BM 59D Homework#3 Report

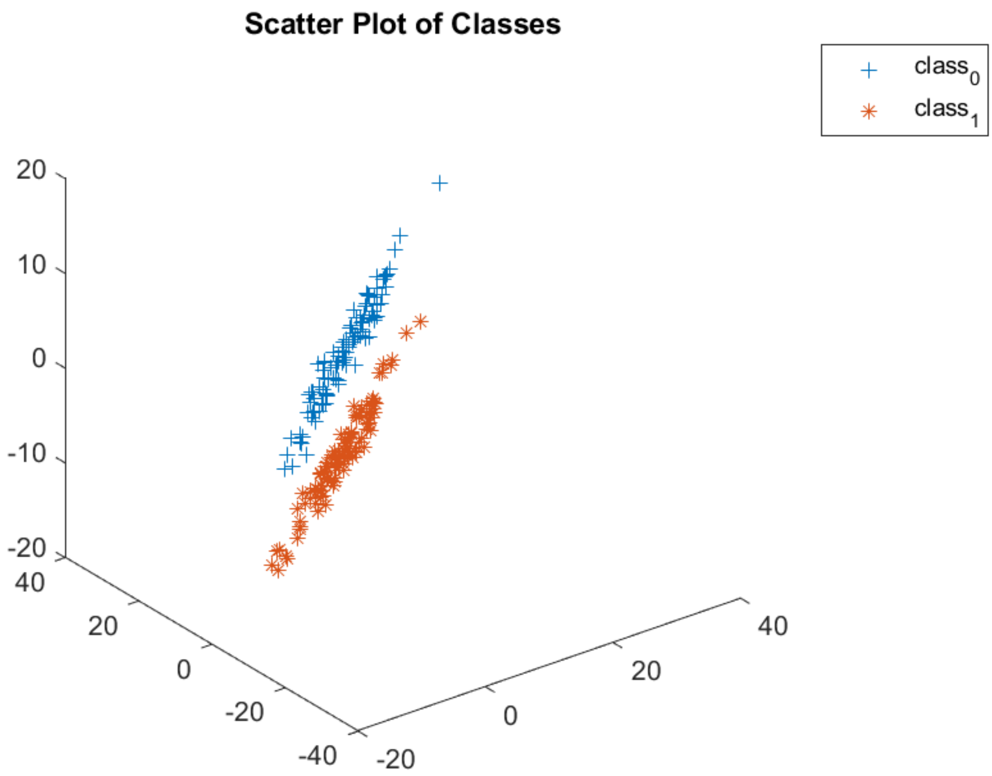
The Confusion Matrix for X:

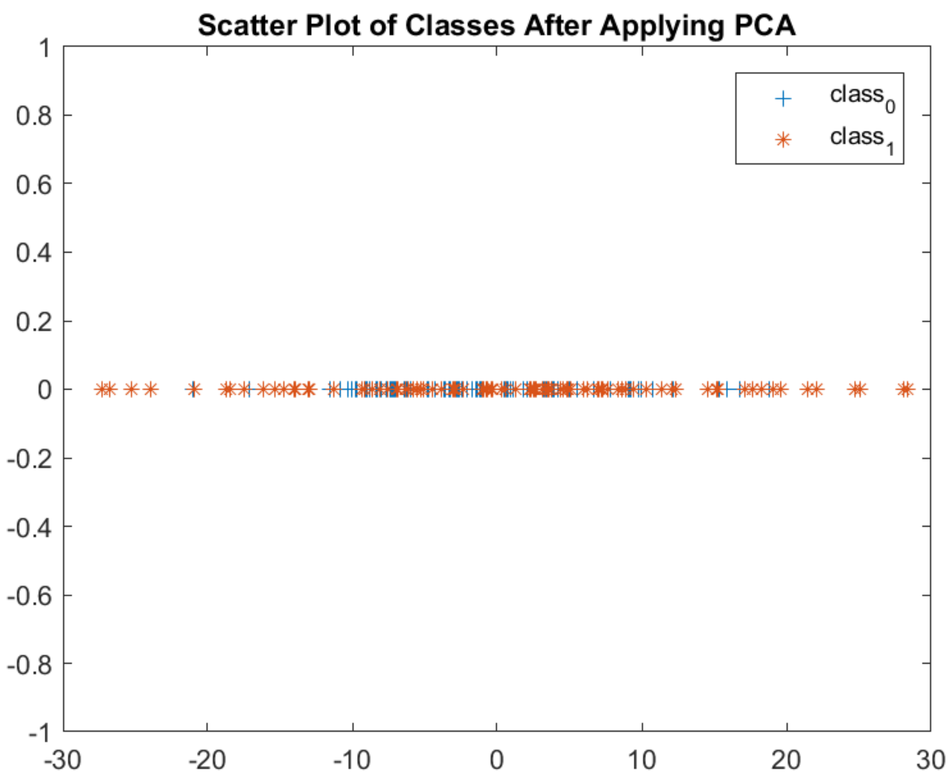
Quadratic Discrimant	Prediction for Class0	Prediction for Class 1
Actual for Class 0	40	0
Actual for Class 1	0	40
Linear Discriminant	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Naive Bayes Discriminant	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	36	4
Actual for Class 1	2	38
Euclidean Discriminant	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	36	4
Actual for Class 1	2	38
Quadratic Discrimant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	0	40
Actual for Class 1	40	0
Linear Discriminant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	24	16
Actual for Class 1	16	24
Naive Bayes Discriminant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	24	16
Actual for Class 1	16	24
Euclidean Discriminant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	24	16
Actual for Class 1	16	24
Quadratic Discrimant-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Linear Discriminant-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Naive Bayes Discr-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Euclidean Discriminant-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40

