IE 6318 Data Mining and Analytics

Homework 4

Classification Using Fisher Linear Discriminant and Perform ROC Analysis

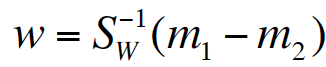
Submitted by

Sarika Kshatriya

UTA ID:1001483313

[Sarika.kshatriya@mavs.uta.edu](mailto:Sarika.kshatriya@mavs.uta.edu)

1. Make a classification function based on Fisher Linear Discriminant. The optimal projection direction *w* is:

.

One can perform classification on the one-dimensional space for the projected data samples *wtx.*

**Matlab Code:**

function [Lpred] = FishersLDA(Dtrain, Ltrain, Dtest, lambda, option)

% Binary Classification using Fisher's linear discriminant

% 1. Find the optimal data projection direction w using the training dataset

% 2. And use Fisher's linear discriminant to make classification for testing dataset

% Note: the class labels use 1 and -1 to represent the two classes to be classified.

% If no input for lambda, assign a defualt matrix to lambda

if (nargin<4 || isempty(lambda))

lambda = [0 1; 1 0];

end

%--------Fisher Linear Discriminant---------%

idx1 = find(Ltrain==1); % the index for class 1

idx2 = find(Ltrain==-1); % the index for class -1

Dtrain\_c1 = Dtrain(idx1, :); % the training samples of class 1

Dtrain\_c2 = Dtrain(idx2, :); % the training samples of class -1

N\_c1 = length(idx1); % the number of samples in class 1

N\_c2 = length(idx2); % the number of samples in class -1

sigma1 = cov(Dtrain\_c1);

mu1 = mean(Dtrain\_c1);

sigma2 = cov(Dtrain\_c2);

mu2 = mean(Dtrain\_c2);

Sw = sigma1 + sigma2;

%% The optimal direction w for sample projection:

w = inv(Sw)\*(mu1-mu2)';

%------The Projected Data-------%

Dtrain\_new = Dtrain\*w; % Projected training data

Ltrain\_new = Ltrain; % Training data label

Dtest\_new = Dtest\*w; % Projected testing data

Dtrain\_new\_c1 = Dtrain\_new(idx1, :); % projected training samples of class 1

Dtrain\_new\_c2 = Dtrain\_new(idx2, :); % projected training samples of class -1

mu1\_new = mean(Dtrain\_new\_c1); % mean of projected samples of class 1

mu2\_new = mean(Dtrain\_new\_c2); % mean of projected samples of class -1

% sigma1\_new = cov(Dtrain\_new\_c1);

% sigma2\_new = cov(Dtrain\_new\_c2);

sigma1\_new = std(Dtrain\_new\_c1); % standard deviation of projected samples of class 1

sigma2\_new = std(Dtrain\_new\_c2); % standard deviation of projected samples of class -1

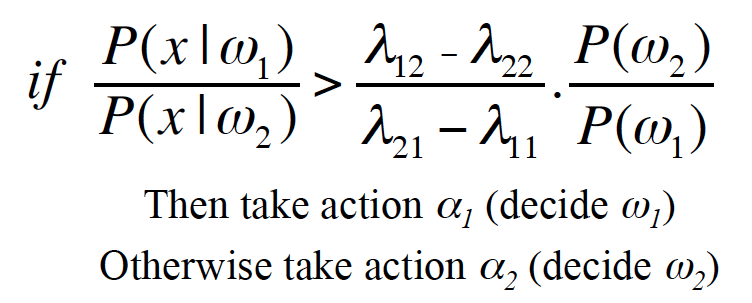
%---------------------------------------------------------%

% Classification on the Projected Data on One-Dimensional Space

Ntest = size(Dtest, 1);

***Make the function with two classification model choices:***

*1)* Using Bayesian Decision Boundary based on the derived decision making rule:



**Matlab Code:**

%% Option 1: Using Derived Bayesian Decision Boundary

% if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

% The Optimal Decision Rule, check Slides of Lecture 4 Bayesian Thoery, Page 27

threshold = 0;

for i = 1:Ntest

feat = Dtest\_new(i);

prior1 = length(idx1)/length(Ltrain);

likelihood1 = normpdf(feat, mu1\_new,sigma1\_new); % likelihood of the current class 1

prior2 = length(idx2)/length(Ltrain);

likelihood2 = normpdf(feat, mu2\_new,sigma2\_new); % likelihood of the current class -1

