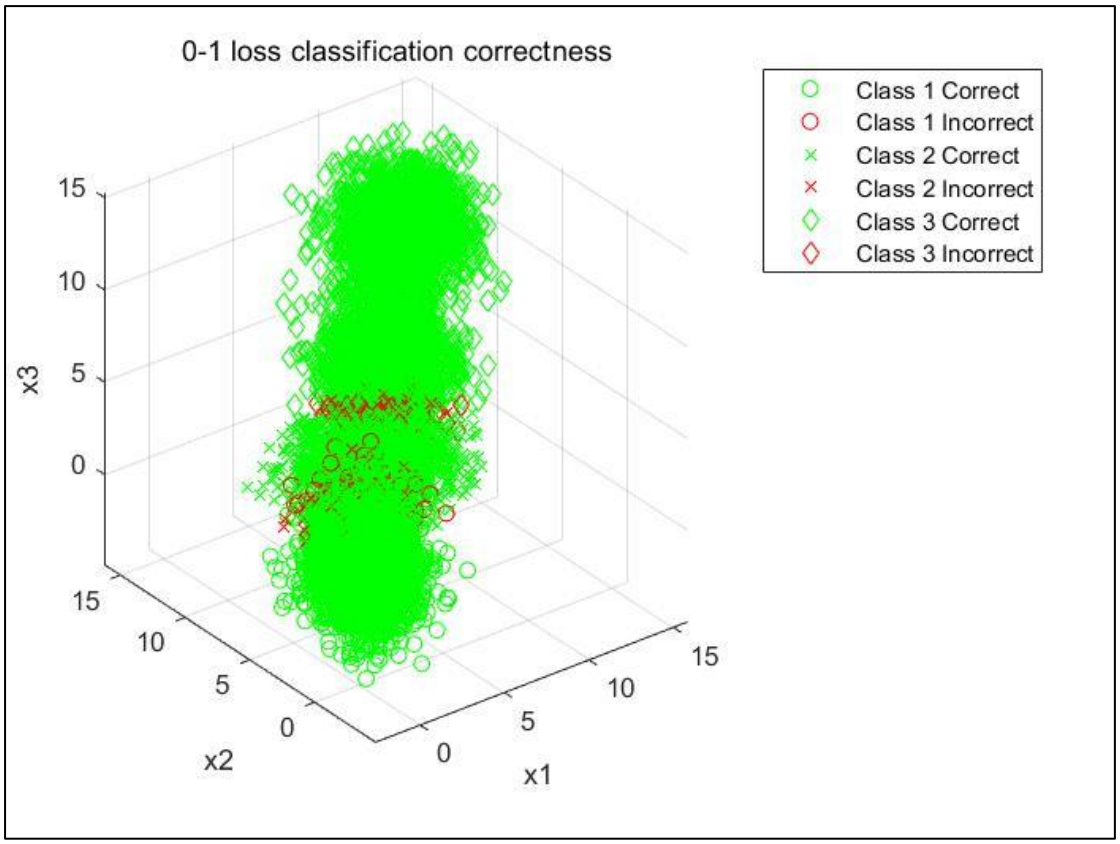

Figure 9: Data visualization with 0-1 loss

Part B

1. Decision Rules:

$$R(\alpha_i \mid x) = \sum_{j=1}^{C} \lambda(\alpha_i \mid w_j) \cdot \mathrm{P}(\mathrm{w_j} \mid x)$$

When L=3, Figure 10 and figure 11 show how the samples were classified according to this decision rule and the confusion matrix when the given decision rule cares 10 times.

