ML exams report

SC63576

Assignment 1

Reading Data

```
data <- read.csv("olive.csv")
```

Task 1

```
df1 <- data.frame(scale(data[,-c(1,2)])) #scaled data for fatty acid variables

cov_matrix <- cov(df1)
eig <- eigen(cov_matrix)

counter <- which(cumsum(eig$values/sum(eig$values)*100) >= 90)

# prob_var <- sprintf("%2.3f", eig$values/sum(eig$values)*100)
```

```
paste("The principle components that are needed to explain 90% variaton of the data are", counter[1],".
```

```
## [1] "The principle components that are needed to explain 90% variaton of the data are 4 ."
```

The principle components that are needed to explain 90% variation of the data are 4. Scaling the data is necessary for consistency to have the same content and format. In addition, scaled values help track data that is not easy to compare otherwise.

Task 2

```
pca <- princomp(df1) # implementing PCA
components <- pca$scores # get a visual of the scores of all principal component scores.

three_components <- data.frame(components[,c(1:3)]) # get the first three principal component scores.

df2 <- cbind(data$Region, three_components) # data for the logistic regression model
colnames(df2) <- c("Region", "Comp1","Comp2","Comp3")

n <- dim(df2)
set.seed(12345)
id2 <- sample(1:n,floor(n*0.5))
train2 <- df2[id2,] # 50% train data
test2 <- df2[-id2,] # 50% test data

library("nnet")
log_reg <- multinom(Region~., data = train2) #fitting the logistic regression model
```

```
## # weights:  15 (8 variable)
## initial  value 314.203115
## iter  10 value 88.100326
## iter  20 value 52.996752
## final  value 52.987997
## converged
```

```
misclass <- function(actual_val,predicted_val){
  conf_mat <- table(actual_val,predicted_val)
  n <- length(actual_val)
  error <- 1 - sum((diag(conf_mat))/n)
  return(error)
}

pred <- predict(log_reg)

train_error <- misclass(train2$Region, pred)
test_error <- misclass(test2$Region, pred)

df_error <- data.frame("Train error" = train_error,
                  "Test error" = test_error)

row.names(df_error) <- c("Misclassification Rates")
knitr::kable(df_error)
```

	Train.error	Test.error
Misclassification Rates	0.0594406	0.6258741

The misclassification rate of the training data set is approximately 6%, which means that the quality of prediction is very good compared to the misclassification rate of the test data, which is almost 60% and that means that more than half of the Region values are predicted wrong.

```
print(log_reg)
```

```
## Call:
## multinom(formula = Region ~ ., data = train2)
##
## Coefficients:
##   (Intercept)     Comp1    Comp2     Comp3
## 2   -3.218617 -2.992163 3.187718 -4.000276
## 3   -5.948747 -5.677377 2.827254 -3.045757
##
## Residual Deviance: 105.976
## AIC: 121.976
```

The decision boundary of the first class equals:

$$-3.218617 - 2.992163Comp1 + 3.187718Comp2 - 4.000276Comp3$$

The decision boundary of the second class equals:

$$-5.948747 - 5.677377Comp1 + 2.827254Comp2 - 3.045757Comp3$$

Task 3

```
Y <- data.frame(ifelse(data$Region == 1, "South", "Other")) # construct target Y
colnames(Y) <- c("Y")

df3 <- cbind(Y,data[,-c(1,2)]) # data for logistic regression
df3$Y <- as.factor(df3$Y)

n <- dim(df3)
set.seed(12345)
id3 <- sample(1:n,floor(n*0.5))
train3 <- df3[id3,] # 50% train data
test3 <- df3[-id3,] # 50% test data

library("glmnet")
```