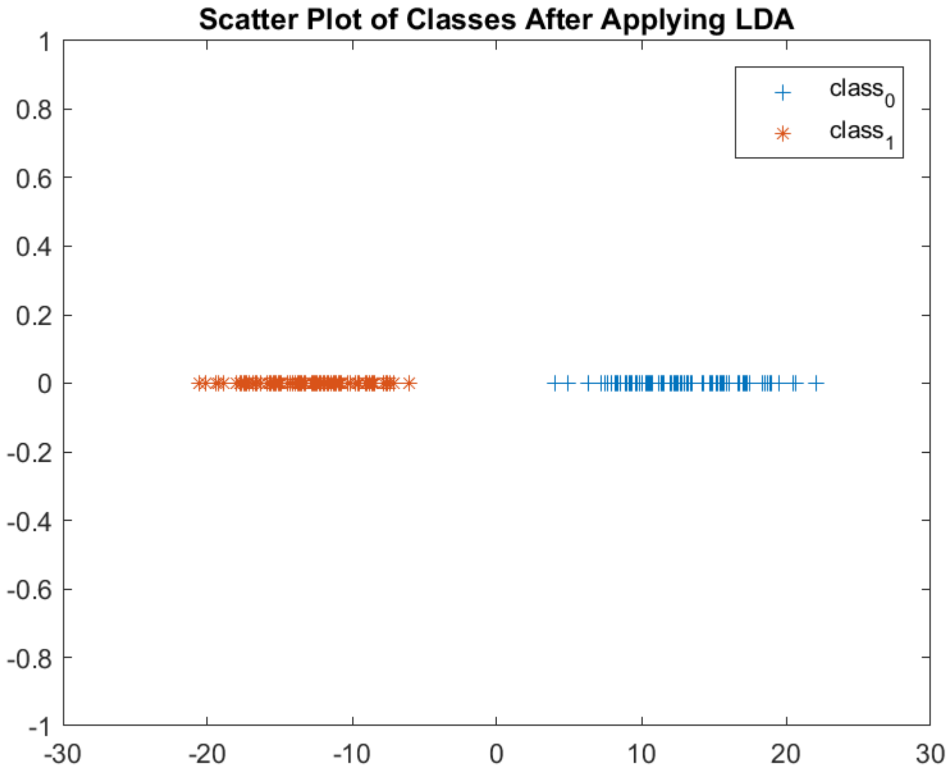
The Confusion Matrix for Y:

Quadratic Discrimant	Prediction for Class0	Prediction for Class 1
Actual for Class 0	39	1
Actual for Class 1	0	40
Linear Discriminant	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Naive Bayes Discriminant	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	37	3
Actual for Class 1	4	36
Euclidean Discriminant	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	38	2
Actual for Class 1	4	36
Quadratic Discrimant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	33	7
Actual for Class 1	3	37
Linear Discriminant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Naive Bayes Discriminant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	38	2
Actual for Class 1	4	36
Euclidean Discriminant-PCA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	38	2
Actual for Class 1	4	36
Quadratic Discrimant-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Linear Discriminant-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	40	0
Actual for Class 1	0	40
Naive Bayes Discr-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	38	2
Actual for Class 1	2	38
Euclidean Discriminant-LDA	**Prediction for Class0**	**Prediction for Class 1**
Actual for Class 0	38	2
Actual for Class 1	2	38