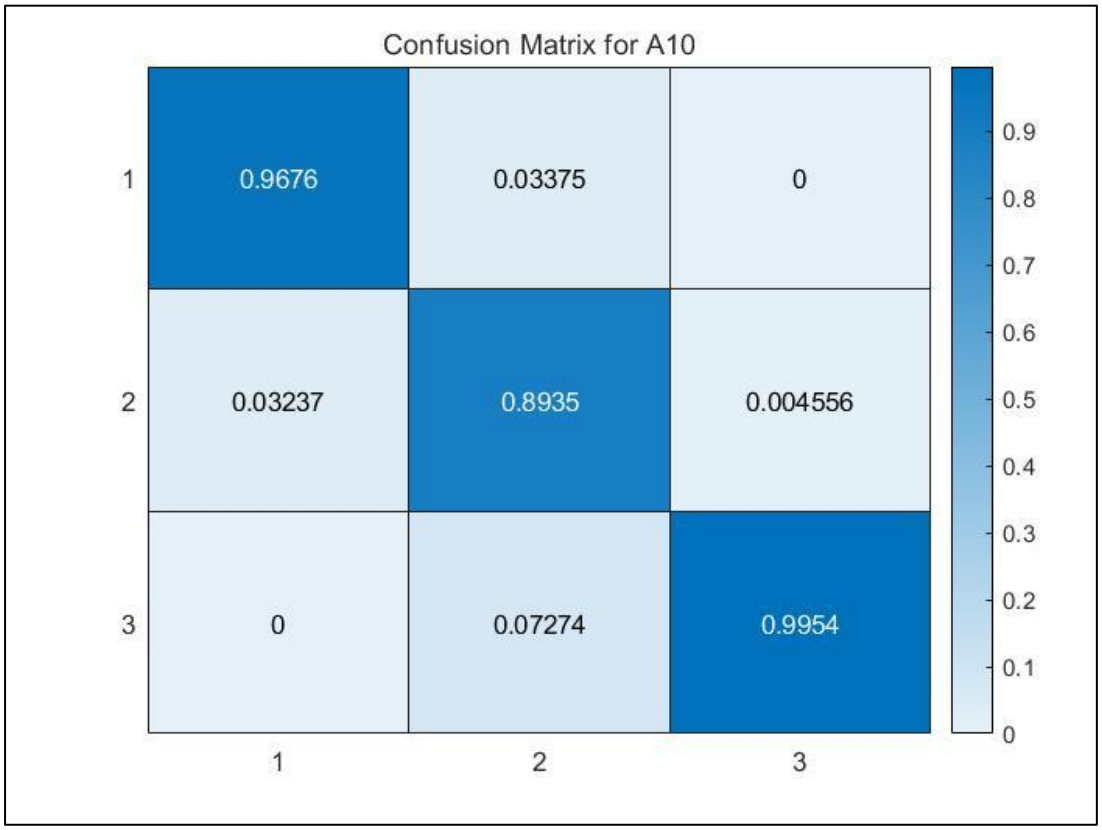

Figure 10: Confusion matrix of loss matrix A10

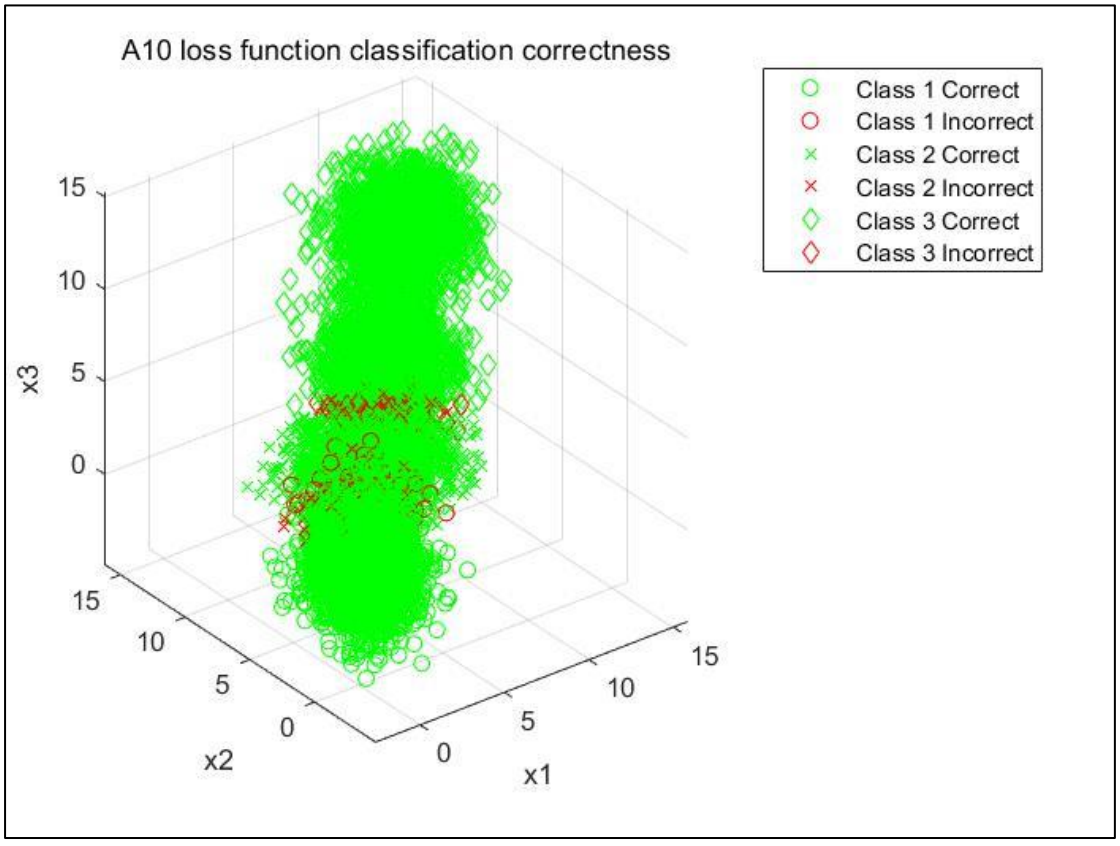

Figure 11: Classification correctness of loss matrix A10

Figure 12 and figure 13 show how the samples were classified according to this decision rule and the confusion matrix when the given decision rule cares 100 times.