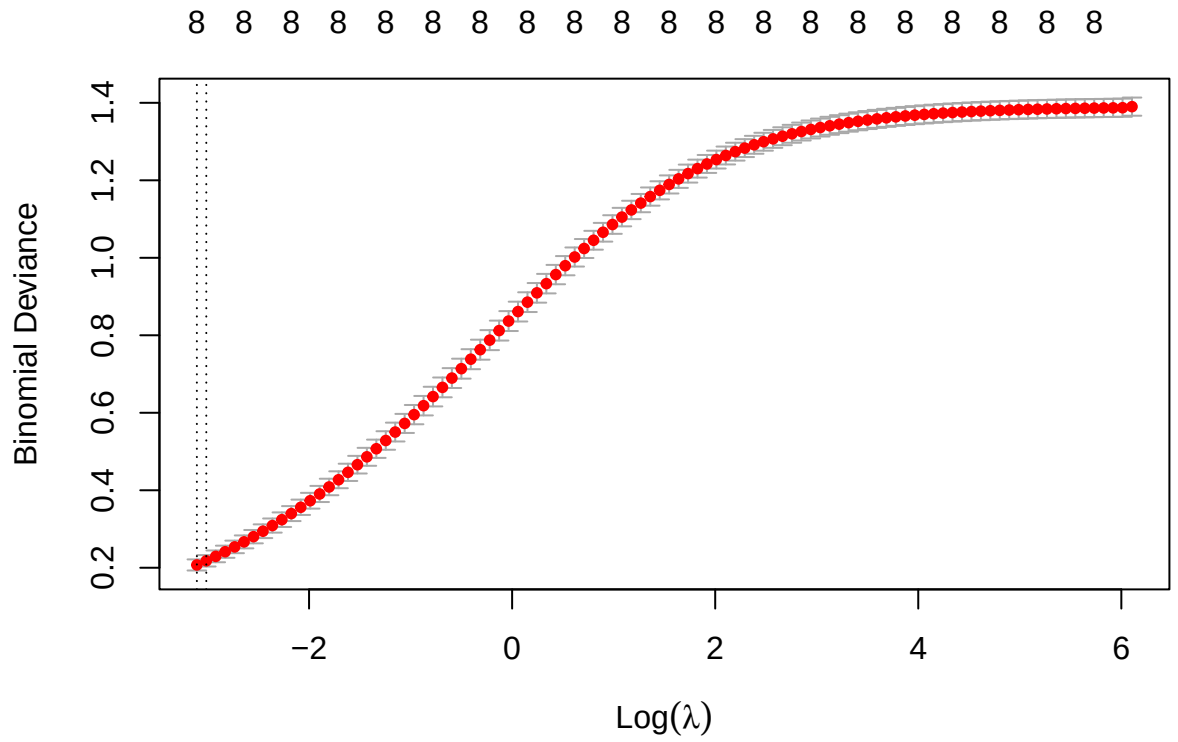
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-2
```

```
Y <- train3$Y
X <- train3[,-c(1)]

ridge <- glmnet(as.matrix(X), as.matrix(Y), alpha = 0, family = "binomial") # ridge regression

cv_ridge <- cv.glmnet(as.matrix(X), as.matrix(Y), alpha = 0, family = "binomial") # cross validation ri
penalty <- cv_ridge$lambda.min
cat("The optimal penalty factor value is", penalty)
```

```
## The optimal penalty factor value is 0.04481765
```

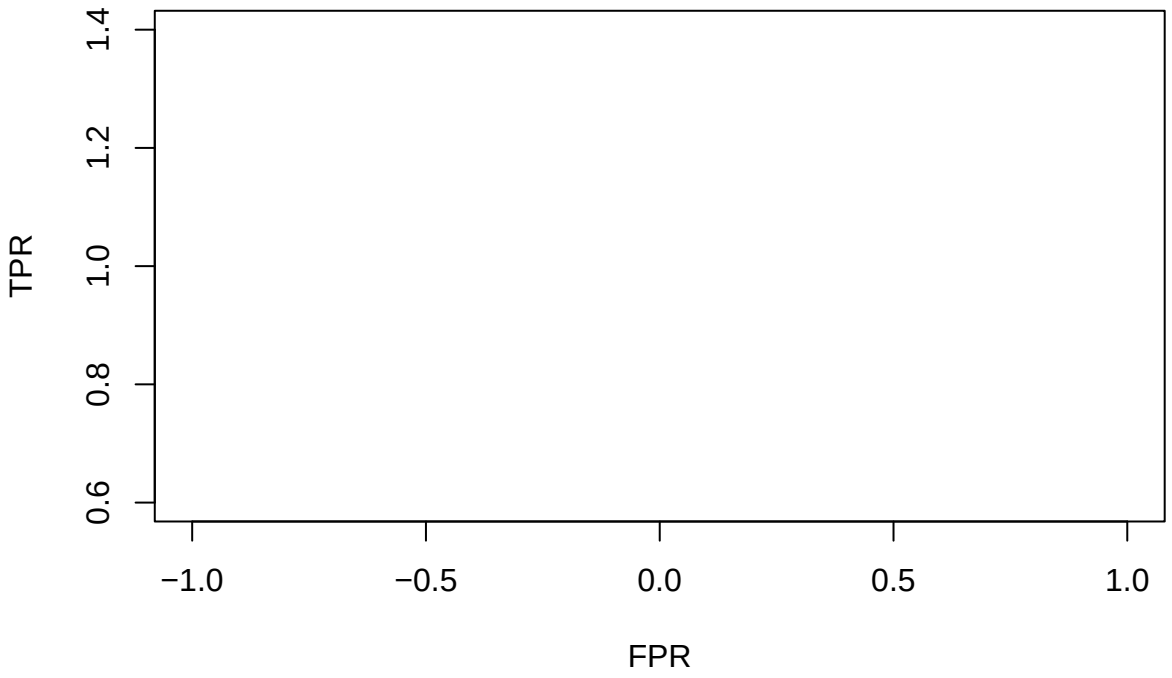
```
plot(cv_ridge)
```



For the dependence of the cross-validation score on the penalty factor, it could be assumed that as the value of the $log(\lambda)$ increases the binomial deviance increases as well and the confidence bounce of each $log(\lambda)$ is the same.

