EECE 5644
Assignment 1

Dhanush Balakrishna
Due Date: 2/21/23

Problem 1

To solve this issue, a 10000 sample, 3-dimensional, real-valued random vector called X was created: p(x) = p(x|L=0) P(L=0) + p(x|L=1), where L is the real class label of the PDF that produced the data, p(L=1). If m is the mean vector and C is the covariance matrix, the class conditional PDFs are defined as p(x|L=0) = g(x|mo, Co) and p(x|L=1) = g(x|m1, C1). The distributions' parameters and priors are displayed below. In Figure 1, a plot of the vector X produced by these parameters is displayed.

$$m_0 = \begin{bmatrix} -\frac{1}{2} \\ -\frac{1}{2} \\ -\frac{1}{2} \\ -\frac{1}{2} \end{bmatrix} \quad m_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$P(L=0) = 0.35 \;\; P(L=1) = 0.65$$

$$C_0 = \begin{bmatrix} 2 & -0.5 & 0.3 & 0 \\ -0.5 & 1 & -0.5 & 0 \\ 0.3 & -0.5 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad C_1 = \begin{bmatrix} 1 & 0.3 & -0.2 & 0 \\ 0.3 & 2 & 0.3 & 0 \\ -0.2 & 0.3 & 1 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

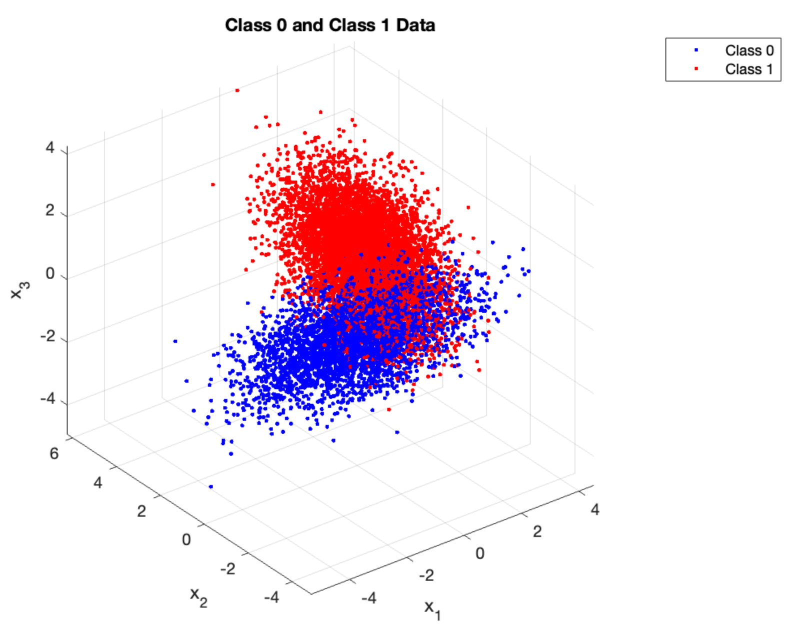

Figure 1: X with True Labels for Problem 1

Part 1: ERM Classification Using Knowledge of True Data

1. Minimum expected risk classification rule in the form of a likelihood-ratio test
   To minimize probability of misclassifications the cost for incorrect classifications should be 1 and the cost for correct classifications should be 0 which results in the gamma equal to 0.538.

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 errors as well as the minimum error determined by a parametric sweep of gamma are marked on the plot.

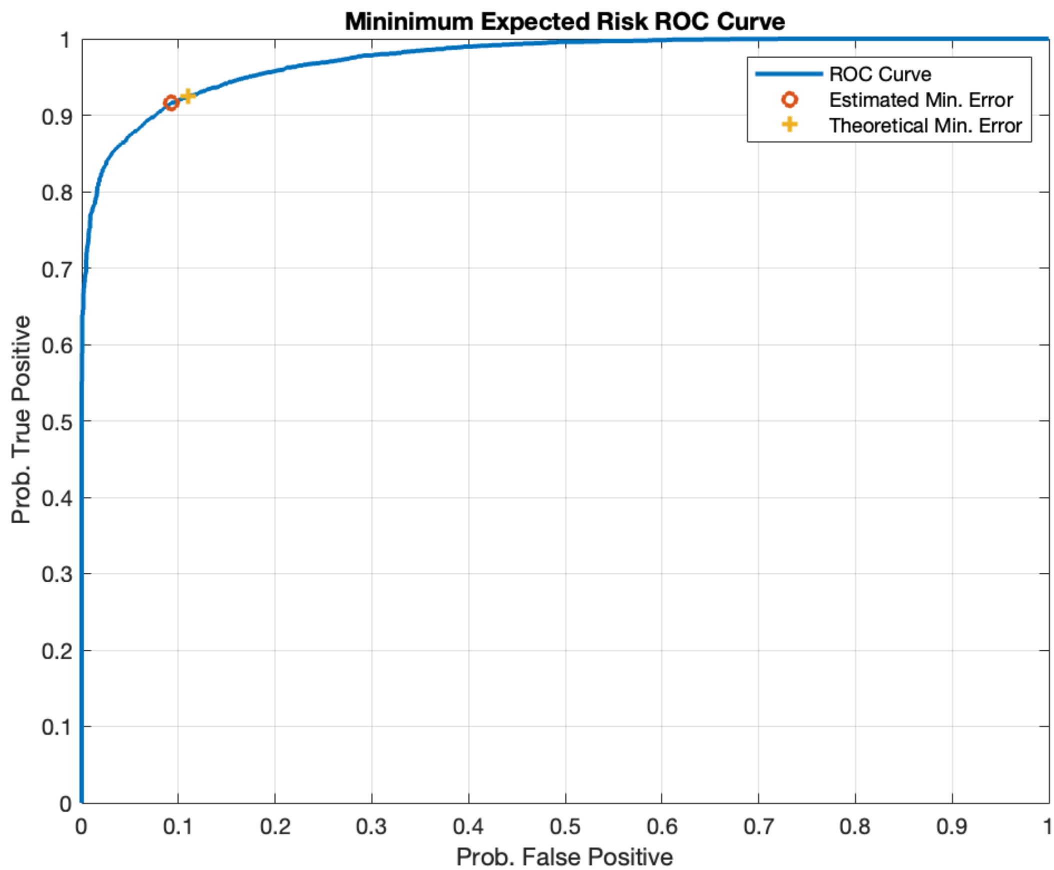

Figure 2: ROC Curve for ERM Classification with Known Data Distributions

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.

	γ	Min. P_{error}
Theoretical	0.54	0.0891
Estimated from Data	0.44	0.0883

Table 1: Comparison of Gammas to find minimum errors.

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 $+\infty$ the probability of error asymptotes to the priors for the two distributions. That is when the γ is set to its minimum value all 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 $+\infty$ all 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.

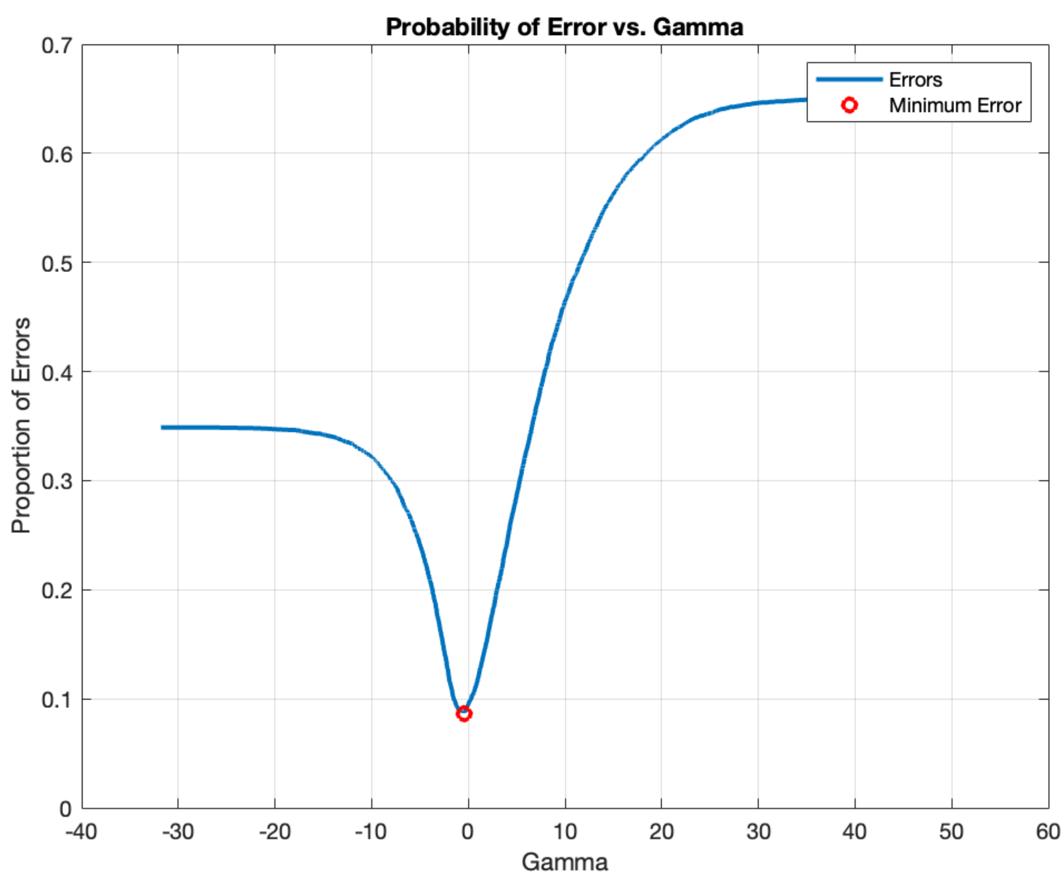

Figure 3: Probability of Error vs ln(Gamma) for ERM Classification

Part B: Naïve Bayesian Classifier

In part two the same data was analyzed using a Naïve Bayesian approach in which the covariance of the data is to be classified is assumed to be the best represented by an identity matrix. Figure 4 below shows a representation of the data being analyzed with the assumed identity covariances. Note that this is just a representation of the assumed data being analyzed while the actual data being analyzed is what is shown in Figure 1. This figure is included to show the difference between the actual and assumed data distributions.