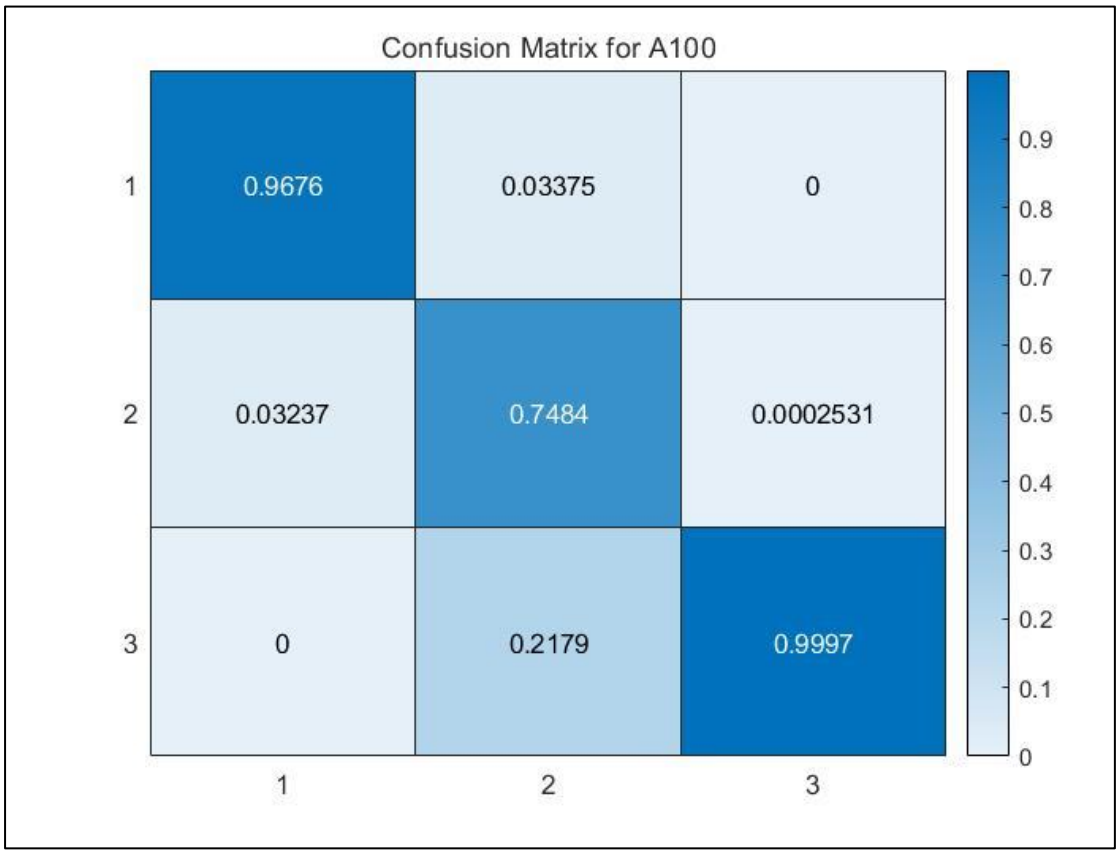

Figure 12: Confusion matrix of loss matrix A100

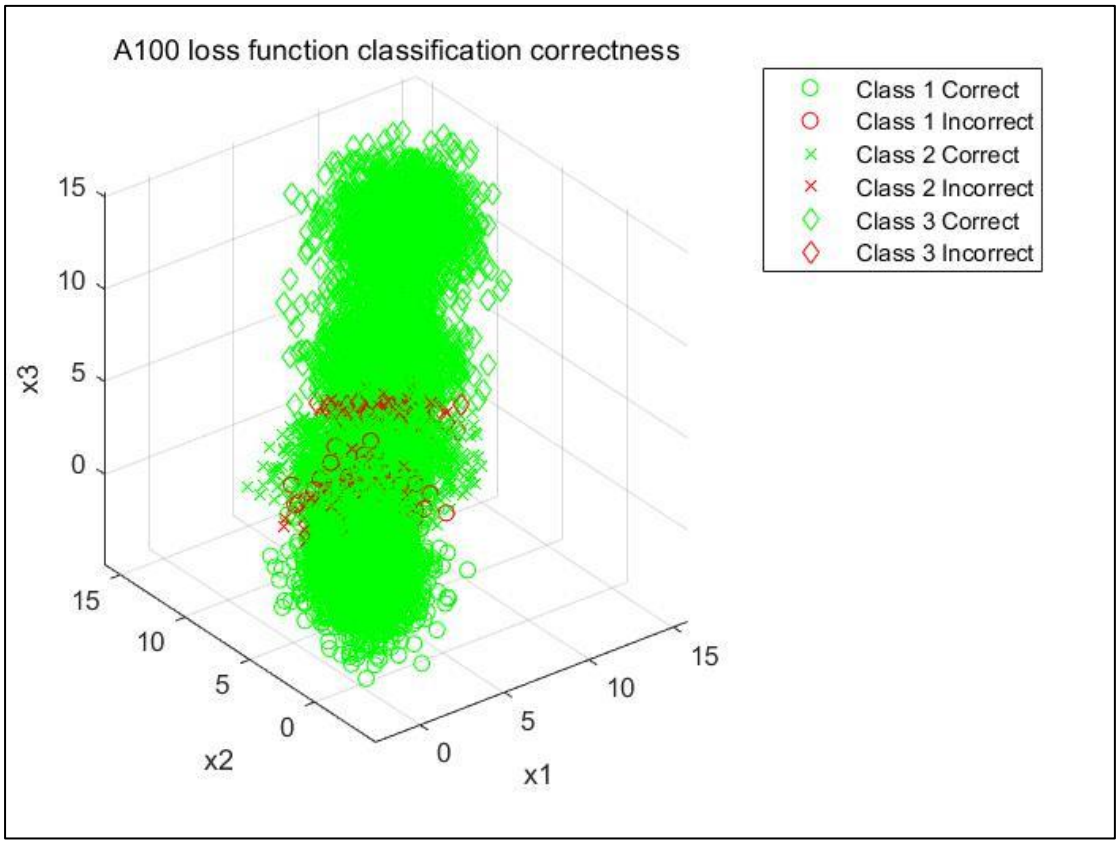

Figure 13: Classification correctness of loss matrix A100

2. Insights

When the loss matrix is modified and the Class 3 with greater risk is classified, the classifier will produce more errors.

When the modified matrix is A10, under the condition of L = 3, more data will be classified as class 3. Fusion matrix shows the classification accuracy of a single class. When the loss matrix is A100, comparing to A10, the accuracy of class 3 will not be significantly improved. At the same time, the classifier will move the decisions of categories 1 and 2 more to class 3. When A100 is applied, a large amount of data will be classified into class 3. This will lead to more incorrect classifications in class 1 and 2.

Appendix code for Question 1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%EECE5644 Fall 2021
% Wang Yinan 001530926 | HW1
%%=========================Question 1=========================%%
% Code help and example from Prof.Deniz
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;close all;clc;

%%=========================Setup=========================%%
n=2; %dimensions
N=10000; %samples

% Label 0 GMM Stats
mu0(:,1) = [3;0];
mu0(:,2) = [0;3];
Sigma0(:,:,1)=[2 0;0 1];
Sigma0(:,:,2)=[1 0;0 2];
alpha0=[0.5 0.5];