```
log_reg2 <- glm(Y~., data = train3, family = binomial, control = list(penalty)) # fitting logistic regr
```

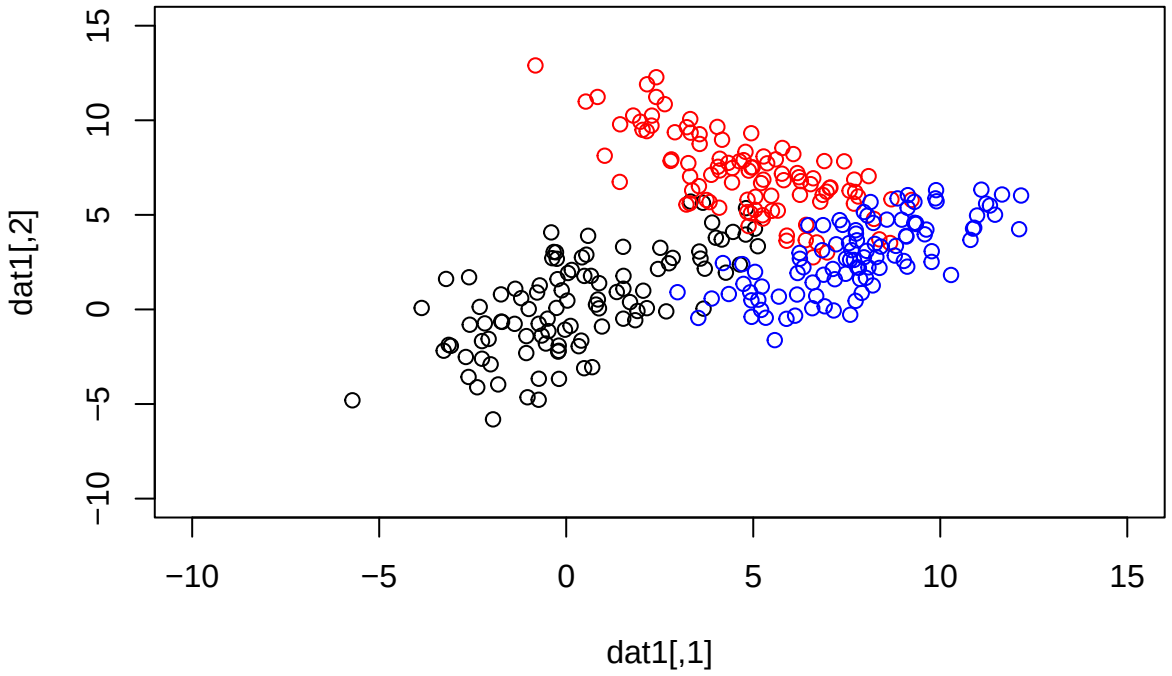
```
mySeq <- seq(0.05,0.95,0.05)

TPR <- numeric(length = length(mySeq))
FPR <- numeric(length = length(mySeq))

for (i in 1:length(mySeq)){
  p <- as.factor(ifelse( predict(log_reg2,
                                 newdata = test3,
                                 type = "response") > mySeq[i],
                         "South","Other"))

  conf_mat <- table(test3$Y,p)