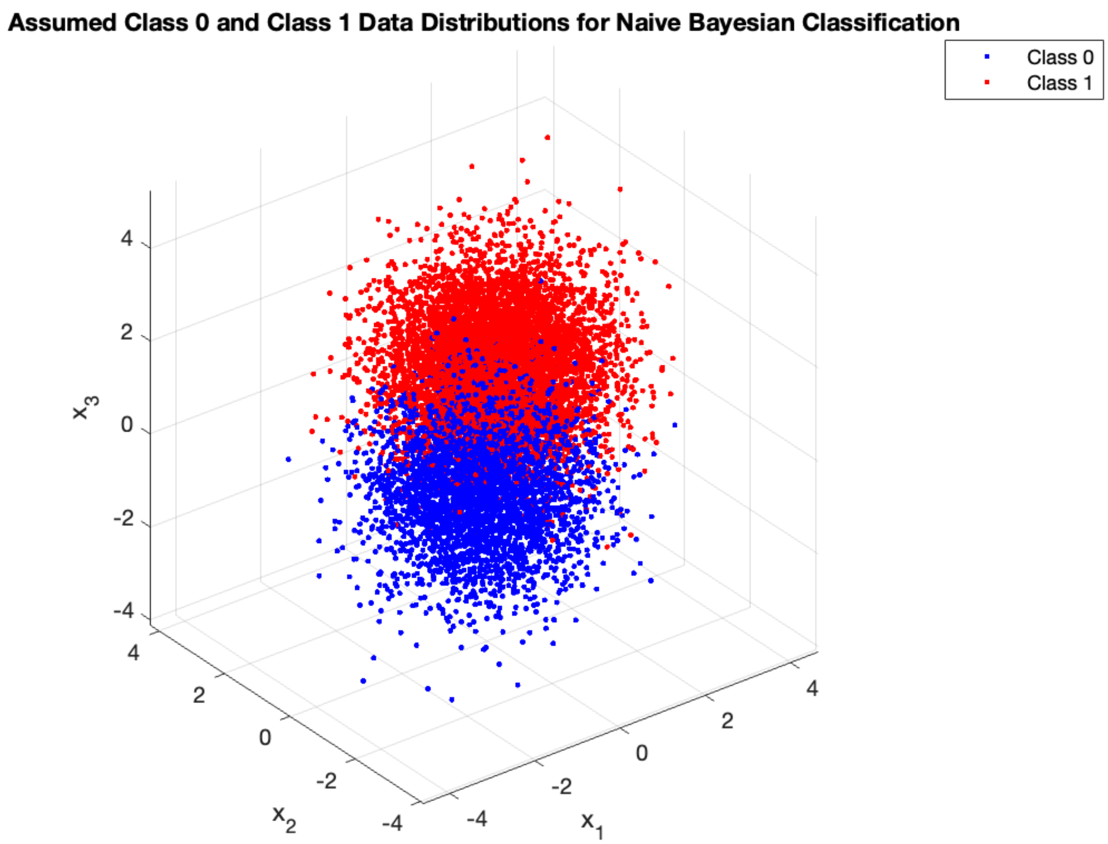

Figure 4: Assumed Distributions for Naïve Bayesian Classification

1. Minimum expected risk classification rule in the form of a likelihood-ratio test. These values are unchanged from the analysis performed in part 1. To minimize the probability of the misclassifications the cost for incorrect classification should be 1 and the cost for correct classifications should be 0 which results in a gamma equal to 0.538.

2. The classifier was implemented for multiple values of gamma and the ROC curve is shown in Figure 5. The location of the minimum error determined by a parametric sweep of gamma is marked on the plot.

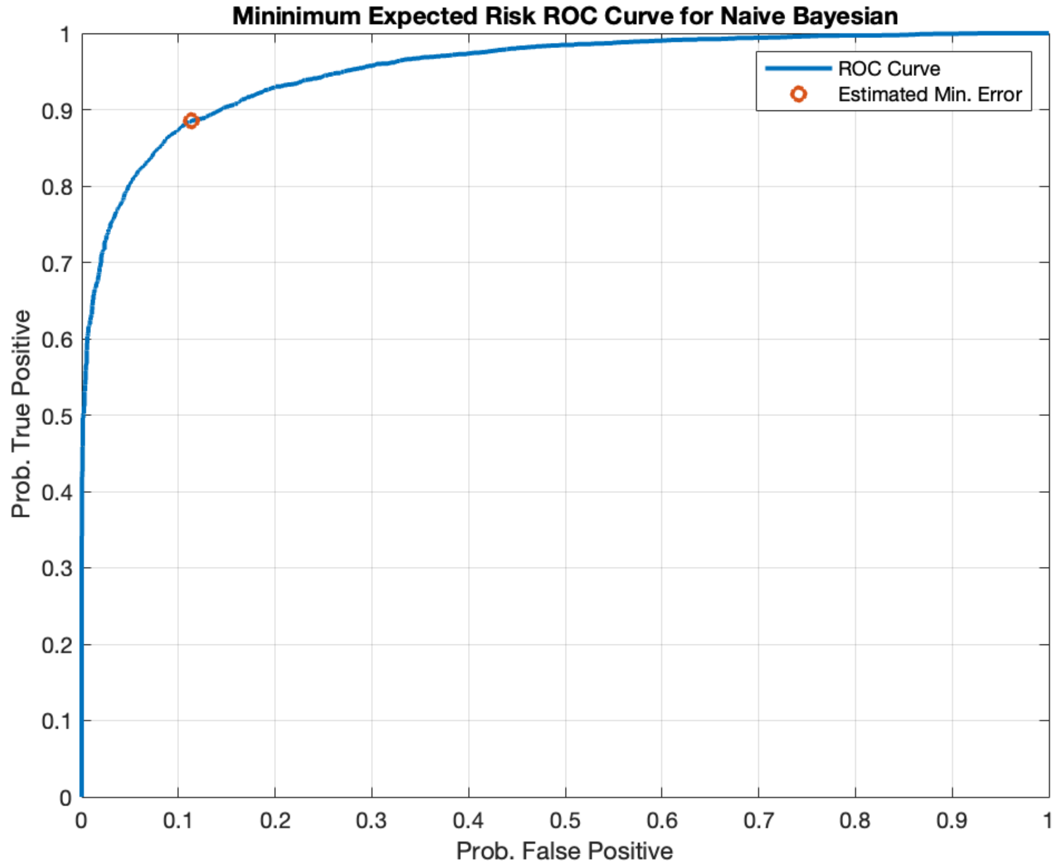

Figure 5: ROC Curve for Naïve Bayesian Classification

3. Table 2 shows a comparison between minimum achievable errors when the covariance of the underlying data is known versus when the covariance is assumed to be best represented by an identity matrix. Both approaches yielded similar results. This is due to the overlap of the two datasets not being significantly altered by the assumption of identity covariance matrices. As can be seen in Figure 6, in both the actual data and the data when represented with identity covariance matrices only the tails overlap. Consequently, the accuracy of classification is only marginally impacted by the lack of the true statistics of the data being classified. In other instances where the means of the data are closer to each other this assumption can result in much worse classification performance.

	γ	Min. P_{error}
Known Covariance	0.44	0.0883
Naïve Bayesian	0.43	0.1089

Table 2: Comparison of Classification Results

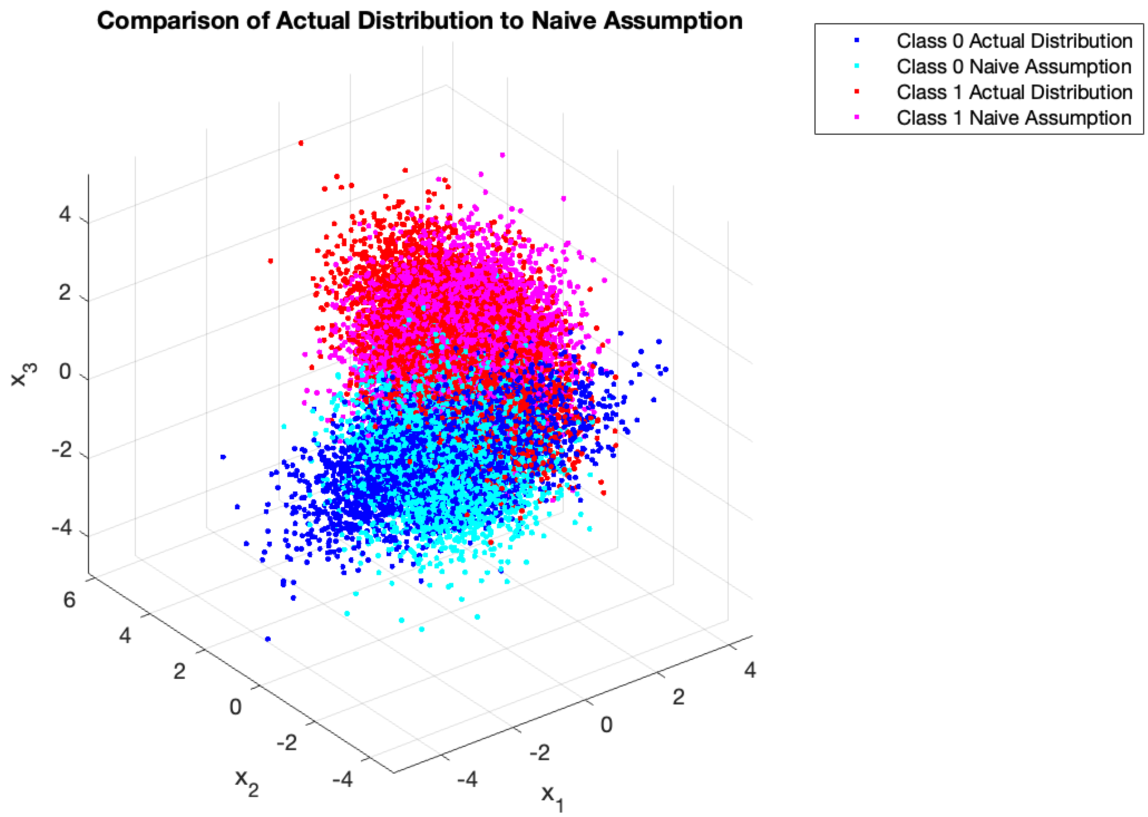

Figure 6: Comparison of Actual Distribution with Naïve Bayesian Assumption

Figure 7 shows the probability of error versus the gamma parameter. The curve has a similar shape to be the one generated in part 1 illustrating the similarity in the two approaches for this set of data and classes.