% Label 1 Single Gaussian Stats
mu1=[2 2]';
Sigma1=[1 0;0 1];
alpha1=1;

% Determine posteriors
p=[0.65,0.35];

% Create appropriate number of data points from each distribution
x=zeros(n,N);
label=rand(1, N) >= p(1);
Nc=[sum(label==0),sum(label==1)];

% Generate data as prescribed in assignment description
x(:,label==0)=randGMM(Nc(1),alpha0,mu0,Sigma0);
x(:,label==1)=randGMM(Nc(2),alpha1,mu1,Sigma1);

% Plot true class labels
figure(1);
plot(x(1,label==0),x(2,label==0),'o',x(1,label==1),x(2,label==1),'+');
title('Class 0 and Class 1 True Class Labels')
xlabel('x_1'),ylabel('x_2')
legend('Class 0','Class 1')
```

```
%%=========================Part A=========================%%
% ERM Classification with True Knowledge
px0=evalGMM(x,alpha0,mu0,Sigma0);
px1=evalGaussian(x,mu1,Sigma1);
discScore=log(px1./px0);
sortDS=sort(discScore);

% Generate vector of gammas for parametric sweep
logGamma=[min(discScore)-eps sort(discScore)+eps];
for ind=1:length(logGamma)
    decision=discScore>logGamma(ind);
    Num_pos(ind)=sum(decision);
    pFP(ind)=sum(decision==1 & label==0)/Nc(1);
    pTP(ind)=sum(decision==1 & label==1)/Nc(2);
    pFN(ind)=sum(decision==0 & label==1)/Nc(1);
    pTN(ind)=sum(decision==0 & label==0)/Nc(2);
    %Two ways to make sure I did it right
    pFE(ind)=(sum(decision==0 & label==1) + sum(decision==1 & label==0))/N;
    pFE2(ind)=(pFP(ind)*Nc(1) + pFN(ind)*Nc(2))/N;
end