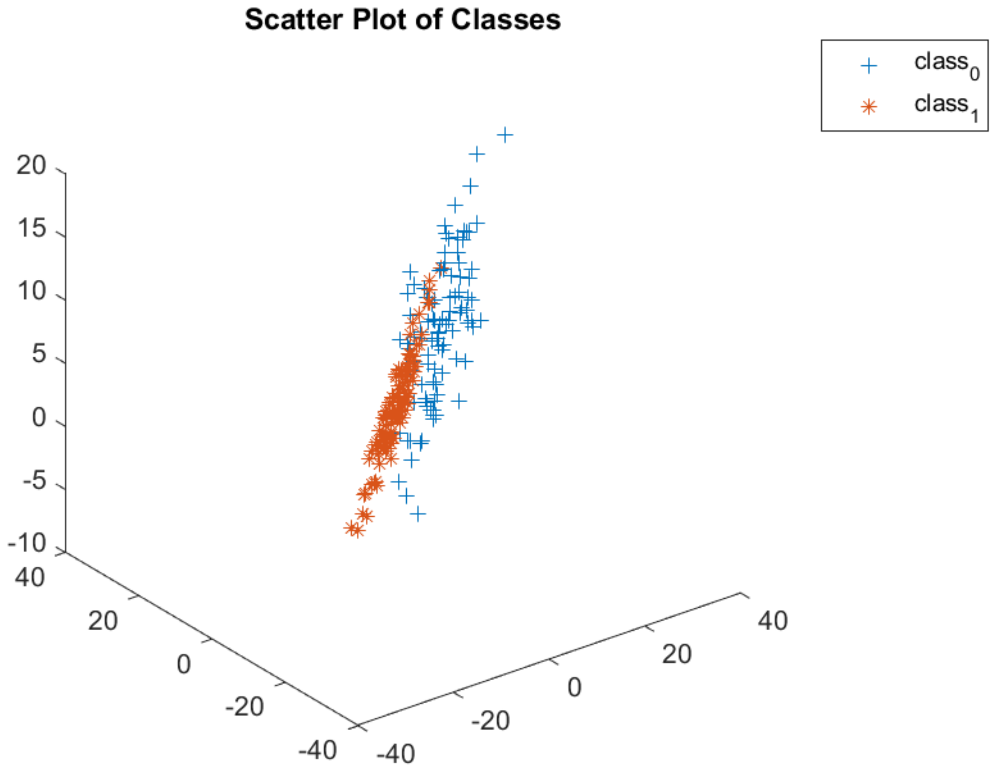
The Plots for X:

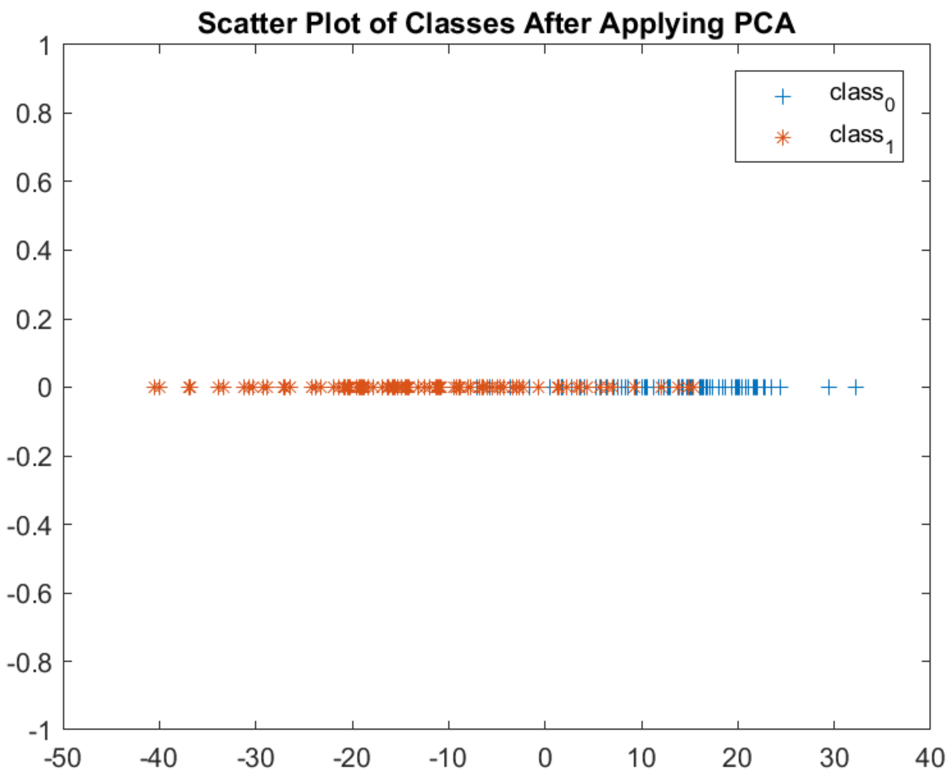


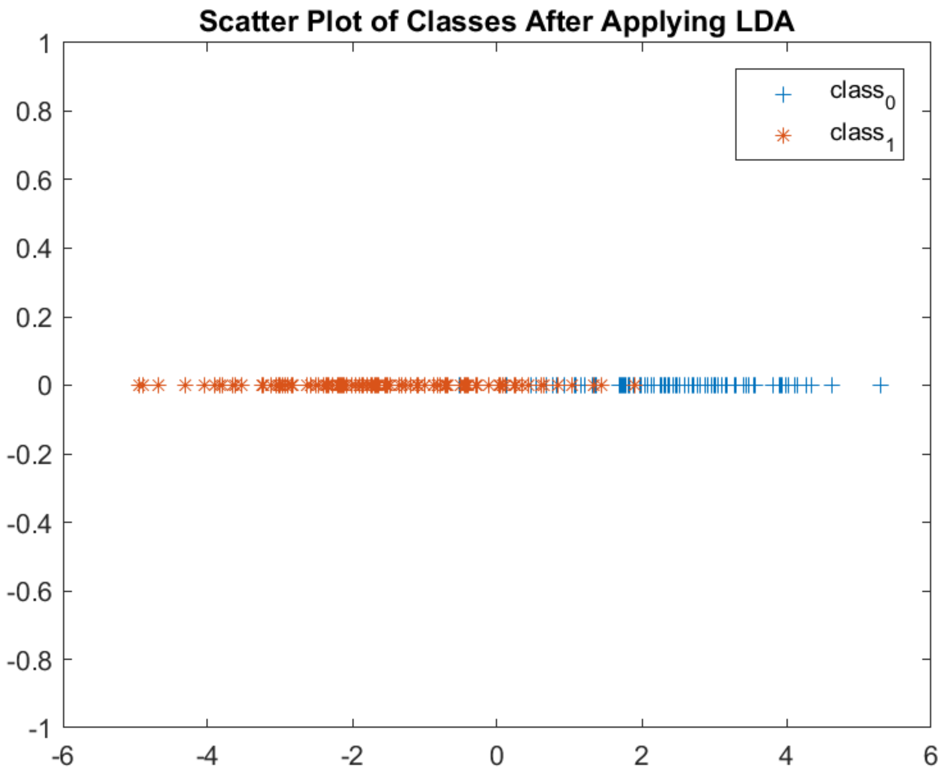



Scatter Plot of Classes After Applying LDA
class0
class1
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-30
-20
-10
0
10
20
30

The Plots for Y:







In the case of different discriminant functions, it is obvious that ignoring the correlations between the variables in the covariance matrix e.g. Naive Bayes and Euclidean discriminations causes accuracy to decrease.

Class 0 and Class 1 in data X have the close means; however, Class 0 and Class 1 in data Y have the diffrent means. In other words, the classes of data Y are more separable than data X when chosen best distance function ("...finding the best discriminant function as the task of finding the best distance function..." page.105) .

After applying PCA and LDA on each data, the projection of Class 0 and Class 1 of the data X becomes non-separable when we look at the plot 2 for the data X. Therefore, the confussion matrix results in the worse accuracy than previous experiences. However, LDA performs well in class separation in the data X, which proves that "LDA direction is superior to PCA direction, since LDA uses the class information" (page 143).