Vcheck = (likelihood1/likelihood2) - (lambda(1,2)-lambda(2,2))/(lambda(2,1)-lambda(1,1))\*(prior2/prior1);

if Vcheck > threshold

pred = 1;

else

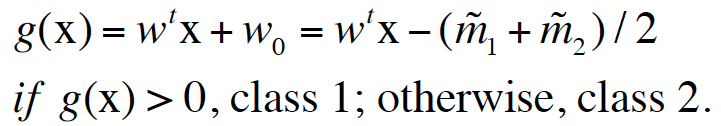
pred = -1;

end

Lpred(i,1) = pred;

end

2) Using the middle line of the projected means as the decision boundary for classification, that is



**Matlab Code:**

%% Option 2: Using the mid-line of projected means

%threshold\_star = (mu1\_new + mu2\_new)/2;

threshold = 0;

for i = 1:Ntest

feat = Dtest\_new(i);

if abs(feat-mu1\_new) - abs(feat-mu2\_new) < threshold

pred = 1;

else

pred = -1;

end

Lpred(i,1) = pred;

end

3) Binary classification for the Breast Cancer Dataset:

For each classification model above, perform reported the classification accuracy, sensitivity, and specificity using 5-fold cross-validation.

For the Bayesian Classification Model, used the following two choices of penalty costs to make classification rule in your program: (I) λ11 = 0, λ22 = 0, λ12 = 2, λ21 = 1; (II) λ11 = 0, λ22 = 0, λ12 = 1, λ21 = 1. Here λ**ij**is the penalty cost to classify a class **j** sample as class **i**.

**Matlab Code:**

%% HW4 main program

clear all; clc; close all;

%-----1.Load Raw Data----------%

% Load feat and label from Breast\_Cancer\_dataset.xlsx in to the workspace

data = importdata('Breast\_Cancer\_dataset.xlsx');

data = data.data;

feat = data(:,1:9); % feature matrix

label = data(:,10); % class label vector

%---------Use IRIS Dataset---------------%

% data = load(['iris.txt']);

% feat = data(1:100,1:4); % feature matrix

% label = data(1:100,5); % class label vector

%----------------------------------------%

idx = find(label==2);

label(idx) = -1; % Use -1 to present benign, then the labels are 1 & -1

%-----2. Prepare N-fold dataset for classification----------%

N = 5; % N-fold cross validation

data\_nfold = divide\_nfold\_data(feat, label, N);

%-----3. Perform N-fold Cross-Validation using KNN Function-----------------%

C = unique(label); %extract label information from label vector

ACC\_SUM = [];

acc\_nfold = [];

senspe\_nfold =[];

auc\_nfold = [];

for ifold = 1:N

%----prepare cross-validation training and testing dataset---%

idx\_test = ifold; % index for testing fold

idx\_train = setdiff(1:N, ifold); % index for training folds

Dtest = []; Ltest = []; % initialize testing data and label

Dtrain = []; Ltrain = []; % initialize testing data and label

%---construct the training and testing dataset for the ith fold cross validatoin

for iC = 1:length(C)

cl = C(iC);

dtest = eval(['data\_nfold.class',num2str(iC), '.fold', num2str(ifold)]);

Dtest = [Dtest; dtest];

Ltest = [Ltest; cl\*ones(size(dtest,1), 1)];

for itr = 1:length(idx\_train)

idx = idx\_train(itr);

dtrain = eval(['data\_nfold.class',num2str(iC), '.fold', num2str(idx)]);

Dtrain = [Dtrain; dtrain];

Ltrain = [Ltrain; cl\*ones(size(dtrain,1), 1)];

end

end

%-----------------------------------------------------%

%----------LDA Classification-------------------------%

% Classification using the function Fisher's Linear Discrimiant Analysis (LDA)

lambda = [0 1; 1 0]; % lambda = [0 1; 1 0]; optionI

%lambda = [0 2; 1 0]; % lambda = [0 2; 1 0]; optionII

% Lpred = FishersLDA(Dtrain, Ltrain, Dtest, lambda, option);

% The updated function FishersLDA\_v2 can calculated AUC value

% Option 1: Using Derived Bayesian Decision Rule

% if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

%disp("Using Derived Bayesian Decision Rule");

Lpred = FishersLDA(Dtrain, Ltrain, Dtest, lambda, 1);