% Calculate Theoretical Minimum Error
logGamma_ideal=log(p(1)/p(2));
decision_ideal=discScore>logGamma_ideal;
pFP_ideal=sum(decision_ideal==1 & label==0)/Nc(1);
pTP_ideal=sum(decision_ideal==1 & label==1)/Nc(2);
pFE_ideal=(pFP_ideal*Nc(1)+(1-pTP_ideal)*Nc(2))/(Nc(1)+Nc(2));

% Estimate Minimum Error
% If multiple minimums are found choose the one closest to the theoretical
% minimum
[min_pFE, min_pFE_ind]=min(pFE);
if length(min_pFE_ind)>1
    [~,minDistTheory_ind]=min(abs(logGamma(min_pFE_ind)-logGamma_ideal));
    min_pFE_ind=min_pFE_ind(minDistTheory_ind);
end

% Find minimum gamma and corresponding false and true positive rates
minGAMMA=exp(logGamma(min_pFE_ind));
min_FP=pFP(min_pFE_ind);
min_TP=pTP(min_pFE_ind);
```

```
% print results
fprintf('Theoretical: Gamma=%1.2f, Error=%1.2f%%\n',...
    exp(logGamma_ideal),100*pFE_ideal);
fprintf('Estimated: Gamma=%1.2f, Error=%1.2f%%\n',minGAMMA,100*min_pFE);

% Plot ROC
figure(2);
plot(pFP,pTP,'DisplayName','ROC Curve','LineWidth',2);
hold all;
plot(min_FP,min_TP,'o','DisplayName','Estimated Min. Error','LineWidth',2);
plot(pFP_ideal,pTP_ideal,'+','DisplayName',...
    'Theoretical Min. Error','LineWidth',2);
xlabel('Prob. False Positive');
ylabel('Prob. True Positive');
title('Mininimum Expected Risk ROC Curve');
legend 'show';
grid on; box on;

% Plot Gamma
figure(3);
plot(logGamma,pFE,'DisplayName','Errors','LineWidth',2);
hold on;
plot(logGamma(min_pFE_ind),pFE(min_pFE_ind),...
    'ro','DisplayName','Minimum Error','LineWidth',2);
xlabel('Gamma');
ylabel('Proportion of Errors');
title('Probability of Error vs. Gamma')
grid on;
legend 'show';


%%=========================Part B=========================%%
% Fisher LDA
% Compute scatter matrices
x0=x(:,label==0)';
x1=x(:,label==1)';
mu0_hat=mean(x0);
mu1_hat=mean(x1);
Sigma0_hat=cov(x0);
Sigma1_hat=cov(x1);

% Compute scatter matrices
Sb=(mu0_hat-mu1_hat)*(mu0_hat-mu1_hat)';
Sw=Sigma0_hat+Sigma1_hat;
```

```
% Eigen decompostion to generate WLDA
[V,D]=eig(inv(Sw)*Sb);
[~,ind]=max(diag(D));
w=V(:,ind);
y=w'*x;
w=sign(mean(y(find(label==1))-mean(y(find(label==0)))))*w;
y=sign(mean(y(find(label==1))-mean(y(find(label==0)))))*y;

% Evaluate for different taus
tau=[min(y)-0.1 sort(y)+0.1];
for ind=1:length(tau)
    decision=y>tau(ind);
    Num_pos_LDA(ind)=sum(decision);
    pFP_LDA(ind)=sum(decision==1 & label==0)/Nc(1);
    pTP_LDA(ind)=sum(decision==1 & label==1)/Nc(2);
    pFN_LDA(ind)=sum(decision==0 & label==1)/Nc(2);
    pTN_LDA(ind)=sum(decision==0 & label==0)/Nc(1);
    pFE_LDA(ind)=(sum(decision==0 & label==1)...
        + sum(decision==1 & label==0))/(Nc(1)+Nc(2));
end

% Estimated Minimum Error
[min_pFE_LDA, min_pFE_ind_LDA]=min(pFE_LDA);
minTAU_LDA=tau(min_pFE_ind_LDA);
min_FP_LDA=pFP_LDA(min_pFE_ind_LDA);
min_TP_LDA=pTP_LDA(min_pFE_ind_LDA);

% print results
fprintf('Estimated for LDA: Tau=%1.2f, Error=%1.2f%%\n',...
    minTAU_LDA,100*min_pFE_LDA);

% Plot Fisher LDA Projection
figure(4);
plot(y(label==0),zeros(1,Nc(1)),'o','DisplayName','Label 0');
hold all;
plot(y(label==1),ones(1,Nc(2)),'o','DisplayName','Label 1');
ylim([-1 2]);
plot(repmat(tau(min_pFE_ind_LDA),1,2),ylim,'m--',...
    'DisplayName','Tau for Min. Error','LineWidth',2);
grid on;
xlabel('y');
title('Fisher LDA Projection of Data');
legend 'show';
```