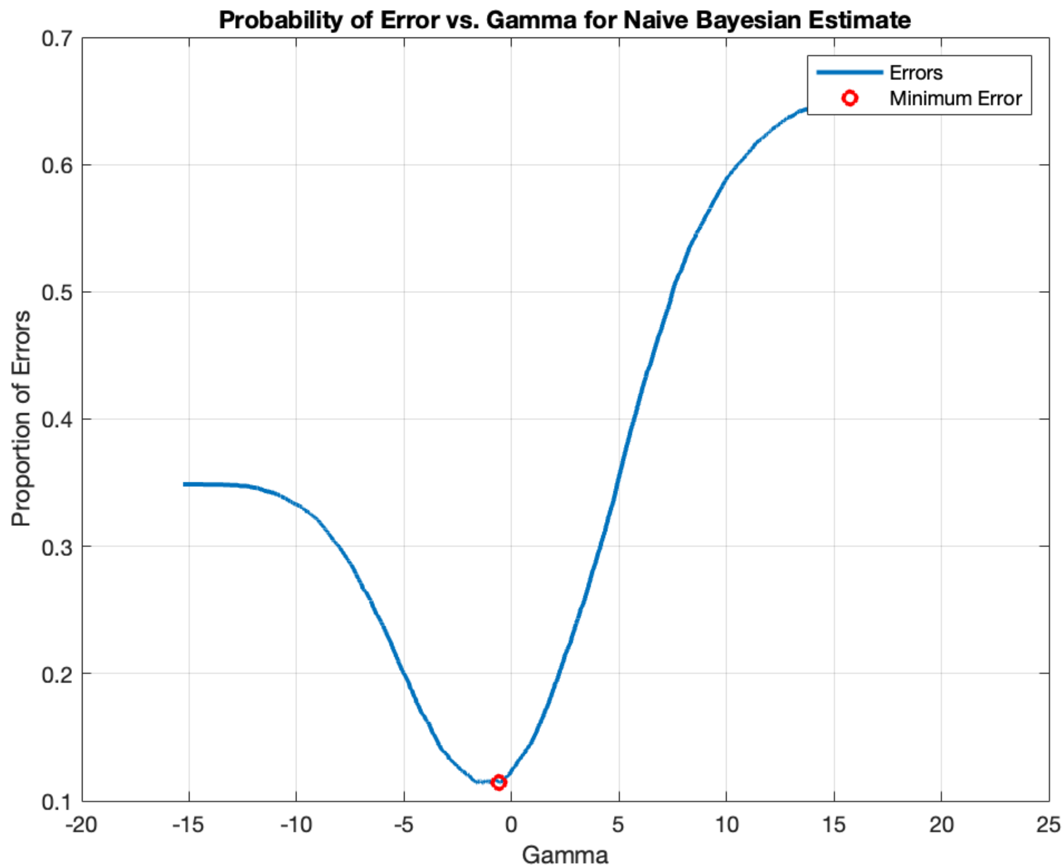

Figure 7: Probability of Error vs. In(Gamma) for Naïve Bayesian Classification

Part C: Fisher Linear Discriminant Analysis (LDA)

In the part 3 classification was performed by using a Fisher LDA approach. In this approach the two datasets being classified are projected into a single dimension. The projection maximizes the ratio of the difference in their means to their variances. This minimizes the overlap between the two datasets aiding the classifications process. In this implementation the analysis was performed using the sample mean and covariances of the two sets of data.

1. Data from projection using Fisher LDA is shown in Figure 8 below. The tau that minimizes the probability of error is also marked. Figure 9 shows the ROC curve. The location of the minimum error determined by a parametric sweep of tau are marked on the plot.

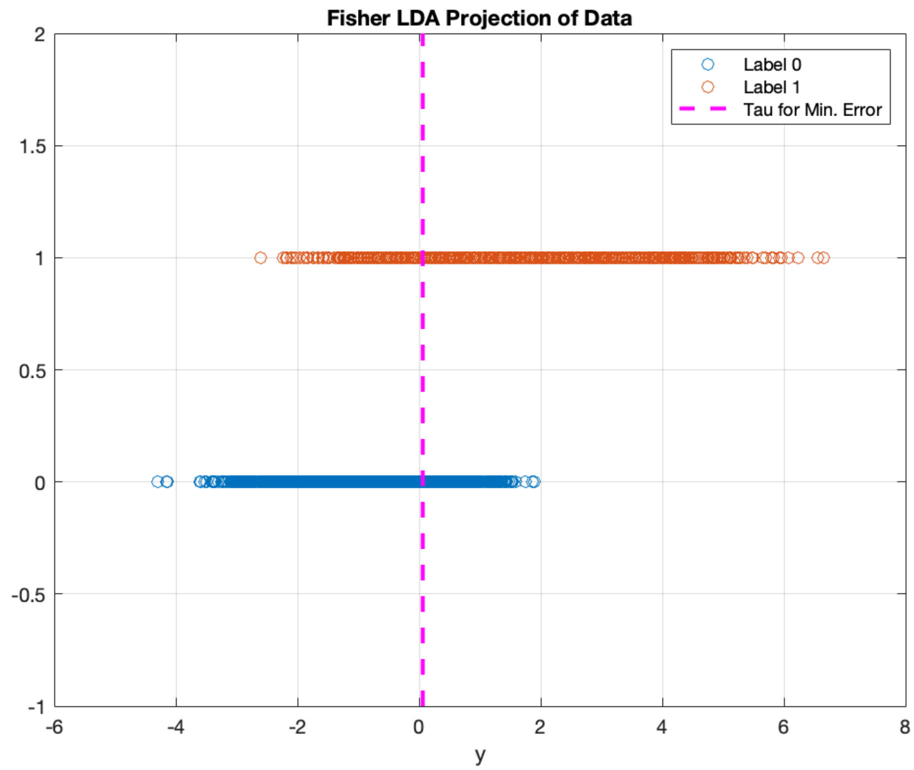

Figure 8: Fisher LDA Projection

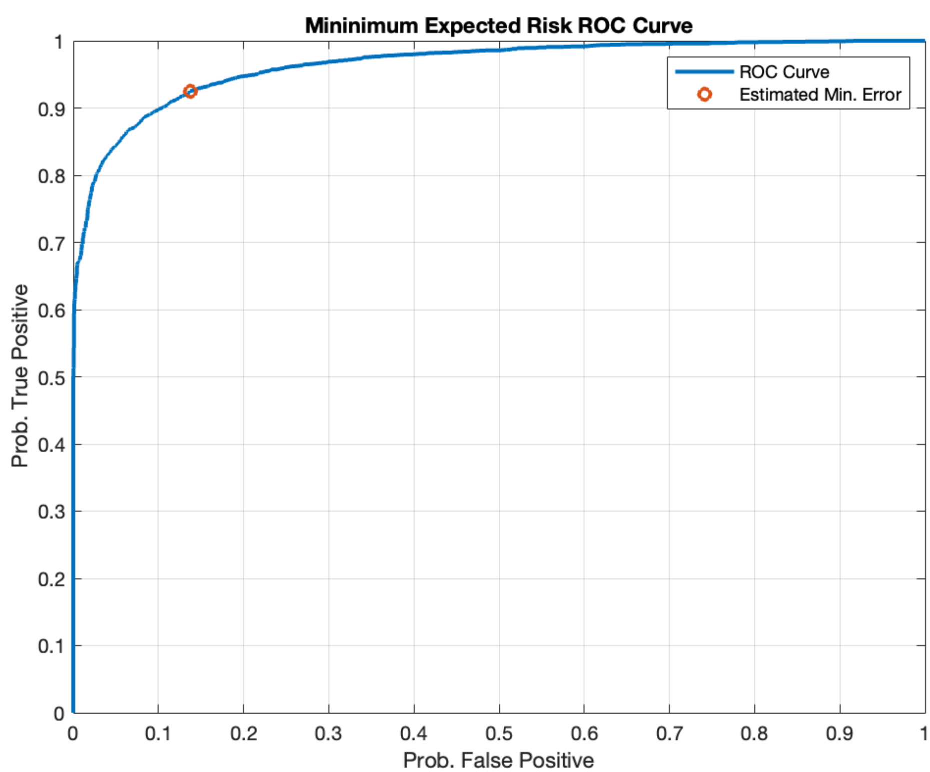

Figure 9: ROC Curve for Fisher LDA

2. A comparison of the probability of errors for the three methods of classification that were performed is shown in Table 3. As can be seen in this table the Fisher LDA performed roughly comparable to the Naïve Bayesian approach. The minimum probability of error was slightly larger than both the theoretical and analytical estimates of probability of error generated using the ERM with known covariances method and was comparable to the results obtained using the Naïve Bayesian approach.

Method	Min. P_{error}
ERM Theoretical	0.0891
ERM Known Covariance	0.0883
ERM Naïve Bayesian	0.1089
Fisher LDA	0.0969

Table 3: Probability of Error for All Classification Methods

Figure 10 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.