%---------------------------------------------------------%

%% Option 2: Using the mid-line of projected means

%threshold\_star = (mu1\_new + mu2\_new)/2;

Lpred = FishersLDA\_v2(Dtrain, Ltrain, Dtest, lambda, 2);

%---------------------------------------------------------%

%Option 3: Using the myBayesPredict funciton

%Lpred = FishersLDA\_v2(Dtrain, Ltrain, Dtest, Ltest, lambda, 3);

%---Calculate Classification Accuracy-----%

acc = sum(Lpred==Ltest)/length(Ltest);

disp("Acuuracy for fold "+ifold+":"+acc);

ConfusionMatrix=confusionmat(Ltest,Lpred);

ConfusionMatrix

%---Calculate Sensitivity & Specificity based on Lpred and Ltest-----%

idx1 = find(Ltest==1); pred1 = Lpred(idx1);

sen = length(find(pred1==1))/length(idx1);

idx2 = find(Ltest==-1); pred2 = Lpred(idx2);

spe = length(find(pred2==-1))/length(idx2);

disp("Sensitivity: "+sen);

disp("Specificity: "+spe);

disp("-------------------------------------------------------------------------");

**Output of above program:**

**Using Derived Bayesian Decision Rule:**

if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

where

**lambda11=0**

**lambda22=0**

**lambda12=1**

**lambda21=0**

Acuuracy for fold 1:0.625

ConfusionMatrix =

11 2

7 4

Sensitivity: 0.36364

Specificity: 0.84615

-------------------------------------------------------------------------

Acuuracy for fold 2:0.63636

ConfusionMatrix =

10 2

6 4

Sensitivity: 0.4

Specificity: 0.83333

-------------------------------------------------------------------------

Acuuracy for fold 3:0.69565

ConfusionMatrix =

11 2

5 5

Sensitivity: 0.5

Specificity: 0.84615

-------------------------------------------------------------------------

Acuuracy for fold 4:0.73913

ConfusionMatrix =

10 3

3 7

Sensitivity: 0.7

Specificity: 0.76923

-------------------------------------------------------------------------

Acuuracy for fold 5:0.45833

ConfusionMatrix =

1 12

1 10

Sensitivity: 0.90909

Specificity: 0.076923

Using Derived Bayesian Decision Rule:

if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

where

**lambda11=0**

**lambda22=0**

**lambda12=2**

**lambda21=1**

Acuuracy for fold 1:0.625

ConfusionMatrix =

13 0

9 2

Sensitivity: 0.18182

Specificity: 1

-------------------------------------------------------------------------

Acuuracy for fold 2:0.625

ConfusionMatrix =

13 0

9 2

Sensitivity: 0.18182

Specificity: 1

-------------------------------------------------------------------------

Acuuracy for fold 3:0.56522

ConfusionMatrix =

12 1

9 1

Sensitivity: 0.1

Specificity: 0.92308

-------------------------------------------------------------------------

Acuuracy for fold 4:0.63636

ConfusionMatrix =

10 2

6 4

Sensitivity: 0.4

Specificity: 0.83333

-------------------------------------------------------------------------

Acuuracy for fold 5:0.43478

ConfusionMatrix =

1 12

1 9

Sensitivity: 0.9

Specificity: 0.076923

-------------------------------------------------------------------------

Using the mid-line of projected means

Acuuracy for fold 1:0.625

ConfusionMatrix =

11 2

7 4

Sensitivity: 0.36364

Specificity: 0.84615

-------------------------------------------------------------------------

Acuuracy for fold 2:0.63636

ConfusionMatrix =

10 2

6 4

Sensitivity: 0.4

Specificity: 0.83333

-------------------------------------------------------------------------

Acuuracy for fold 3:0.66667

ConfusionMatrix =

11 2

6 5

Sensitivity: 0.45455

Specificity: 0.84615

-------------------------------------------------------------------------

Acuuracy for fold 4:0.73913

ConfusionMatrix =

10 3

3 7

Sensitivity: 0.7

Specificity: 0.76923

-------------------------------------------------------------------------

Acuuracy for fold 5:0.43478

ConfusionMatrix =

1 12

1 9

Sensitivity: 0.9

Specificity: 0.076923

Using Bayes Predict function

Acuuracy for fold 1:0.625

ConfusionMatrix =

11 2

7 4

Sensitivity: 0.36364

Specificity: 0.84615

-------------------------------------------------------------------------

Acuuracy for fold 2:0.65217

ConfusionMatrix =

10 2

6 5

Sensitivity: 0.45455

Specificity: 0.83333

-------------------------------------------------------------------------

Acuuracy for fold 3:0.65217

ConfusionMatrix =

11 2

6 4

Sensitivity: 0.4

Specificity: 0.84615

-------------------------------------------------------------------------

Acuuracy for fold 4:0.73913

ConfusionMatrix =

10 3

3 7

Sensitivity: 0.7

Specificity: 0.76923

-------------------------------------------------------------------------

Acuuracy for fold 5:0.43478

ConfusionMatrix =

1 12

1 9

Sensitivity: 0.9

Specificity: 0.076923

-------------------------------------------------------------------------

2) ROC analysis:

Used the first **38 samples** in Class 1 and the first **48 samples** in Class 2 to construct **training** dataset, and the remaining samples for testing dataset.

I.

**Matlab Code:**

Option 1: Using Derived Bayesian Decision Boundary

if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

Where lambda11=0,lambda22=0,lambda12=1,lambda21=1

%homework 4 :problem 2

%Perform ROC analyses for the three classification models developed in problem 1

clear all; close all; clc

data = importdata('Breast\_Cancer\_dataset.xlsx');

data = data.data;

X = data(:,1:9); % feature matrix

Y = data(:,10); % class label vector

totalSamples=116;

%----------------------------------------------------------

%assign binary values 1 and -1

for i=1:116

if Y(i)<2

Y(i)=-1;%For healthy ppl

else

Y(i)=1;%for patients

end

end

%used the first 38 samples in Class 1 and the first 48 samples in Class 2 to construct training dataset,

%and the remaining samples for testing dataset.

trainDataA=X (1:38,:);

trainLabelA=Y (1:38,:);

testDataA=X (39:52,:);

testLabelA=Y (39:52,:);

trainDataB=X (53:101,:);

trainLabelB=Y (53:101,:);

testDataB=X (102:116,:);

testLabelB=Y (102:116,:);

trainData=[trainDataA ;trainDataB];

testData=[testDataA ;testDataB];

trainLabel=[trainLabelA; trainLabelB];

testLabel=[testLabelA ;testLabelB];

lambda = [0 1; 1 0];

option = 1;

[Lpred, w, AUC, ROC, senspe] = FishersLDA\_v2(trainData, trainLabel, testData,testLabel, lambda, option);

%---------------------------------------------------------%

%---Calculate Classification Accuracy-----%

acc = sum(Lpred==testLabel)/length(testLabel);

disp("Accuracy: "+acc);

%---Calculate Sensitivity & Specificity based on Lpred and Ltest-----%

idx1 = find(testLabel==1); pred1 = Lpred(idx1);

sen = length(find(pred1==1))/length(idx1);

idx2 = find(testLabel==-1); pred2 = Lpred(idx2);

spe = length(find(pred2==-1))/length(idx2);

disp("Sensitivity: "+sen);

disp("Specificity: "+spe);

disp("AUC: "+AUC)

Code for option 1:

%% Perform ROC Analysis for Option 1

if option==1

%% Option 1: Using Derived Bayesian Decision Boundary

% if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

% The Optimal Decision Rule, check Slides of Lecture 4 Bayesian Thoery, Page 27

senspe = [];

bin = range(Vlist)/150;

Vlist\_sort = sort(Vlist, 'ascend');

threshold\_list = (Vlist\_sort(1)-bin):bin:(Vlist\_sort(end)+bin);

for ilist = 1:length(threshold\_list)

threshold = threshold\_list(ilist);

Lpred = [];

for i = 1:Ntest

feat = Dtest\_new(i);

prior1 = length(idx1)/length(Ltrain);

likelihood1 = normpdf(feat, mu1\_new,sigma1\_new); % likelihood of the current class 1

prior2 = length(idx2)/length(Ltrain);

likelihood2 = normpdf(feat, mu2\_new,sigma2\_new); % likelihood of the current class -1