```
% Plot ROC
figure(5);
plot(pFP_LDA,pTP_LDA,'DisplayName','ROC Curve','LineWidth',2);
hold all;
plot(min_FP_LDA,min_TP_LDA,'o','DisplayName',...
    'Estimated Min. Error','LineWidth',2);
xlabel('Prob. False Positive');
ylabel('Prob. True Positive');
title('Mininimum Expected Risk ROC Curve');
legend 'show';
grid on; box on;

% Plot Gamma
figure(6);
plot(tau,pFE_LDA,'DisplayName','Errors','LineWidth',2);
hold on;
plot(tau(min_pFE_ind_LDA),pFE_LDA(min_pFE_ind_LDA),'ro',...
    'DisplayName','Minimum Error','LineWidth',2);
xlabel('Tau');
ylabel('Proportion of Errors');
title('Probability of Error vs. Tau for Fisher LDA')
grid on;
legend 'show';

%%=========================Question 1 Functions=========================%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Functions credit to Prof.Deniz
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function g=evalGaussian(x,mu,Sigma)
%Evaluates the Gaussian pdf N(mu,Sigma) at each coumn of X
[n,N]=size(x);

C=((2*pi)^n*det(Sigma))^(-1/2);%coefficient
E=-0.5*sum((x-repmat(mu,1,N)).*(inv(Sigma)*(x-repmat(mu,1,N))),1);%exponent
g=C*exp(E);%final gaussian evaluationend
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x,labels] = randGMM(N,alpha,mu,Sigma)
d = size(mu,1); % nality of samples
cum_alpha = [0,cumsum(alpha)];
u = rand(1,N); x = zeros(d,N); labels = zeros(1,N);
for m = 1:length(alpha)
    ind = find(cum_alpha(m)<u & u<=cum_alpha(m+1));
```

```
    x(:,ind) = randGaussian(length(ind),mu(:,m),Sigma(:,:,m));
    labels(ind)=m-1;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function x = randGaussian(N,mu,Sigma)
% Generates N samples from a Gaussian pdf with mean mu covariance Sigma
n = length(mu);
z = randn(n,N);
A = Sigma^(1/2);
x = A*z + repmat(mu,1,N);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function gmm = evalGMM(x,alpha,mu,Sigma)
gmm = zeros(1,size(x,2));
for m = 1:length(alpha) % evaluate the GMM on the grid
    gmm = gmm + alpha(m)*evalGaussian(x,mu(:,m),Sigma(:,:,m));
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix code for Question 2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%EECE5644 Fall 2021
% Wang Yinan 001530926 | HW1
%%=========================Question 2=========================%%
% Code help and example from Prof.Deniz
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;close all;clc;

%%=========================Setup=========================%%
n=3; %dimensions
N=10000; %samples

% Class means and covariances
mu(:,1) = [1; 1; 1];
mu(:,2) = [4; 4; 4];
mu3(:,1) = [7; 7; 7];
mu3(:,2) = [11; 11; 11];

Sigma(:,:,1)=[1 0 0; 0 2 0; 0 0 3];
Sigma(:,:,2)=[3 0 0; 0 2 0; 0 0 1];
Sigma3(:,:,1)=[2 0 0; 0 2 0; 0 0 2];
Sigma3(:,:,2)=[2 0 0; 0 2 0; 0 0 2];


% Class priors and true label
prior = [0.3 0.3 0.4];
x=zeros(n,N);
label=zeros(1,N);
for i=1:N
    r=rand(1);
    if r <= 0.3
        label(i)=1;
    elseif (0.3<r)&&(r<=0.6)
        label(i)=2;
    else
        label(i)=3;
    end
end
Nc=[sum(label==1),sum(label==2),sum(label==3)];

% Generate data as prescribed in assignment description
x(:,label==1)=randGMM(Nc(1),1,mu(:,1),Sigma(:,:,1));
```

```
x(:,label==2)=randGMM(Nc(2),1,mu(:,2),Sigma(:,:,2));
x(:,label==3)=randGMM(Nc(3),[0.5 0.5],mu3,Sigma3);

% Plot true class label
figure(7);
X = x(1, label==1);
Y = x(2, label==1);
Z = x(3, label==1);
scatter3(X,Y,Z,'r','filled');
hold on;
X = x(1, label==2);
Y = x(2, label==2);
Z = x(3, label==2);
scatter3(X,Y,Z,'g','filled');
hold on;
X = x(1, label==3);
Y = x(2, label==3);
Z = x(3, label==3);
scatter3(X,Y,Z,'b','filled');
title('Selected Gaussian PDF Samples');
legend('Class 1','Class 2','Class 3');
xlabel('x1');
ylabel('x2');
zlabel('x3');
hold off;