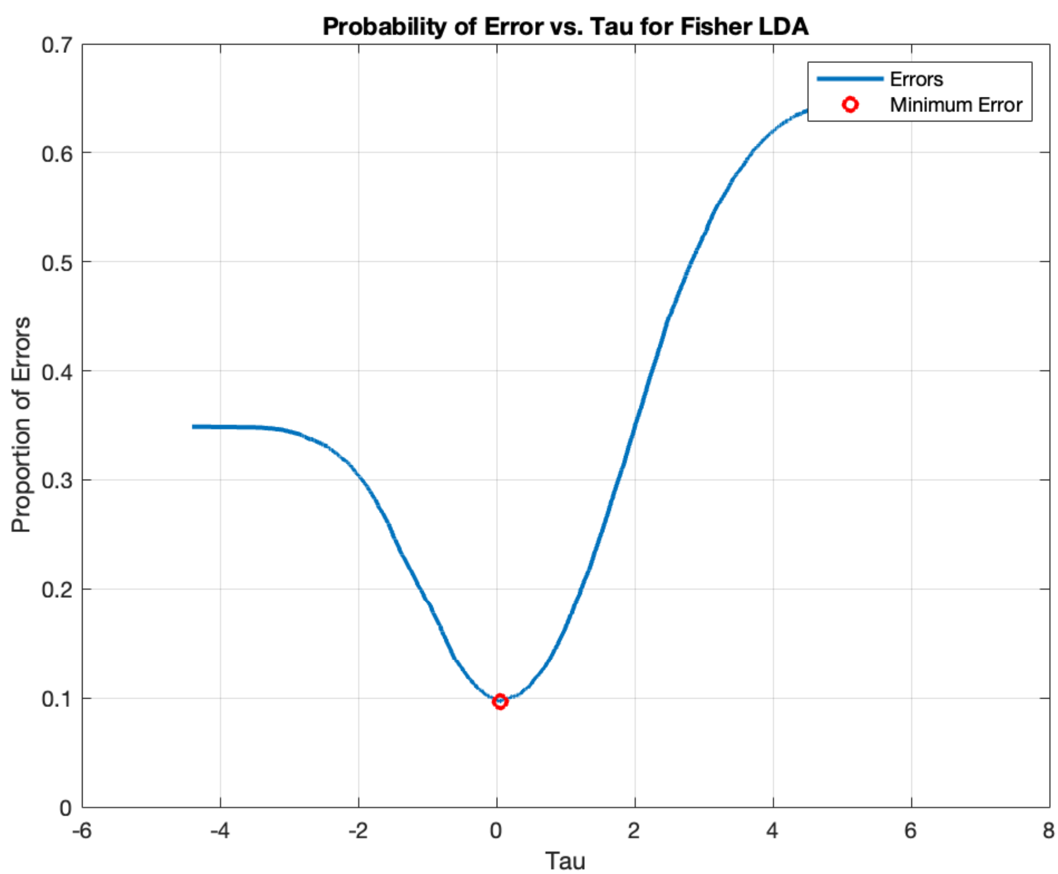

Figure 10: Fisher LDA Probability of Error vs. Tau

Problem 2

The class parameters and priors are as follows:

$$P(X|L=0): P(L=1) = 0.3, \mu = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \; \sigma^2 = \begin{bmatrix} 5 & 1 & 1 \\ 1 & 5 & 1 \\ 1 & 1 & 5 \end{bmatrix}$$

$$P(X|L=0): P(L=2) = 0.3, \mu = \begin{bmatrix} 8 \\ 8 \\ 8 \end{bmatrix} \; \sigma^2 = \begin{bmatrix} 5 & 1 & 1 \\ 1 & 5 & 1 \\ 1 & 1 & 5 \end{bmatrix}$$

$$P(X|L=0): P(L=3) = 0.3, \mu = \begin{bmatrix} 0 \\ 8 \\ 8 \end{bmatrix} \; \sigma^2 = \begin{bmatrix} 5 & 1 & 1 \\ 1 & 5 & 1 \\ 1 & 1 & 5 \end{bmatrix}$$

$$or\ P(X|L=0): P(L=3) = 0.3, \mu = \begin{bmatrix} 8 \\ 0 \\ 8 \end{bmatrix} \; \sigma^2 = \begin{bmatrix} 5 & 1 & 1 \\ 1 & 5 & 1 \\ 1 & 1 & 5 \end{bmatrix}$$

The mean and covariance matrix values given in Question 2 were used to first generate 10000 samples pictured in Figure 11 below.

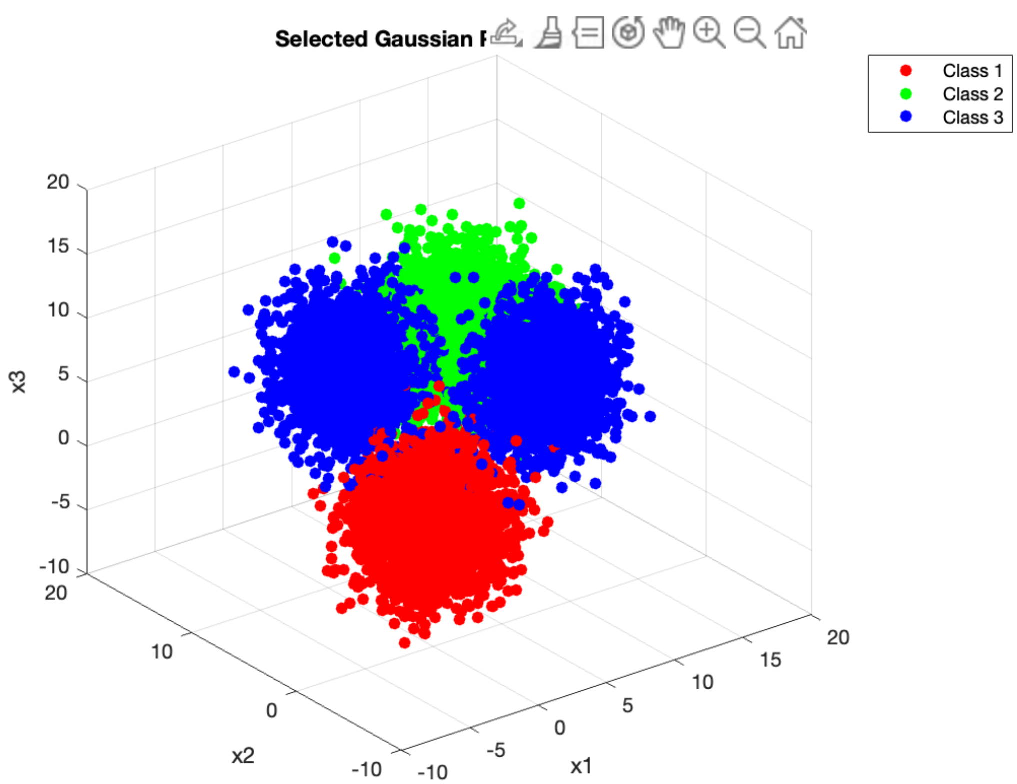

Figure 11: Question 2 Class Distributions

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

$$R(\alpha_i|x) = \sum_{j=1}^{c} \lambda(\alpha_i|w_j).P(w_j|x)$$

$$= \sum_{j \neq i} P(w_j|x)$$

Figure 12 shows the confusion matrix of how the samples were classified according to this decision rule.

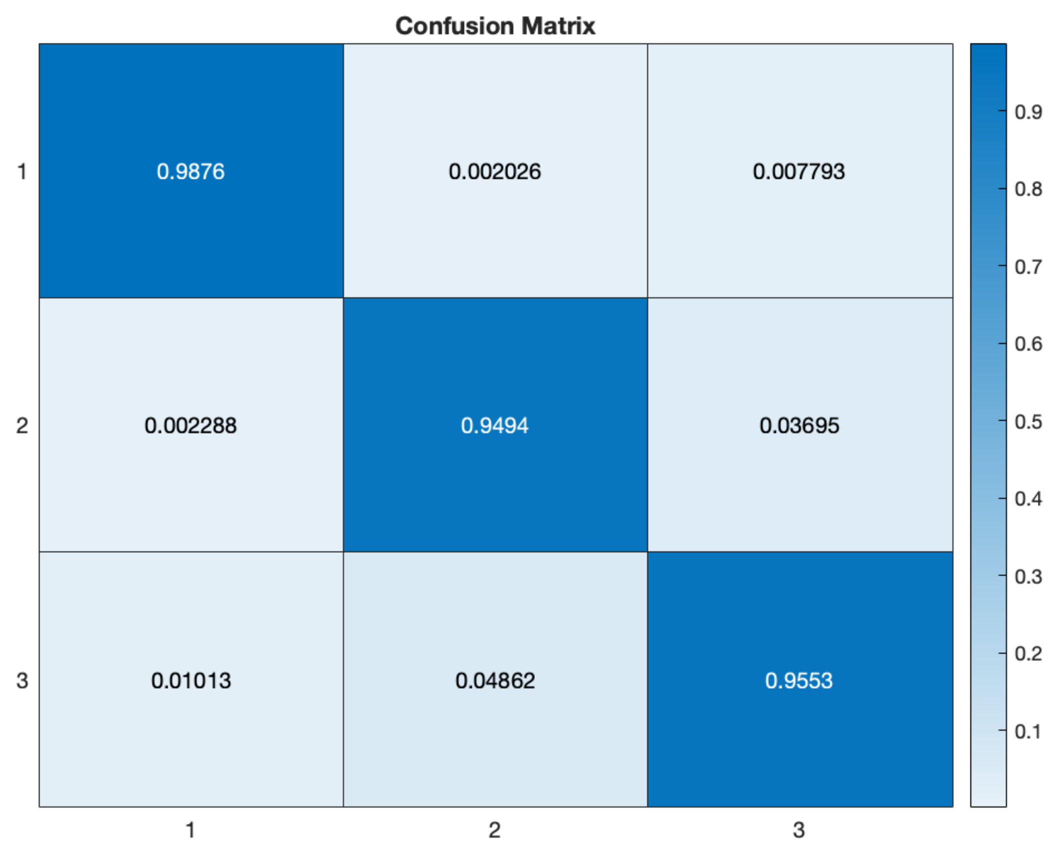

Figure 12: Confusion matrix over the true classification with 0-1 loss

Figure 13 shows the visualization of the data.

