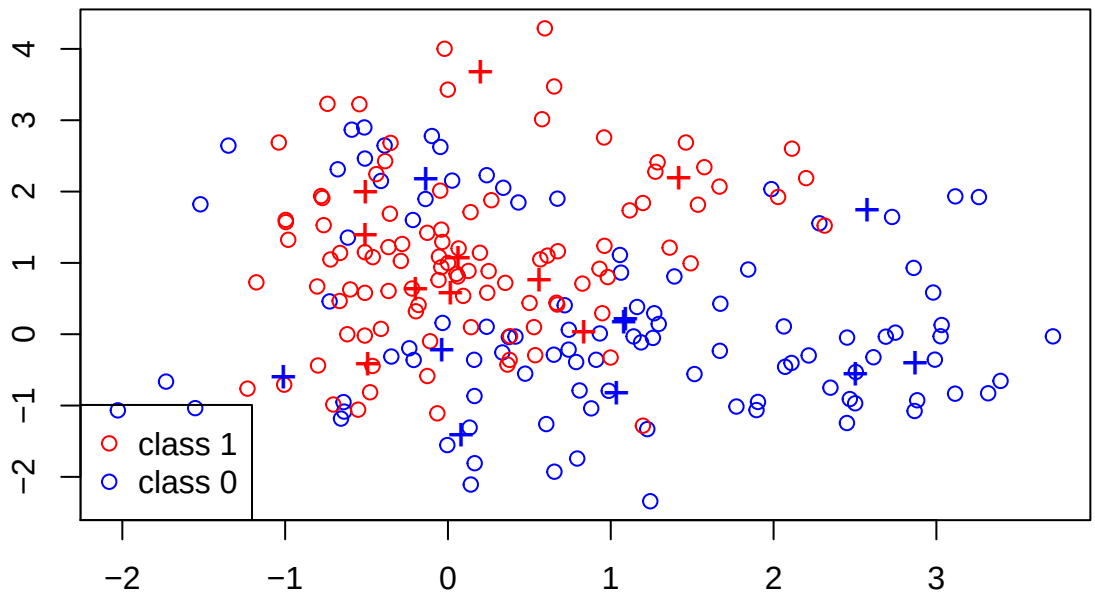
Coding Assignment 1

```
######################
#Prepare Data:Generate the 20 centers, 10 for each group.
######################
set.seed(6682)

csize = 10;       # number of centers
p = 2;
s = 1;            # sd for generating the centers within each class
m1 = matrix(rnorm(csize*p), csize, p)*s + cbind( rep(1,csize), rep(0,csize));
m0 = matrix(rnorm(csize*p), csize, p)*s + cbind( rep(0,csize), rep(1,csize));

#Generate training data.
n=100;
# Randomly allocate the n samples for class 1  to the 10 clusters
id1 = sample(1:csize, n, replace = TRUE);
# Randomly allocate the n samples for class 1 to the 10 clusters
id0 = sample(1:csize, n, replace = TRUE);

s= sqrt(1/5);                              # sd for generating x.

traindata = matrix(rnorm(2*n*p), 2*n, p)*s + rbind(m1[id1,], m0[id0,])
Ytrain = factor(c(rep(1,n), rep(0,n)))

#Generate test data.
N = 5000;
id1 = sample(1:csize, N, replace=TRUE);
id0 = sample(1:csize, N, replace=TRUE);
testdata = matrix(rnorm(2*N*p), 2*N, p)*s +
  rbind(m1[id1,], m0[id0,])
Ytest = factor(c(rep(1,N), rep(0,N)))

######################
#Visulization
######################

plot(traindata[, 1], traindata[, 2], type = "n", xlab = "", ylab = "")

points(traindata[1:n, 1], traindata[1:n, 2], col = "blue");
points(traindata[(n+1):(2*n), 1], traindata[(n+1):(2*n), 2], col="red");

points(m1[1:csize, 1], m1[1:csize, 2], pch="+", cex=1.5, col="blue");
points(m0[1:csize, 1], m0[1:csize, 2], pch="+", cex=1.5, col="red");

legend("bottomleft", pch = c(1,1), col = c("red", "blue"),
       legend = c("class 1", "class 0"))
```




```
######################
#K-NN method
######################

library(class)

## Choice of the neighhood size.
myk = c(151, 101, 69, 45, 31, 21, 11, 7, 5, 3, 1)
m = length(myk);

train_err_knn = rep(0,m);
test_err_knn = rep(0, m);

for( j in 1:m){
  Ytrain_pred_knn = knn(traindata, traindata, Ytrain, k = myk[j])
  train_err_knn[j] = sum(Ytrain != Ytrain_pred_knn)/(2*n)
  Ytest_pred_knn = knn(traindata, testdata, Ytrain,k = myk[j])
  test_err_knn[j] = sum(Ytest != Ytest_pred_knn)/(2*N)
}

######################
#Least Sqaure Method
######################
ls_cutoff = 0.5

RegModel = lm(as.numeric(Ytrain) - 1 ~ traindata)
```

```
Ytrain_pred_ls = as.numeric(RegModel$fitted > ls_cutoff)
Ytest_pred_ls = RegModel$coef[1] + RegModel$coef[2] * testdata[,1] +
                RegModel$coef[3] * testdata[,2]
Ytest_pred_ls = as.numeric(Ytest_pred_ls > ls_cutoff)

## cross tab for training data and training error
#table(Ytrain, Ytrain_pred_LS);
train_err_ls = sum(Ytrain !=  Ytrain_pred_ls) / (2*n);

## cross tab for test data and test error
#table(Ytest, Ytest_pred_LS);
test_err_ls = sum(Ytest !=  Ytest_pred_ls) / (2*N);

######################
#Calculate Bayes Error
######################

#Since we need to repeatly evaluate a mixture of normal with 10 components on each test point,
#I wrote a function first mixnorm, and then use the command apply to apply the same function on each ro

mixnorm=function(x){
  ## return the density ratio for a point x, where each
  ## density is a mixture of normal with 10 components
  sum(exp(-apply((t(m1)-x)^2, 2, sum)*5/2))/sum(exp(-apply((t(m0)-x)^2, 2, sum)*5/2))
}

Ytest_pred_Bayes = apply(testdata, 1, mixnorm)
Ytest_pred_Bayes = as.numeric(Ytest_pred_Bayes > 1);
#table(Ytest, Ytest_pred_Bayes);
test_err_Bayes = sum(Ytest !=  Ytest_pred_Bayes) / (2*N)

######################
#Plot the Performance
######################
plot(c(0.5,m), range(test_err_ls, train_err_ls, test_err_knn, train_err_knn),
     type="n", xlab="Degree of Freedom", ylab="Error", xaxt="n")

df = round((2*n)/myk)
axis(1, at = 1:m, labels = df)
axis(3, at = 1:m, labels = myk)

points(1:m, test_err_knn, col="red", pch=1);
lines(1:m, test_err_knn, col="red", lty=1);
points(1:m, train_err_knn, col="blue", pch=1);
lines(1:m, train_err_knn, col="blue", lty=2);

points(3, train_err_ls, pch=2, cex=2, col="blue")
points(3, test_err_ls, pch=2, cex=2, col="red")

abline(test_err_Bayes, 0, col="purple")

legend("bottomleft", pch = c(NA, NA, 2, 2, NA), col = c("red", "blue", "red", "blue", "purple"),
       lty=c(1, 2, NA, NA, 1),
```

```
      legend = c("Test Error KNN", "Train Error KNN", "Test Error LS", "Train Error LS", "Bayes Error")
```