%%=======================Part A=======================%%
% Probabilities and class posteriors
pxgivenl(1,:)=evalGaussian(x,mu(:,1), Sigma(:,:,1));
pxgivenl(2,:)=evalGaussian(x,mu(:,2), Sigma(:,:,2));
pxgivenl(3,:)=evalGMM(x,[0.5 0.5],mu3,Sigma3);
px=prior*pxgivenl;
plgivenx=pxgivenl.*repmat(prior',1,N)./repmat(px,3,1); % Bayes theorem

% 0-1 loss matrix, expected risks, decision
lossMatrix=ones(3,3)-eye(3);
[decision,confusionMatrix]=runClassif(lossMatrix, plgivenx, label, Nc);

% Expected risk
estRisk = expRiskEstimate(lossMatrix, decision, label, N, 3);

% Confusion matrix
conf_mat = [sum(decision(label==1)==1) sum(decision(label==2)==1) sum(decision(la
bel==3)==1); ...
```

```
                sum(decision(label==1)==2) sum(decision(label==2)==2) sum(decision
(label==3)==2);
                sum(decision(label==1)==3) sum(decision(label==2)==3) sum(decision
(label==3)==3)] ./ [sum(label==1) sum(label==2) sum(label==3)];
figure(8)
h = heatmap(conf_mat);
h.Title = 'Confusion Matrix';

% Plot samples with marked correct & incorrect decision
figure(9);
plot3(x(1,label==1&decision==1), ...
    x(2,label==1&decision==1), ...
    x(3,label==1&decision==1), strcat('go'));
axis equal;
hold on;
plot3(x(1,label==1&decision~=1), ...
    x(2,label==1&decision~=1), ...
    x(3,label==1&decision~=1), strcat('ro'));
axis equal;
hold on;
plot3(x(1,label==2&decision==2), ...
    x(2,label==2&decision==2), ...
    x(3,label==2&decision==2), strcat('gx'));
axis equal;
hold on;
plot3(x(1,label==2&decision~=2), ...
    x(2,label==2&decision~=2), ...
    x(3,label==2&decision~=2), strcat('rx'));
axis equal;
hold on;
plot3(x(1,label==3&decision==3), ...
    x(2,label==3&decision==3), ...
    x(3,label==3&decision==3), strcat('gd'));
axis equal;
hold on;
plot3(x(1,label==3&decision~=3), ...
    x(2,label==3&decision~=3), ...
    x(3,label==3&decision~=3), strcat('rd'));
axis equal;
hold on;
grid on
xlabel('x1');ylabel('x2');zlabel('x3');
legend('Class 1 Correct', 'Class 1 Incorrect', 'Class 2 Correct', ...
    'Class 2 Incorrect','Class 3 Correct','Class 3 Incorrect');
```

```
hold off;
title('0-1 loss classification correctness');

%%=======================Part B=======================%%
% Loss matrix A10
lossMatrix10 = [0 1 10; 1 0 10; 1 1 0];
[decision10,confusionMatrix10]=runClassif(lossMatrix10, plgivenx, label, Nc);

% Expected risk 10
estRisk10=expRiskEstimate(lossMatrix10, decision10, label, N, 3);

% Confusion matrix for A10
conf_mat_10 = [sum(decision10(label==1)==1) sum(decision10(label==2)==1) sum(deci
sion10(label==3)==1); ...
               sum(decision10(label==1)==2) sum(decision10(label==2)==2) sum(deci
sion10(label==3)==2);
               sum(decision10(label==1)==3) sum(decision10(label==2)==3) sum(deci
sion10(label==3)==3)] ./ [sum(label==1) sum(label==2) sum(label==3)];
figure(10)
h = heatmap(conf_mat_10);
h.Title = 'Confusion Matrix for A10';

% Plot Risk10 Results
figure(11);
plot3(x(1,label==1&decision==1), ...
    x(2,label==1&decision==1), ...
    x(3,label==1&decision==1), strcat('go'));
axis equal;
hold on;
plot3(x(1,label==1&decision~=1), ...
    x(2,label==1&decision~=1), ...
    x(3,label==1&decision~=1), strcat('ro'));
axis equal;
hold on;
plot3(x(1,label==2&decision==2), ...
    x(2,label==2&decision==2), ...
    x(3,label==2&decision==2), strcat('gx'));
axis equal;
hold on;
plot3(x(1,label==2&decision~=2), ...
    x(2,label==2&decision~=2), ...
    x(3,label==2&decision~=2), strcat('rx'));
axis equal;
hold on;
```

```
plot3(x(1,label==3&decision==3), ...
    x(2,label==3&decision==3), ...
    x(3,label==3&decision==3), strcat('gd'));
axis equal;
hold on;
plot3(x(1,label==3&decision~=3), ...
    x(2,label==3&decision~=3), ...
    x(3,label==3&decision~=3), strcat('rd'));
axis equal;
hold on;
grid on
xlabel('x1');ylabel('x2');zlabel('x3');
legend('Class 1 Correct', 'Class 1 Incorrect', 'Class 2 Correct', ...
    'Class 2 Incorrect','Class 3 Correct','Class 3 Incorrect');
hold off;
title('A10 loss function classification correctness');