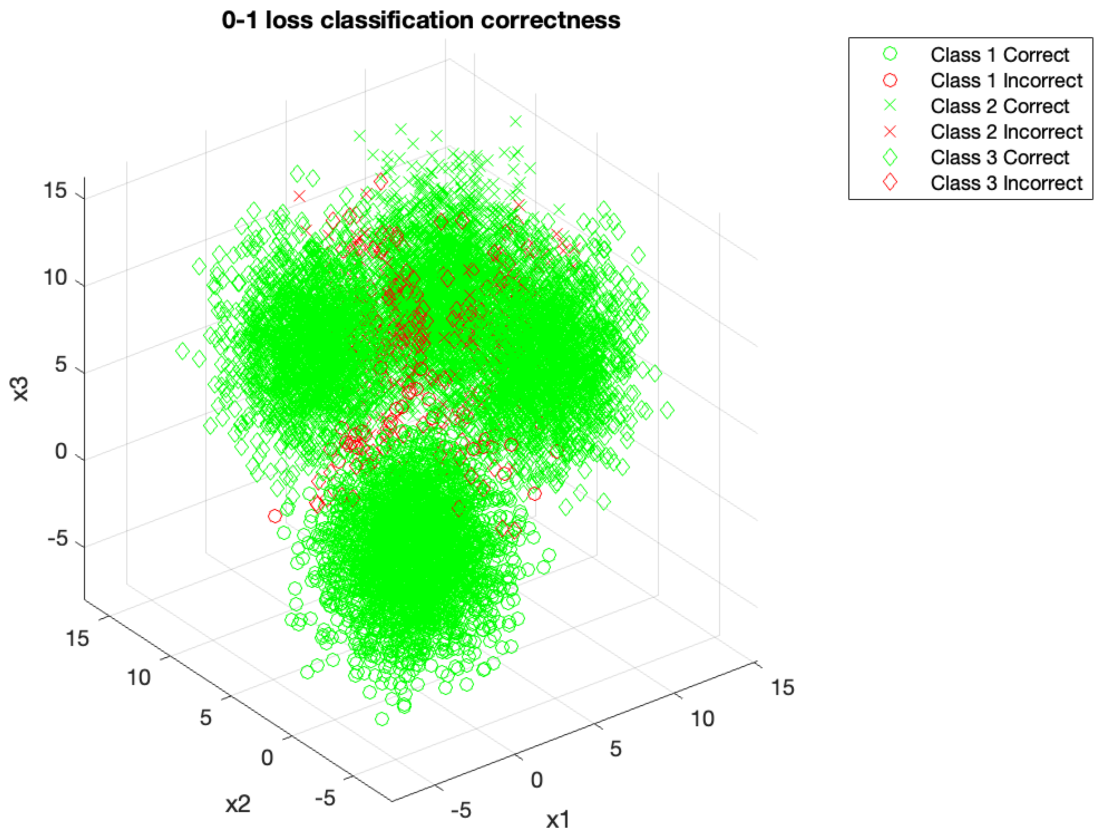

Figure 13: Data Visualization with 0-1 Loss

Part B

1. Decision Rules:

$$R(\alpha_i|x) = \sum_{j=1}^{C} \lambda(\alpha_i|w_j).P(w_j|x)$$

When L=3, Figure 14, and Figure 15 show how the samples were classified according to this decision rule and the confusion matrix when the given decision cares 10 times.

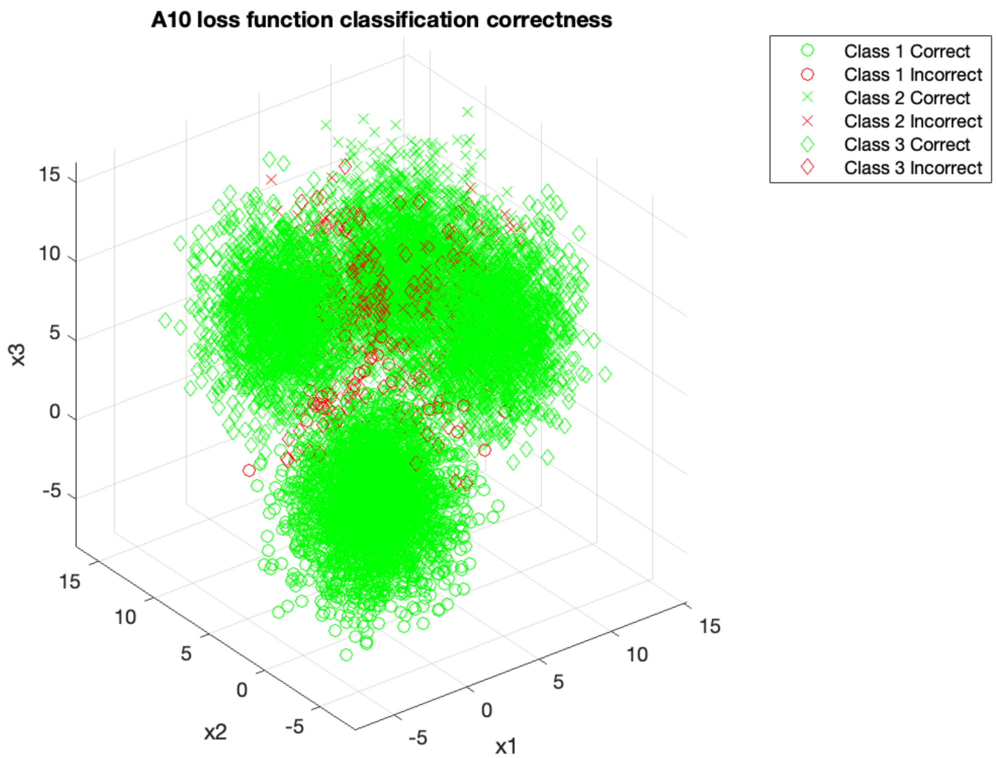

Figure 14: Classification correctness of loss matrix A10

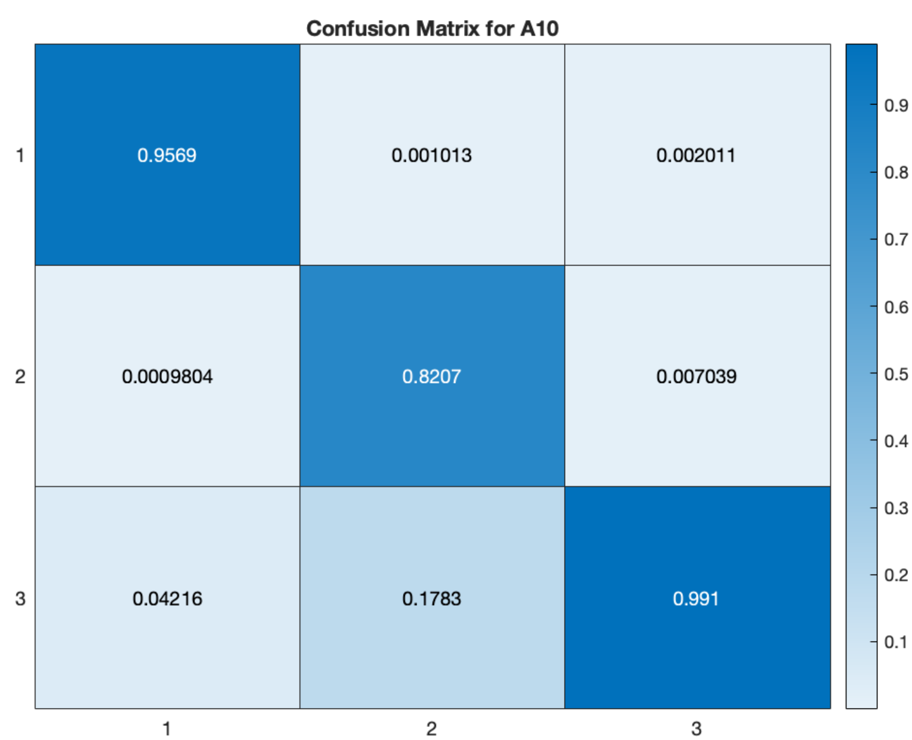

Figure 15: Confusion Matrix of Loss matrix A10

Figure 16 and Figure 17 show how the samples were classified according to this decision rule and the confusion matrix when the given decision rule cares 100 times.