  TPR[i] <- conf_mat[1,1]/(conf_mat[1,1] + conf_mat[1,2])
  FPR[i] <- conf_mat[2,1]/(conf_mat[2,1] + conf_mat[2,2])
}

df_plot <- data.frame("TPR" = TPR,
                      "FPR" = FPR)

plot(FPR,TPR, type = "l", col = "navy")
```



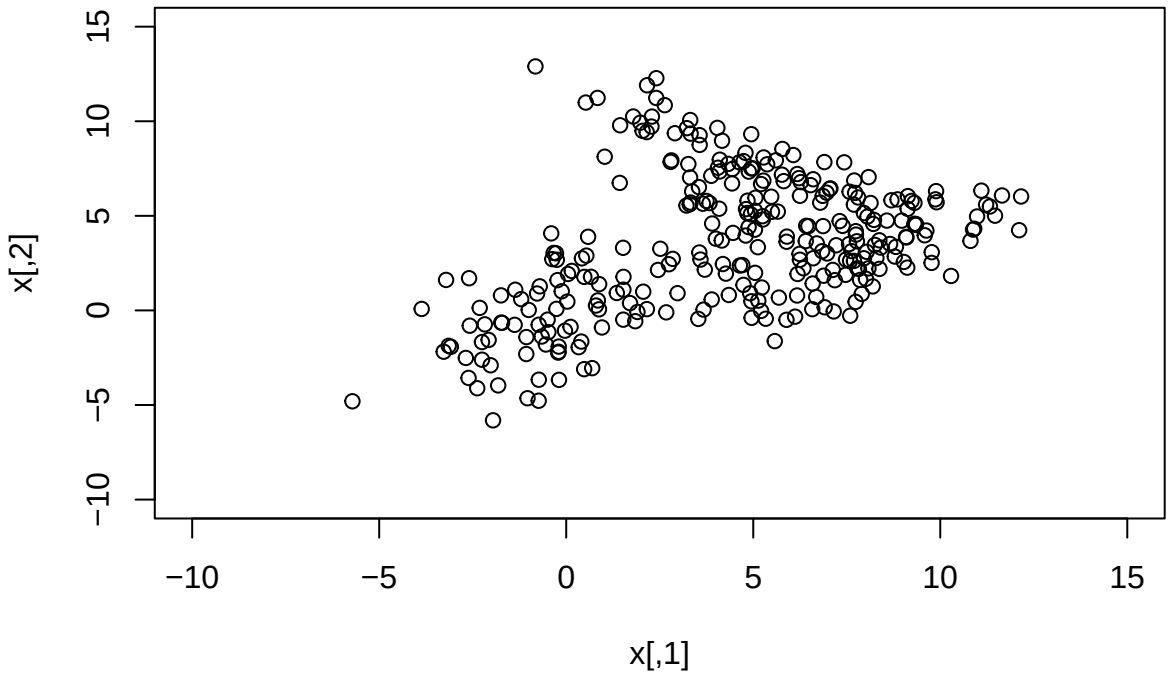
Assignment 2

Gaussian Mixture Models

```
library(mvtnorm)
set.seed(1234567890)
min_change <- 0.001 # min parameter change between two consecutive iterations
N=300 # number of training points
D=2 # number of dimensions
x <- matrix(nrow=N, ncol=D) # training data# Producing the training data
mu1<-c(0,0)
Sigma1 <- matrix(c(5,3,3,5),D,D)
dat1<-rmvnorm(n = 100, mu1, Sigma1)
mu2<-c(5,7)
Sigma2 <- matrix(c(5,-3,-3,5),D,D)
dat2<-rmvnorm(n = 100, mu2, Sigma2)
mu3<-c(8,3)
Sigma3 <- matrix(c(3,2,2,3),D,D)
dat3<-rmvnorm(n = 100, mu3, Sigma3)
plot(dat1,xlim=c(-10,15),ylim=c(-10,15))
points(dat2,col="red")
points(dat3,col="blue")
```