Vcheck = (likelihood1/likelihood2) - (lambda(1,2)-lambda(2,2))/(lambda(2,1)-lambda(1,1))\*(prior2/prior1);

if Vcheck > threshold

pred = 1;

else

pred = -1;

end

Lpred(i,1) = pred;

end

[sen, spe] = cal\_senspe(Lpred, Ltest);

senspe = [senspe; sen spe];

end

%% To calculate ROC

% ROC = [senspe(:,1), 1-senspe(:,2)];

sen = senspe(:,1);

if sen(1) > sen(end)

senspe = senspe(end:-1:1, :);

end

ROC = [senspe(:,1), 1-senspe(:,2)];

bin = ROC(:,2) - [0; ROC(1:end-1,2)];

AUC = sum(ROC(:,1).\*bin);

plot(smooth(ROC(:,2)), smooth(ROC(:,1)));

title("Derived Bayesian Decision Boundary");

xlim([-0.2 1.2]);

ylim([0 1.2]);

end

**Output of above program:**

I.Using Derived Bayesian Decision Boundary

Option 1: Using Derived Bayesian Decision Boundary

if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

**Where lambda11=0,lambda22=0,lambda12=1,lambda21=1**

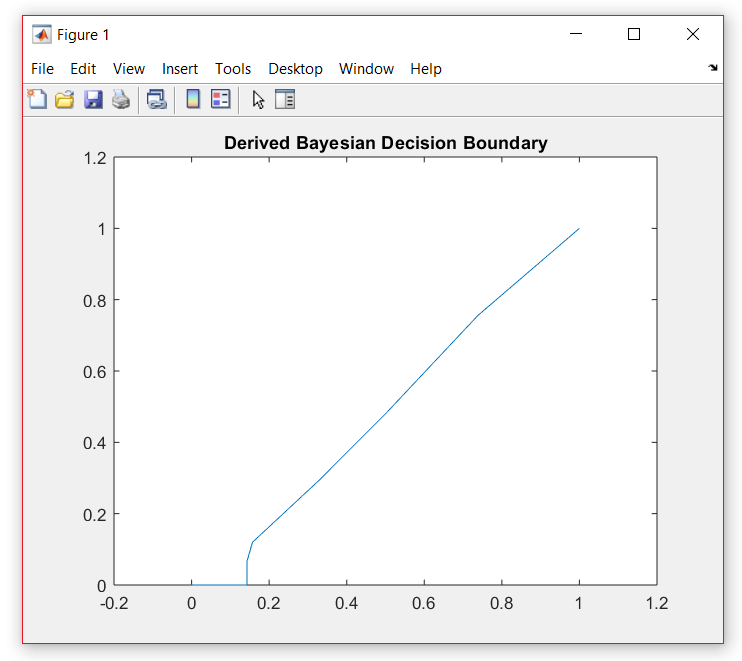
Accuracy: 0.48276

Sensitivity: 0

Specificity: 1

AUC: 0.75714

ROC Curve:



II.Using Derived Bayesian Decision Boundary

Option 1: Using Derived Bayesian Decision Boundary

if Ratio of likelihood > [(lambda12-lambda22)/(lambda21-lambda11)]\*ratio of prior

**Where lambda11=0,lambda22=0,lambda12=2,lambda21=1**

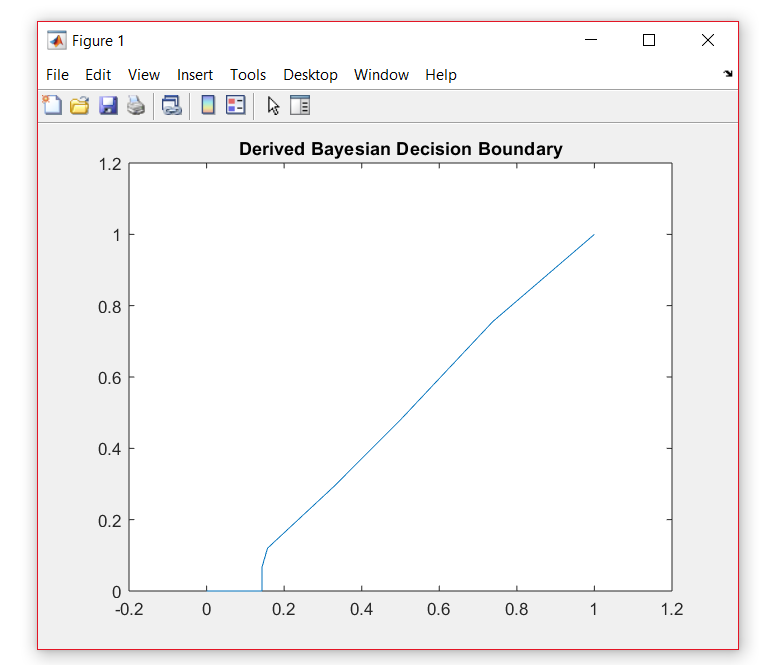
Accuracy: 0.48276

Sensitivity: 0

Specificity: 1

AUC: 0.75714

ROC Curve:



III: Using the mid-line of projected means

if option==2

%% Option 2: Using the mid-line of projected means

senspe = [];

bin = range(Vlist)/150;

Vlist\_sort = sort(Vlist, 'ascend');

threshold\_list = (Vlist\_sort(1)-bin):bin:(Vlist\_sort(end)+bin);

for ilist = 1:length(threshold\_list)

threshold = threshold\_list(ilist);

Lpred = [];

for i = 1:Ntest

feat = Dtest\_new(i);

if abs(feat-mu1\_new) - abs(feat-mu2\_new) < threshold

pred = 1;

else

pred = -1;

end

Lpred(i,1) = pred;

end

[sen, spe] = cal\_senspe(Lpred, Ltest);

senspe = [senspe; sen spe];

end

%% To calculate ROC

% ROC = [senspe(:,1), 1-senspe(:,2)];

sen = senspe(:,1);

if sen(1) > sen(end)

senspe = senspe(end:-1:1, :);

end

ROC = [senspe(:,1), 1-senspe(:,2)];

bin = ROC(:,2) - [0; ROC(1:end-1,2)];

AUC = sum(ROC(:,1).\*bin);

plot(smooth(ROC(:,2)), smooth(ROC(:,1)));

title("Using the mid-line of projected means");

xlim([-0.2 1.2]);

ylim([0 1.2]);

end

**Output of above program:**

Option 2: Using the mid-line of projected means

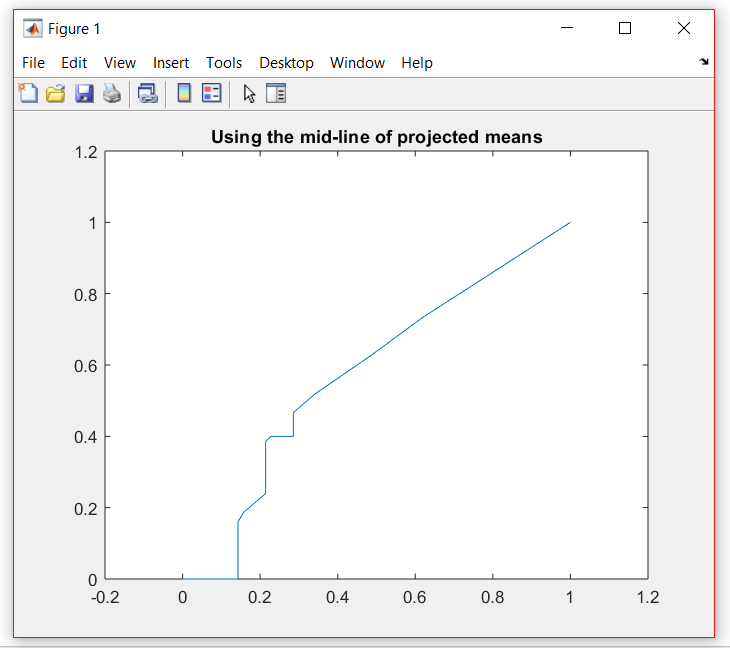
Accuracy: 0.51724

Sensitivity: 1

Specificity: 0

AUC: 0.68095

ROC Curve:



Conclusion:

Area under ROC curve is often used as a measure of quality of the classification models. A random classifier has an area under the curve of 0.5, while AUC for a perfect classifier is equal to 1.

From Above outputs and ROC curves, the AUC using midline as a projected mean is 0.68095 which is less than AUC using derived bayesian boundary which is Same 0.75714 for different lambda values. So ,**Bayesian Decision Boundary based on the derived decision making rule** isthe best choice.

References:

<https://in.mathworks.com/help/> <https://in.mathworks.com/help/stats/examples/classification.html>

Lecture3\_Classification Basics

Lecture4\_Bayesian Theory

Lecture5\_Linear Discriminant Functions