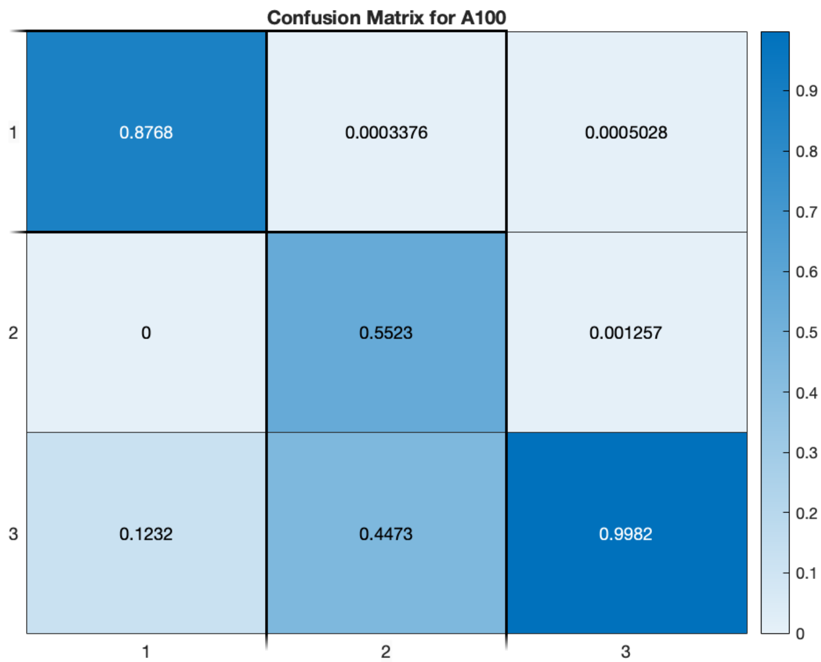

Figure 16: Confusion Matrix of Loss matrix A100

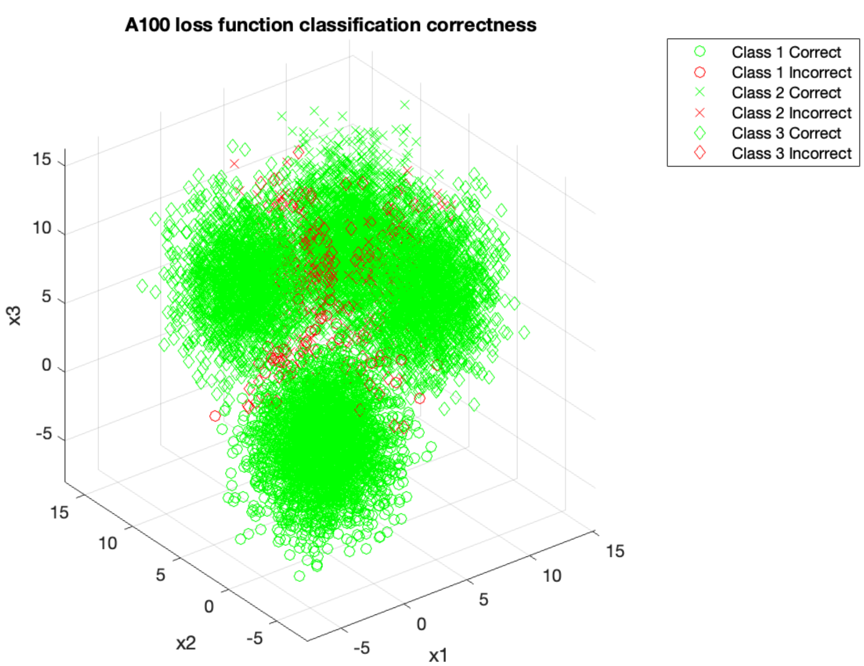

Figure 17: Classification correctness of loss matrix A100

Insights:
The classifier will make more mistakes when the loss matrix is changed and the Class 3 with higher risk is classified. More data will be categorized as Class 3 when the updated matrix is A10 and L=3. The classification accuracy of a single class is displayed in a fusion matrix. The accuracy of Class 3 will not greatly increase when the loss matrix is A100 as opposed to A10. The decisions from categories 1 and 2 will also be added to Class 3 at the same time by the classifier. A significant percentage of data will be classed as Class 3 when A100 is used due to which more inaccurate classifications are observed in Classes 1 and 2.

Problem 3

In this Problem, we can see that the White Wine Dataset is far more classified than the HAR Dataset, whose classification is highly overlapped. We can see the classification with the help of the Fisher LDA plots of both the datasets. (Figure 18 and Figure 19).

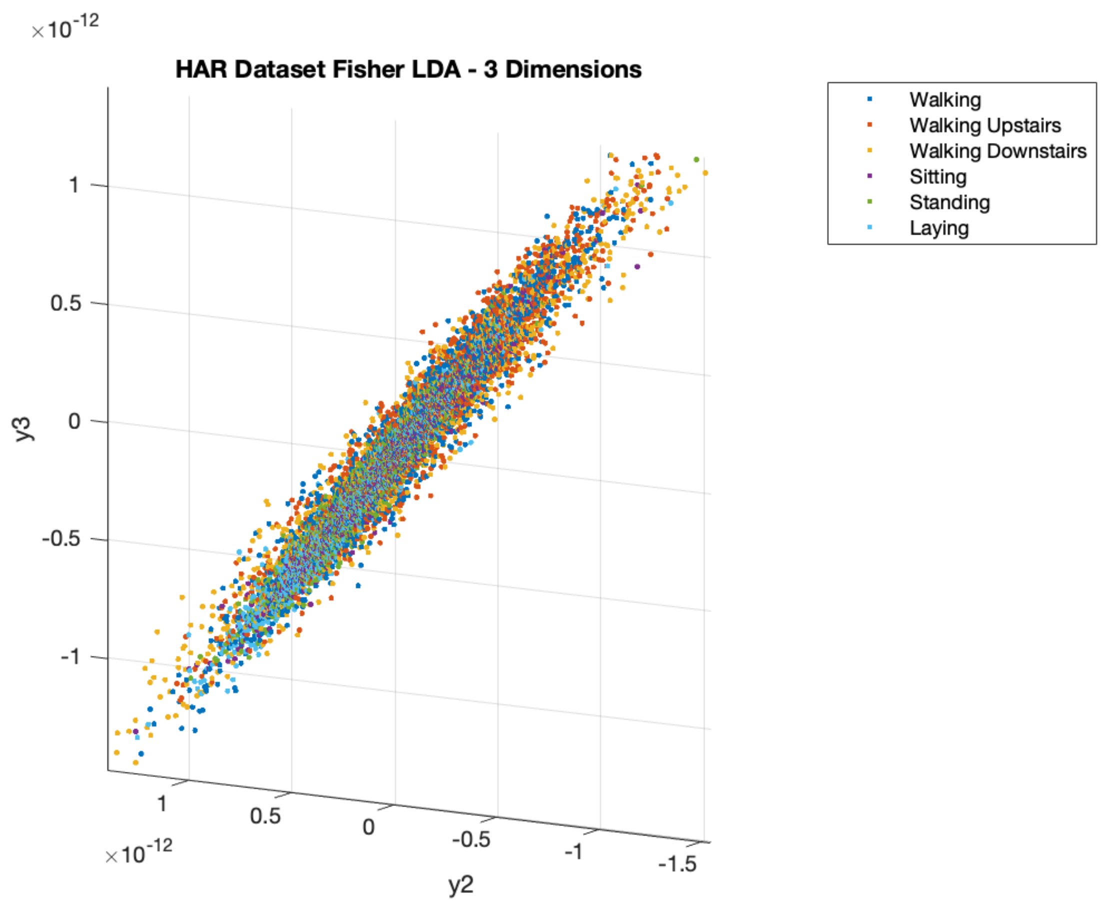

Figure 18: Classification in HAR Dataset

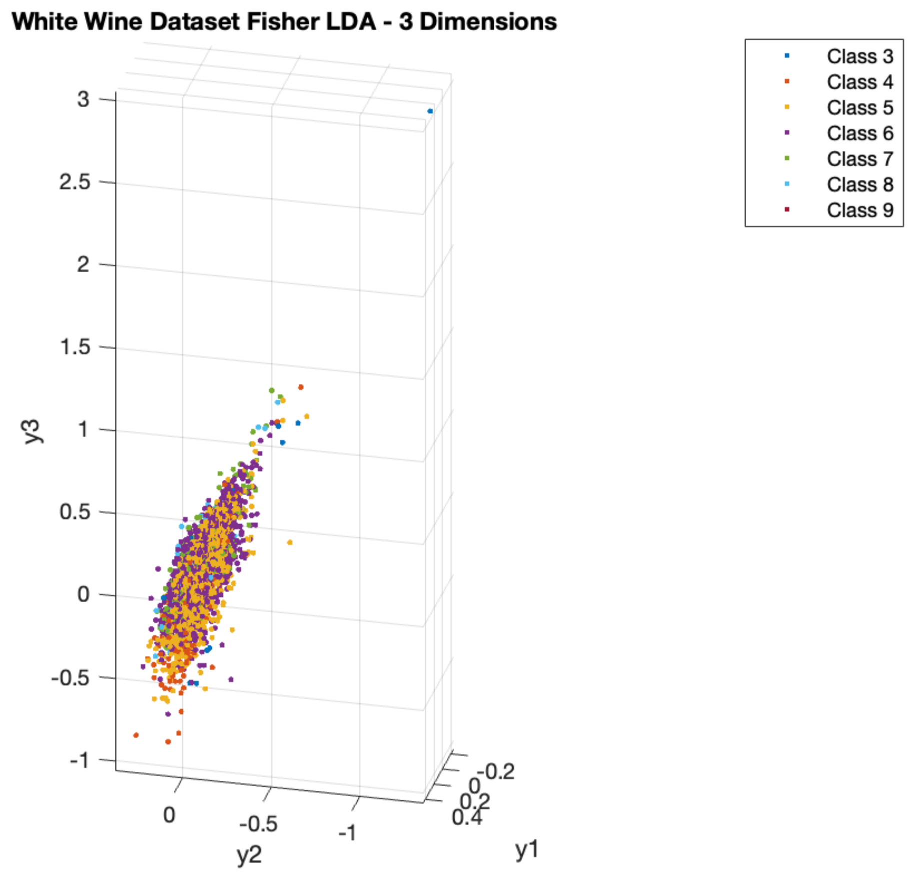

Figure 19: Classification for White Wine Dataset

The Confusion Matrices for both the datasets are as shown below:

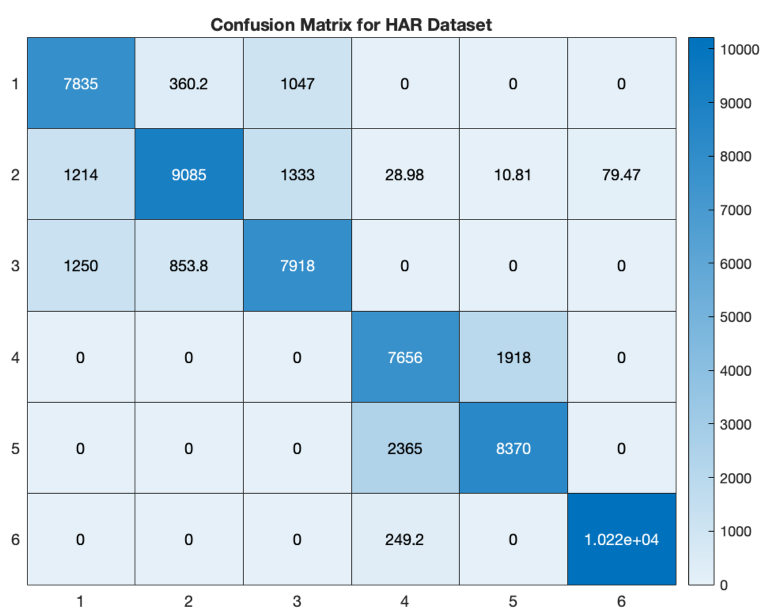

Figure 20: Confusion Matrix for HAR Dataset

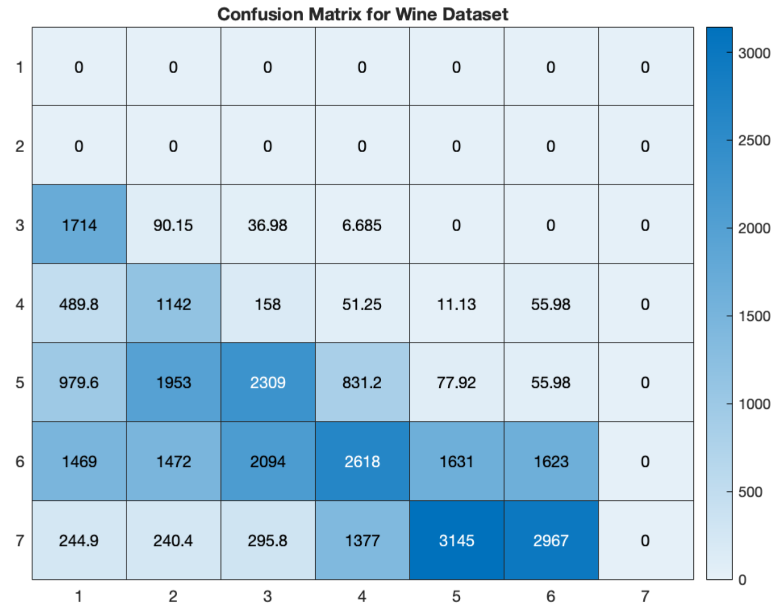

Figure 21: Confusion Matrix for Wine Dataset

The Confusion Matrices clearly depict that the algorithm well classified the Wine dataset than the HAR Dataset.

Appendix:

Codes for all the questions can be found here.

Question 1:

```
clear; close all;
%Initialize Parameters and Generate Data
N = 10000; %Number of data points
n=4; %Dimensions of data
p0 = 0.35; %Prior for label 0
p1 = 0.65; %Prior for label 1
u = rand(1,N)>=p0; %Determine posteriors
%Create appropriate number of data points from each distribution
N0 = length(find(u==0));
N1 = length(find(u==1));
N=N0+N1;
label=[zeros(1,N0) ones(1,N1)];
%Parameters for two classes
mu0 = [-1/2;-1/2;-1/2;-1/2];
Sigma0 = [2,-0.5,0.3,0;
    -0.5,1,-0.5,0;
    0.3,-0.5,1,0;
    0,0,0,2];
mu1 = [1;1;1;1];
Sigma1 = [1,0.3,-0.2,0;
    0.3,2,0.3,0;
    -0.2,0.3,1,0;
    0,0,0,3];
%Generate data as prescribed in assignment description
r0 = mvnrnd(mu0, Sigma0, N0);
r1 = mvnrnd(mu1, Sigma1, N1);
%Plot data showing two classes
figure;
plot3(r0(:,1),r0(:,2),r0(:,3),'.b','DisplayName','Class 0');
axis equal;
hold on;
plot3(r1(:,1),r1(:,2),r1(:,3),'.r','DisplayName','Class 1');
axis equal;
hold on;
xlabel('x_1');ylabel('x_2');zlabel('x_3');
grid on;
title('Class 0 and Class 1 Data');
legend 'show';
%Combine data from each distribution into a single dataset
x=zeros(N,n);
x(label==0,:)=r0;
x(label==1,:)=r1;
%Part 1: ERM Classification with True Knowledge
discScore=log(evalGaussian(x' ,mu1,Sigma1)./evalGaussian(x' ,mu0,Sigma0));
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)/N0;
    pTP(ind)=sum(decision==1 & label==1)/N1;
    pFN(ind)=sum(decision==0 & label==1)/N1;
    pTN(ind)=sum(decision==0 & label==0)/N0;
    %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)*N0 + pFN(ind)*N1)/N;
end
%Calculate Theoretical Minimum Error
logGamma_ideal=log(p0/p1);
decision_ideal=discScore>logGamma_ideal;
pFP_ideal=sum(decision_ideal==1 & label==0)/N0;
pTP_ideal=sum(decision_ideal==1 & label==1)/N1;
pFE_ideal=(pFP_ideal*N0+(1-pTP_ideal)*N1)/(N0+N1);
%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);
%Plot
figure;
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;
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);
figure;
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 2: Naive Bayesian Classifier
Sigma_NB=eye(4); %Assumed covariance
%Generate data to illustrate assumptions
```

```
r0_NB = mvnrnd(mu0, Sigma_NB, N0);
r1_NB = mvnrnd(mu1, Sigma_NB, N1);
%Plot Data demonstrating Naive Assumption
figure;
plot3(r0_NB(:,1),r0_NB(:,2),r0_NB(:,3),'.b','DisplayName','Class 0');
axis equal;
hold on;
plot3(r1_NB(:,1),r1_NB(:,2),r1_NB(:,3),'.r','DisplayName','Class 1');
axis equal;
xlabel('x_1');ylabel('x_2');zlabel('x_3');
grid on;
title('Assumed Class 0 and Class 1 Data Distributions for Naive Bayesian
Classification');
legend 'show';
%Plot comparison of actual data to naive assumption
figure;
plot3(r0(:,1),r0(:,2),r0(:,3),'.b','DisplayName','Class 0 Actual
Distribution');
hold on;
plot3(r0_NB(:,1),r0_NB(:,2),r0_NB(:,3),...
'.c','DisplayName','Class 0 Naive Assumption');
plot3(r1(:,1),r1(:,2),r1(:,3),'.r',...
'DisplayName','Class 1 Actual Distribution');
plot3(r1_NB(:,1),r1_NB(:,2),r1_NB(:,3),...
'.m','DisplayName','Class 1 Naive Assumption');
axis equal;
xlabel('x_1');ylabel('x_2');zlabel('x_3');
grid on;
title('Comparison of Actual Distribution to Naive Assumption');
legend 'show';
%Evaluate for different gammas
discScore_NB=...
log(evalGaussian(x' ,mu1,Sigma_NB)./evalGaussian(x' ,mu0,Sigma_NB));
logGamma_NB=[min(discScore_NB)-0.1 sort(discScore_NB)+0.1];
for ind=1:length(logGamma_NB)
    decision=discScore_NB>logGamma_NB(ind);
    Num_pos_NB(ind)=sum(decision);
    pFP_NB(ind)=sum(decision==1 & label==0)/N0;
    pTP_NB(ind)=sum(decision==1 & label==1)/N1;
    pFN_NB(ind)=sum(decision==0 & label==1)/N1;
    pTN_NB(ind)=sum(decision==0 & label==0)/N0;
    pFE_NB(ind)=(sum(decision==0 & label==1)...
    + sum(decision==1 & label==0))/(N0+N1);
    pFE2_NB(ind)=pFP(ind)*p0+pFN(ind)*p1;
end
%Estimated Minimum Error
[min_pFE_NB, min_pFE_ind_NB]=min(pFE_NB);
minGAMMA_NB=exp(logGamma(min_pFE_ind_NB));
min_FP_NB=pFP_NB(min_pFE_ind_NB);
min_TP_NB=pTP_NB(min_pFE_ind_NB);
%Plot Results
figure;
plot(pFP_NB,pTP_NB,'DisplayName','ROC Curve','LineWidth',2);
hold all;
plot(min_FP_NB,min_TP_NB,'o','DisplayName',...
```

```
'Estimated Min. Error','LineWidth',2);
xlabel('Prob. False Positive');
ylabel('Prob. True Positive');
title('Mininimum Expected Risk ROC Curve for Naive Bayesian');
legend 'show';
grid on; box on;
figure;
plot(logGamma_NB,pFE_NB,'DisplayName','Errors','LineWidth',2);
hold on;
plot(logGamma_NB(min_pFE_ind_NB),pFE_NB(min_pFE_ind_NB),'ro',...
'DisplayName','Minimum Error','LineWidth',2);
xlabel('Gamma');
ylabel('Proportion of Errors');
title('Probability of Error vs. Gamma for Naive Bayesian Estimate')
grid on;
legend 'show';
fprintf('Estimated for NB: Gamma=%1.2f, Error=%1.2f%%\n',...
minGAMMA_NB,100*min_pFE_NB);
%Part 3: Fisher LDA
%Compute Sample Mean and covariances
mu0_hat=mean(r0)';
mu1_hat=mean(r1)';
Sigma0_hat=cov(r0);
Sigma1_hat=cov(r1);
%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(label==1)-mean(y(label==0))))*w;
y=sign(mean(y(label==1)-mean(y(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)/N0;
    pTP_LDA(ind)=sum(decision==1 & label==1)/N1;
    pFN_LDA(ind)=sum(decision==0 & label==1)/N1;
    pTN_LDA(ind)=sum(decision==0 & label==0)/N0;
    pFE_LDA(ind)=(sum(decision==0 & label==1)...
    + sum(decision==1 & label==0))/(N0+N1);
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);
%Plot results
figure;
plot(y(label==0),zeros(1,N0),'o','DisplayName','Label 0');
hold all;
```

```
plot(y(label==1),ones(1,N1),'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';
figure;
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;
figure;
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';
fprintf('Estimated for LDA: Tau=%1.2f, Error=%1.2f%%\n',...
minTAU_LDA,100*min_pFE_LDA);
%% ======================= Question 1: Functions ====================== %%
%reference g/Code/evalGaussian.m
function g = evalGaussian(x ,mu,Sigma)
%Evaluates the Gaussian pdf N(mu, Sigma ) at each column 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); %finalgaussianevaluation
end
```

