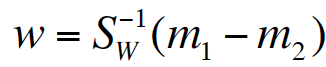
**IE 6318 Data Mining and Analytics**

**Homework 4**

Classification Using Fisher Linear Discriminant and Perform ROC Analysis

1. Make a classification function based on Fisher Linear Discriminant. From the lecture, we introduced the optimal projection direction *w* is:

.

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

xlim([-0.2 1.2]);

ylim([0 1.2]);

end

%% Perform ROC Analysis for Option 2

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

xlim([-0.2 1.2]);

ylim([0 1.2]);

end

%% ROC Analysis for Option 3 is not available

if option==3

disp('This option is not available now.');

ROC = [];

AUC = [];

senspe = [];

end

One can perform classification on the one-dimensional space for the projected data samples *wtx. Make the function with two classification model choices:*

Fixing the Nfold Data Set and then proceeding to do the remaining classification as each time nfold classification varies.

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];

option = 1;

%threshold\_list = -75:1:75;

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

[Lpred, w, AUC, ROC, senspe] = FishersLDA\_v2(Dtrain, Ltrain, Dtest, Ltest, lambda, option);

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

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

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

%---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);

%---Record the results----%

acc\_nfold(ifold, 1) = acc;

senspe\_nfold = [senspe\_nfold; sen, spe];

auc\_nfold = [auc\_nfold; AUC];

end

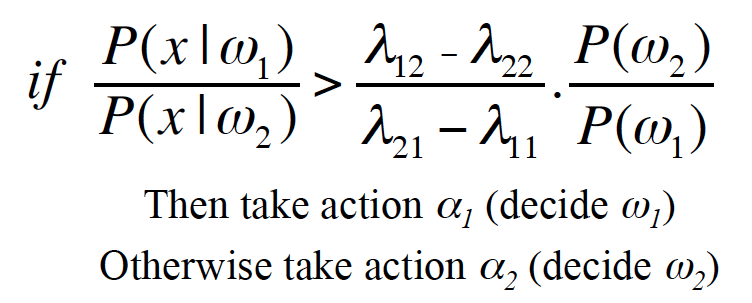
acc\_ave = mean(acc\_nfold); % average of N folds of cross validations

senspe\_ave = mean(senspe\_nfold);

auc\_ave = mean(auc\_nfold);

ACC\_SUM = [ACC\_SUM; acc\_ave, senspe\_ave];

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



(1)for lamda=(0 2;1 0)

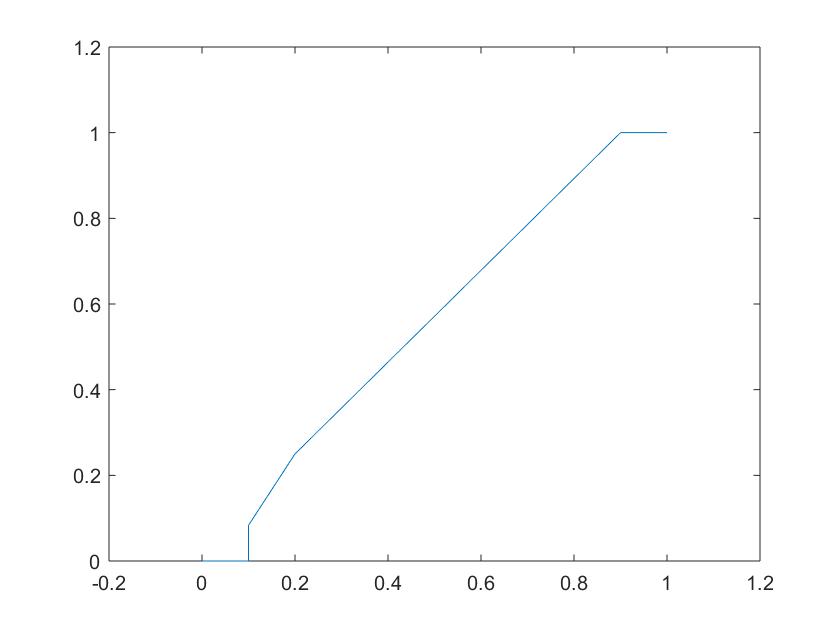
Average accuracy:- 0.448155467720685

Sensitivity :- 0

Specitivity :- 1

Area Under The Curve Average:- 0.8767

**ROC Curve:-**



(2) for lamda = (0 1;1 0)

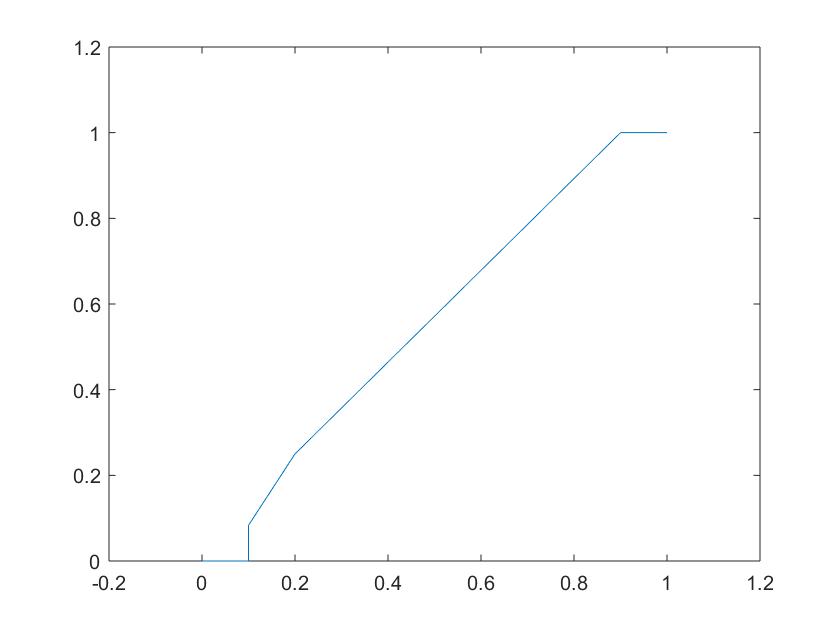
Average accuracy:- 0.448155467720685

Sensitivity :- 0

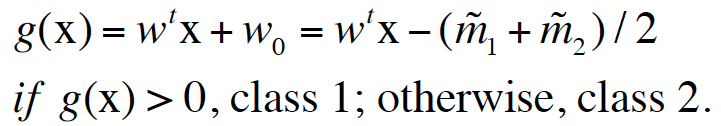
Specitivity :- 1

Area Under The Curve Average:- 0.8767

**ROC Curve:-**



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



Average accuracy:- 0.5518

Sensitivity :- 1

Specitivity :- 0

Area Under The Curve Average :- 0.79533

**ROC Curve:-**