Appendix

main.m

```
data= load('C:\Users\Bernisko\Google Drive\Masters\BM
59D\HW3\BM59D_Hw3_Data.mat')
X = data.X;
display('X')
classification(X)

display('Y')
Y = data.Y;
classification(Y)
```

classification.m

```
function classification(data)

x_t = data(:,1:3);
x_r= data(:,4);

[training_labels,class_0,class_1,training_0,training_1,validation_0,validation_1
] = split_train_validation_data(x_t,x_r);
plot3(class_0(:,1),class_0(:,2),class_0(:,3),'+',class_1(:,1),class_1(:,2),class
_1(:,3),'*');
title('Scatter Plot of Classes');
legend('class_0','class_1');
saveas(gcf,'plot_1','png');
[mean_0,cov_0]= calc_class_params(training_0);
prior_0 = length(training_labels (training_labels ==
0,:))/length(training_labels );
[mean_1,cov_1] =calc_class_params(training_1);
prior_1 = length(training_labels (training_labels ==
1,:))/length(training_labels );
cov= prior_0*cov_0 + prior_1*cov_1;
display('Quadratic Discriminant')
prediction('quadratic',validation_0,validation_1,mean_0,cov_0,prior_0,mean_1,cov
_1,prior_1);
display('Linear Discriminant')
prediction('linear',validation_0,validation_1,mean_0,cov,prior_0,mean_1,cov,prio
r_1);
display('Naive Bayes Discriminant')
prediction('naivebayes',validation_0,validation_1,mean_0,cov,prior_0,mean_1,cov,
prior_1);
display('Euclidean Discriminant')
prediction('euclidean',validation_0,validation_1,mean_0,cov,prior_0,mean_1,cov,p
rior_1);

display('PCA')
[mean_t,cov_t]= calc_class_params(x_t);
[eigenvector,eigenvalues] = eig(cov_t);
[max_eigenvalue,max_eigenvector_idx]=max(diag(eigenvalues));
red_x_t=(eigenvector(:,max_eigenvector_idx).'*(x_t -mean_t).').';
[red_training_labels,red_class_0,red_class_1,red_training_0,red_training_1,red_v
alidation_0,red_validation_1] = split_train_validation_data(red_x_t,x_r);
plot(red_class_0(:,1),zeros(1,100),'+',red_class_1(:,1),zeros(1,100),'*');
title('Scatter Plot of Classes After Applying PCA')
legend('class_0','class_1');
saveas(gcf,'plot_2','png');
[red_mean_0,red_cov_0]= calc_class_params(red_training_0);
red_prior_0 = length(red_training_labels(red_training_labels ==
0,:))/length(red_training_labels );
[red_mean_1,red_cov_1] =calc_class_params(red_training_1);
```

```
red_prior_1 = length(red_training_labels (red_training_labels ==
1,:))/length(red_training_labels );
red_cov= red_prior_0*red_cov_0 + red_prior_1*red_cov_1;
display('Quadratic Discriminant')
prediction('naivebayes',red_validation_0,red_validation_1,red_mean_0,red_cov_0,r
ed_prior_0,red_mean_1,red_cov_1,red_prior_1);
display('Linear Discriminant')
prediction('linear',red_validation_0,red_validation_1,red_mean_0,red_cov,red_pri
or_0,red_mean_1,red_cov,red_prior_1);
display('Naive Bayes Discriminant')
prediction('naivebayes',red_validation_0,red_validation_1,red_mean_0,red_cov,red
_prior_0,red_mean_1,red_cov,red_prior_1);
display('Euclidean Discriminant')
prediction('euclidean',red_validation_0,red_validation_1,red_mean_0,red_cov,red_
prior_0,red_mean_1,red_cov,red_prior_1);

display('LDA')
w = (inv(cov_0 + cov_1)*(mean_0-mean_1).');
z_t = (w.'*x_t.').';
[z_training_labels,z_class_0,z_class_1,z_training_0,z_training_1,z_validation_0,
z_validation_1] = split_train_validation_data(z_t,x_r);
plot(z_class_0(:,1),zeros(1,100),'+',z_class_1(:,1),zeros(1,100),'*');
title('Scatter Plot of Classes After Applying LDA')
legend('class_0','class_1');
saveas(gcf,'plot_3','png');
[z_mean_0,z_cov_0]= calc_class_params(z_training_0);
z_prior_0 = length(z_training_labels(z_training_labels ==
0,:))/length(z_training_labels );
[z_mean_1,z_cov_1] =calc_class_params(z_training_1);
z_prior_1 = length(z_training_labels(z_training_labels ==
1,:))/length(z_training_labels );
z_cov= z_prior_0*z_cov_0 + z_prior_1*z_cov_1;
display('Quadratic Discriminant')
prediction('quadratic',z_validation_0,z_validation_1,z_mean_0,z_cov_0,z_prior_0,
z_mean_1,z_cov_1,z_prior_1);
display('Linear Discriminant')
prediction('linear',z_validation_0,z_validation_1,z_mean_0,z_cov,z_prior_0,z_mea
n_1,z_cov,z_prior_1);
display('Naive Bayes Discriminant')
prediction('naivebayes',z_validation_0,z_validation_1,z_mean_0,z_cov,z_prior_0,z
_mean_1,z_cov,z_prior_1);
display('Euclidean Discriminant')
prediction('euclidean',z_validation_0,z_validation_1,z_mean_0,z_cov,z_prior_0,z_
mean_1,z_cov,z_prior_1);

end
```