% loss matrix A100
lossMatrix100 = [0 1 100; 1 0 100; 1 1 0];
[decision100,confusionMatrix100]=runClassif(lossMatrix100, plgivenx, label, Nc);

% Expected risk 100
estRisk100=expRiskEstimate(lossMatrix100, decision100, label, N, 3);

% Confusion matrix for A100
conf_mat_100 = [sum(decision100(label==1)==1) sum(decision100(label==2)==1) sum(d
ecision100(label==3)==1); ...
                sum(decision100(label==1)==2) sum(decision100(label==2)==2) sum(de
cision100(label==3)==2);
                sum(decision100(label==1)==3) sum(decision100(label==2)==3) sum(de
cision100(label==3)==3)] ./ [sum(label==1) sum(label==2) sum(label==3)];
figure(12)
h = heatmap(conf_mat_100);
h.Title = 'Confusion Matrix for A100';

% Plot Risk100 Results
figure(13);
plot3(x(1,label==1&decision==1), ...
    x(2,label==1&decision==1), ...
    x(3,label==1&decision==1), strcat('go'));
axis equal;
hold on;
plot3(x(1,label==1&decision~=1), ...
    x(2,label==1&decision~=1), ...
```

```
    x(3,label==1&decision~=1), strcat('ro'));
axis equal;
hold on;
plot3(x(1,label==2&decision==2), ...
    x(2,label==2&decision==2), ...
    x(3,label==2&decision==2), strcat('gx'));
axis equal;
hold on;
plot3(x(1,label==2&decision~=2), ...
    x(2,label==2&decision~=2), ...
    x(3,label==2&decision~=2), strcat('rx'));
axis equal;
hold on;
plot3(x(1,label==3&decision==3), ...
    x(2,label==3&decision==3), ...
    x(3,label==3&decision==3), strcat('gd'));
axis equal;
hold on;
plot3(x(1,label==3&decision~=3), ...
    x(2,label==3&decision~=3), ...
    x(3,label==3&decision~=3), strcat('rd'));
axis equal;
hold on;
grid on
xlabel('x1');ylabel('x2');zlabel('x3');
legend('Class 1 Correct', 'Class 1 Incorrect', 'Class 2 Correct', ...
    'Class 2 Incorrect','Class 3 Correct','Class 3 Incorrect');
hold off;
title('A100 loss function classification correctness');

%%=========================Question 2 Functions=========================%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Functions credit to Prof.Deniz
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function r = expRiskEstimate(lossMatrix, decision, label, N, C)
    r = 0;
    for d=1:C
        for l=1:C
            r=r+(lossMatrix(d,l) + sum(decision(label==l)==d));
        end
    end
    r=r/N;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Make decision & confusion matrix
function[decision,confusionMatrix]=runClassif(lossMatrix, classPosteriors, label,
 Nc)
    expRisk=lossMatrix*classPosteriors;
    [~,decision]=min(expRisk,[],1);

    confusionMatrix=zeros(3);
    for l=1:3
        classDecision=decision(label == l);
        for d=1:3
            confusionMatrix(d,l)=sum(classDecision==d)/Nc(l);
        end
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% evalGaussian
function g=evalGaussian(x,mu,Sigma)
    % Evaluates the Gaussian pdf N(mu,Sigma) at each coumn of X
    [n,N] = size(x);
    C = ((2*pi)^n * det(Sigma))^(-1/2);
    E = -0.5*sum((x-repmat(mu,1,N)).*(inv(Sigma)*(x-repmat(mu,1,N))),1);
    g = C*exp(E);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x,labels] = randGMM(N,alpha,mu,Sigma)
d = size(mu,1); % nality of samples
cum_alpha = [0,cumsum(alpha)];
u = rand(1,N); x = zeros(d,N); labels = zeros(1,N);
for m = 1:length(alpha)
    ind = find(cum_alpha(m)<u & u<=cum_alpha(m+1));
    x(:,ind) = randGaussian(length(ind),mu(:,m),Sigma(:,:,m));
    labels(ind)=m-1;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function x = randGaussian(N,mu,Sigma)
% Generates N samples from a Gaussian pdf with mean mu covariance Sigma
n = length(mu);
z = randn(n,N);
A = Sigma^(1/2);
x = A*z + repmat(mu,1,N);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function gmm = evalGMM(x,alpha,mu,Sigma)
```

```
gmm = zeros(1,size(x,2));
for m = 1:length(alpha) % evaluate the GMM on the grid
    gmm = gmm + alpha(m)*evalGaussian(x,mu(:,m),Sigma(:,:,m));
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```