Question 2:

```
clear;close all;clc;
n=3; %dimensions
N=10000; %samples

% Class means and covariances
mu(:,1) = [0; 0; 0];
mu(:,2) = [8; 8; 8];
mu3(:,1) = [0; 8; 8];
mu3(:,2) = [8; 0; 8];

Sigma(:,:,1)=[5 1 1; 1 5 1; 1 1 5];
Sigma(:,:,2)=[5 1 1; 1 5 1; 1 1 5];
Sigma3(:,:,1)=[5 1 1; 1 5 1; 1 1 5];
```

```
Sigma3(:,:,2)=[5 1 1; 1 5 1; 1 1 5];


% 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(11);
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(label==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(12)
h = heatmap(conf_mat);
h.Title = 'Confusion Matrix';

% Plot samples with marked correct & incorrect decision
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('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(decision10(label==3)==1); ...
               sum(decision10(label==1)==2) sum(decision10(label==2)==2)
sum(decision10(label==3)==2);
               sum(decision10(label==1)==3) sum(decision10(label==2)==3)
sum(decision10(label==3)==3)] ./ [sum(label==1) sum(label==2) sum(label==3)];
figure(14)
h = heatmap(conf_mat_10);
h.Title = 'Confusion Matrix for A10';

% Plot Risk10 Results
figure(15);
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(decision100(label==3)==1); ...
               sum(decision100(label==1)==2) sum(decision100(label==2)==2)
sum(decision100(label==3)==2);
               sum(decision100(label==1)==3) sum(decision100(label==2)==3)
sum(decision100(label==3)==3)] ./ [sum(label==1) sum(label==2)
sum(label==3)];
figure(16);
h = heatmap(conf_mat_100);
h.Title = 'Confusion Matrix for A100';

% Plot Risk100 Results
figure(17);
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
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Question 3:

```
clearvars; close all; clear;

% Initial parameters
lambda=0.00001; %for regularization

% White Wine Quality Dataset
wine_raw_data = dlmread('winequality-white.csv',';',1,0)';

%Separate dataset into data and true class labels
x_W=wine_raw_data(1:end-1,:);
label_W=wine_raw_data(end,:);

% HAR Dataset (true class labels are separate files so no need to separate)
load X_train.txt;
load X_test.txt;
x_H=vertcat(X_train,X_test)';

%load true class labels
load y_train.txt;
load y_test.txt;
label_H=vertcat(y_train, y_test)';

% Dimensions and size of datasets from matrices
[n_W, N_W]=size(x_W);
[n_H,N_H]=size(x_H);

% Class labels and number of classes
class_W=unique(label_W);
C_W=length(class_W);
class_H=unique(label_H);
C_H=length(class_H);

% Class priors
priors_W=zeros(1,C_W);
for l=1:C_W
```

```
    priors_W(l)=sum(label_W==class_W(l))/N_W;
end
priors_H=zeros(1,C_H);
for l=1:C_H
    priors_H(l)=sum(label_H == class_H(l))/N_H;
end

display(priors_W);
display(priors_H);

% Estimated mean vectors
mu_W=zeros(n_W, C_W);
for l=1:C_W
    samples=x_W(:,label_W == class_W(l));
    mu_W(:,l)=mean(samples, 2);
end
mu_H=zeros(n_H, C_H);
for l=1:C_H
    samples=x_H(:,label_H == class_H(l));
    mu_H(:,l)=mean(samples, 2);
end

display(mu_W);
display(mu_H);

% Estimated covariance matrices
Sigma_W=zeros(n_W, n_W, C_W);
for l=1:C_W
    Sigma_W(:,:,l)=cov(x_W(:,label_W==class_W(l))')+(lambda*eye(n_W));
end
Sigma_H = zeros(n_H, n_H, C_H);
for l=1:C_H
    Sigma_H(:,:,l)=cov(x_H(:,label_H==class_H(l))')+(lambda*eye(n_H));
end

display(Sigma_W);
display(Sigma_H);
%[QW,DW]=eig(Sigma_W(:,:,1));
%[QH,DH]=eig(Sigma_H(:,:,1));

% Class posteriors and loss matrices
classPosterior_W=classPosterior(x_W, mu_W, Sigma_W, N_W, C_W, priors_W);
classPosterior_H=classPosterior_Mvnpdf(x_H, mu_H, N_H, C_H, priors_H);

lossMatrix_W=zeros(C_W);
 for i=1:C_W
    for j=1:C_W
        lossMatrix_W(i,j) = abs(i-j);
    end
 end
lossMatrix_H=ones(C_H,C_H)-eye(C_H);

% Run classification for confusion matrices and create pError
[decisions_W,confusionMatrix_W]=runClassif(lossMatrix_W, classPosterior_W,
label_W, class_W, priors_W);
```

```
[decisions_H,confusionMatrix_H]=runClassif(lossMatrix_H, classPosterior_H,
label_H, class_H, priors_H);
pError_W=calculatePErr(confusionMatrix_W, priors_W);
pError_H=calculatePErr(confusionMatrix_H, priors_H);
y_W=LDA(x_W',label_W)';
y_H=LDA(x_H',label_H)';

%Plot first 3 dimensions of LDAprojections on each dataset
figure(1);
for l=1:C_W
    scatter3(y_W(1, label_W==class_W(l)), y_W(2, label_W==class_W(l)), y_W(3,
label_W==class_W(l)), '.');
    hold on;
    axis equal;
end
title('White Wine Dataset Fisher LDA - 3 Dimensions');
xlabel('y1'); ylabel('y2'); zlabel('y3');
legend('Class 3', 'Class 4', 'Class 5', 'Class 6', 'Class 7', 'Class 8',
'Class 9');

figure(2);
for l=1:C_H
    scatter3(y_H(1, label_H==class_H(l)), y_H(2, label_H==class_H(l)), y_H(3,
label_H==class_H(l)), '.');
    hold on;
    axis equal;
end
title('HAR Dataset Fisher LDA - 3 Dimensions');
xlabel('y1'); ylabel('y2'); zlabel('y3');
legend('Walking', 'Walking Upstairs', 'Walking Downstairs', 'Sitting',
'Standing', 'Laying')

%Confusion Matrices
figure(3);
h1=heatmap(confusionMatrix_W);
h1.Title='Confusion Matrix for Wine Dataset';

figure(4);
h2=heatmap(confusionMatrix_H);
h2.Title='Confusion Matrix for HAR Dataset';

%given a confusion matrix and corresponding class priors calculate
%probability of error for the classifier
function pErr=calculatePErr(confusionMatrix, prior)
    C=length(prior);
    pErr=0;
    for l=1:C
        for d=1:C
            if d~=l
                pErr=pErr+confusionMatrix(d,l) * prior(l);
            end
        end
    end
end
```

```
% Class posterior for samples
function p=classPosterior(x, mu, Sigma, N, C, priors)
    for l=1:C
        pxgivenl(l,:)=evalGaussian(x,mu(:,l),Sigma(:,:,l)');
    end
    px=priors*pxgivenl;
    p=pxgivenl.*repmat(priors',1,N)./repmat(px,C,1);
end

% Class posterior for HAR dataset using mvnpdf and omitting sigma, since the
large covariance
% matrices cause issues
function p=classPosterior_Mvnpdf(x, mu, N, C, priors)
    for l=1:C
        pxgivenl(l,:)=mvnpdf(x',mu(:,l)');
    end
    px=priors*pxgivenl;
    p=pxgivenl.*repmat(priors',1,N)./repmat(px,C,1);
end

% Make decisions and confusion matrix
function[decision, confusionMatrix]=runClassif(lossMatrix, classPosterior,
label, class, labelCount)
    C=length(class);
    expRisk=lossMatrix*classPosterior;
    [~,decisionInds]=min(expRisk,[],1);
    decision=class(decisionInds);

    confusionMatrix=zeros(C);
    for l=1:C
        classDecision=decision(label==class(l));
        for d=1:C
            confusionMatrix(d,l)=sum(classDecision==d)/labelCount(l);
        end
    end
end

% evalGaussian
% credit to Prof. Deniz's shared code file
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

% LDA
% credit online
function Y = LDA(X,L)
    Classes=unique(L)';
    k=numel(Classes);
    n=zeros(k,1);
    C=cell(k,1);
    M=mean(X);
```

```
    S=cell(k,1);
    Sw=0;
    Sb=0;
    for j=1:k
        Xj=X(L==Classes(j),:);
        n(j)=size(Xj,1);
        C{j}=mean(Xj);
        S{j}=0;
        for i=1:n(j)
            S{j}=S{j}+(Xj(i,:)-(C{j})'*(Xj(i,:)-C{j}));
        end
        Sw=Sw+S{j};
        Sb=Sb+n(j)*(C{j}-M)'*(C{j}-M);
    end
    [W, LAMBDA]=eig(Sb,Sw);
    lambda=diag(LAMBDA);
    [~, SortOrder]=sort(lambda,'descend');
    W=W(:,SortOrder);
    Y=X*W;
end
```

Citations:
Codes and notes by Prof. Deniz Erdogmus, juliangp98 (GitHub), emilyjcosta(GitHub), the internet.