```
x[1:100,]<-dat1
x[101:200,]<-dat2
x[201:300,]<-dat3
plot(x,xlim=c(-10,15),ylim=c(-10,15))
```



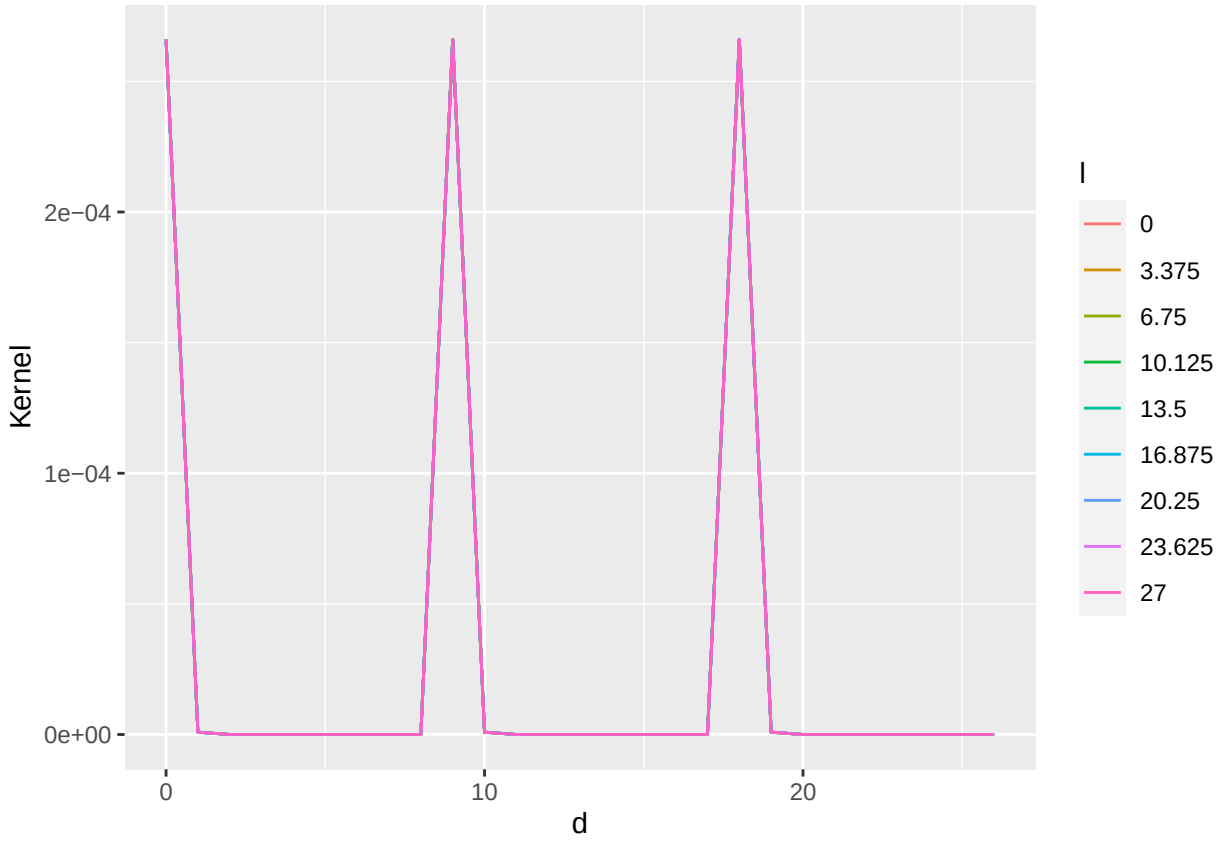

```
K=3 # number of classes
w <- matrix(nrow=N, ncol=K) # fractional class assignments
pi <- vector(length=K) # mixing coefficients
mu <- matrix(nrow=K, ncol=D) # class conditional means
Sigma <- array(dim=c(D,D,K)) # class conditional covariances
```

Kernel Models

Class 1

```
set.seed(1234567890)

N_class1 <- 1500
data_class1 <- NULL

for(i in 1:N_class1){
  a <- rbinom(n = 1, size = 1, prob = 0.3)
  b <- rnorm(n = 1, mean = 15, sd = 3) * a + (1-a) * rnorm(n = 1, mean = 4, sd = 2)
  data_class1 <- c(data_class1,b)
}

h1 <- seq(0,max(data_class1),by = 1)

m1 <- 9
mySeq1 <- seq(0,27, length.out = m1)

n1 <- 1500
results1 <- lapply(mySeq1, function(l) (1/n1) * sum((dnorm(x = l, mean = 0, sd = 1))))