split_train_validation_data.m

```
function varargout = split_train_validation_data(sample,label)
class_0 = sample(label == 0,:);
class_1 = sample(label == 1,:);
training = cat(1,class_0(1:60,:), class_1(1:60,:)) ;
training_labels = cat(1,label(1:60,:), label(101:160,:)) ;
validation =  cat(1,class_0(61:100,:), class_1(61:100,:));
validation_labels = cat(1,label(61:100,:), label(161:200,:));
training_0 = class_0(1:60,:);
training_1 = class_1(1:60,:);
validation_0 = class_0(61:100,:);
validation_1=class_1(61:100,:);
varargout =
{training_labels,class_0,class_1,training_0,training_1,validation_0,validation_1
};
```

```
end
```

calc_class_params.m

```
function [mu cov] = calc_class_params(class)
[sample_size,dim] =size(class);
mu = mean(class);
cov =((class-mu)'*(class-mu))/sample_size;
end
```

prediction.m

```
function
prediction(disc_func,validation_0,validation_1,mean_0,cov_0,prior_0,mean_1,cov_1
,prior_1)
pred_labels_0
=predict_class(disc_func,validation_0,mean_0,cov_0,prior_0,mean_1,cov_1,prior_1)
;
pred_labels_1
=predict_class(disc_func,validation_1,mean_0,cov_0,prior_0,mean_1,cov_1,prior_1)
;
pred_labels = cat(1,pred_labels_0, pred_labels_1);
calc_confusion_matrix(pred_labels_0,pred_labels_1);
end
```

predict_class.m

```
function [pred_labels] =
predict_class(disc_func,validation,mean_0,cov_0,prior_0,mean_1,cov_1,prior_1)
likelihood_0 = calc_discriminant(disc_func,validation, mean_0,cov_0,prior_0);
likelihood_1 = calc_discriminant(disc_func,validation, mean_1,cov_1,prior_1);
for i=1:length(validation)
    if likelihood_0(i) > likelihood_1(i)
        pred_labels(i)=0;
    else
        pred_labels(i)=1;
    end
end
pred_labels = pred_labels.';
end
```

calc_discriminant.m

```
function [discriminant] = calc_discriminant(discriminant_func,class, mean,cov,
prior)
[sample_size,dim] =size(class);
 if isequal(discriminant_func,'quadratic')
    discriminant = (-0.5) * log(det(cov)) - (0.5*sum((class.' -
mean.').'*inv(cov)*(class.' - mean.'),2)) + log(prior);
 elseif isequal(discriminant_func,'linear')
     discriminant = - (0.5*sum((class.' - mean.').'*inv(cov)*(class.' -
mean.'),2)) + log(prior);
 elseif isequal(discriminant_func,'naivebayes')
     discriminant = - (0.5*sum(((class - mean)./diag(cov).').*((class -
mean)./diag(cov).'),2)) + log(prior);
 elseif isequal(discriminant_func,'euclidean')
     discriminant =- ((0.5/det(diag(diag(cov))))*sum((class - mean).*(class -
mean),2)) + log(prior);
end
```

calc_confusion_matrix.m

```
function calc_confusion_matrix(pred_labels_0,pred_labels_1)
tp = length(find(pred_labels_0==0));
fn = length(find(pred_labels_0==1));
fp = length(find(pred_labels_1==0));
tn =  length(find(pred_labels_1==1));
cm=[[tp,fn];[fp,tn]];
end
```