# plot(1/n1 * dnorm(x =mySeq1, mean = 0, sd = 1))

plot_df1 <- data.frame("Kernel" = unlist(results1),
```

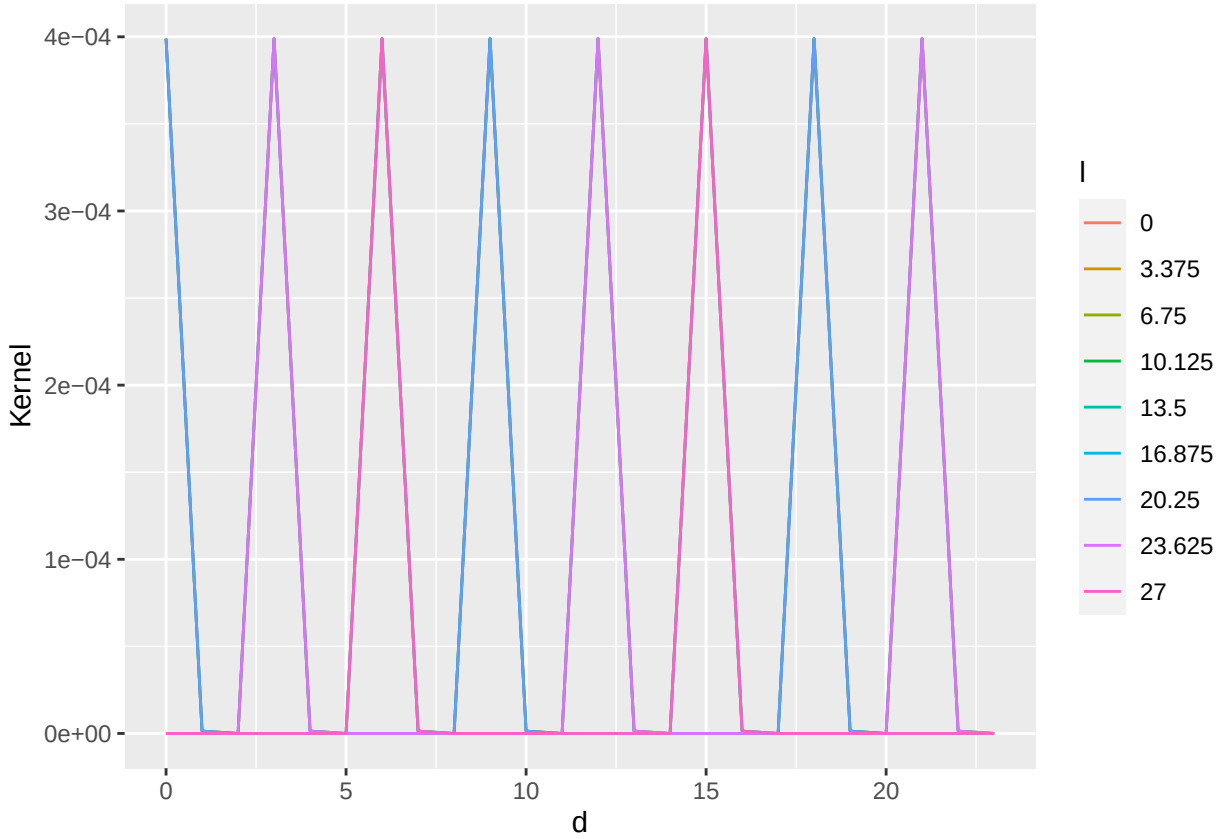
```
                        "l" = rep(mySeq1, each = length(h1)),
                        "d" = rep(h1, times = length(m1)))

plot_df1$l <- as.factor(plot_df1$l)

library("ggplot2")
ggplot(plot_df1, aes(x=d, y=Kernel, col = l)) +
  geom_line()
```




```
k1 <- (1/n1) * sum((dnorm(x = 3.375, mean = 0, sd = 1)))
```

```
N_class2 <- 1000
data_class2 <- NULL

for(i in 1:N_class2){
  a <- rbinom(n = 1, size = 1, prob = 0.1)
  b <- rnorm(n = 1, mean = 10, sd = 5) * a + (1-a) * rnorm(n = 1, mean = 15, sd = 2)
  data_class2 <- c(data_class2,b)
}

h2 <- seq(0,max(data_class2),by = 1)

m2 <- 9
mySeq2 <- seq(0,27, length.out = m2)

n2 <- 1000
results2 <- lapply(mySeq2, function(l) (1/n2) * sum((dnorm(x = l, mean = 0, sd = 1))))
```

```r
# plot(1/n2 * dnorm(x =mySeq2, mean = 0, sd = 1))


plot_df2 <- data.frame("Kernel" = unlist(results2),
                       "l" = rep(mySeq2, each = length(h2)),
                       "d" = rep(h2, times = length(m2)))

plot_df2$l <- as.factor(plot_df2$l)

library("ggplot2")
ggplot(plot_df2, aes(x=d, y=Kernel, col = l)) +
  geom_line()
```




```r
k2 <- (1/n2) * sum((dnorm(x = 3.375, mean = 0, sd = 1)))
```