Introduction to Machine Learning

Lab 2 Block 2

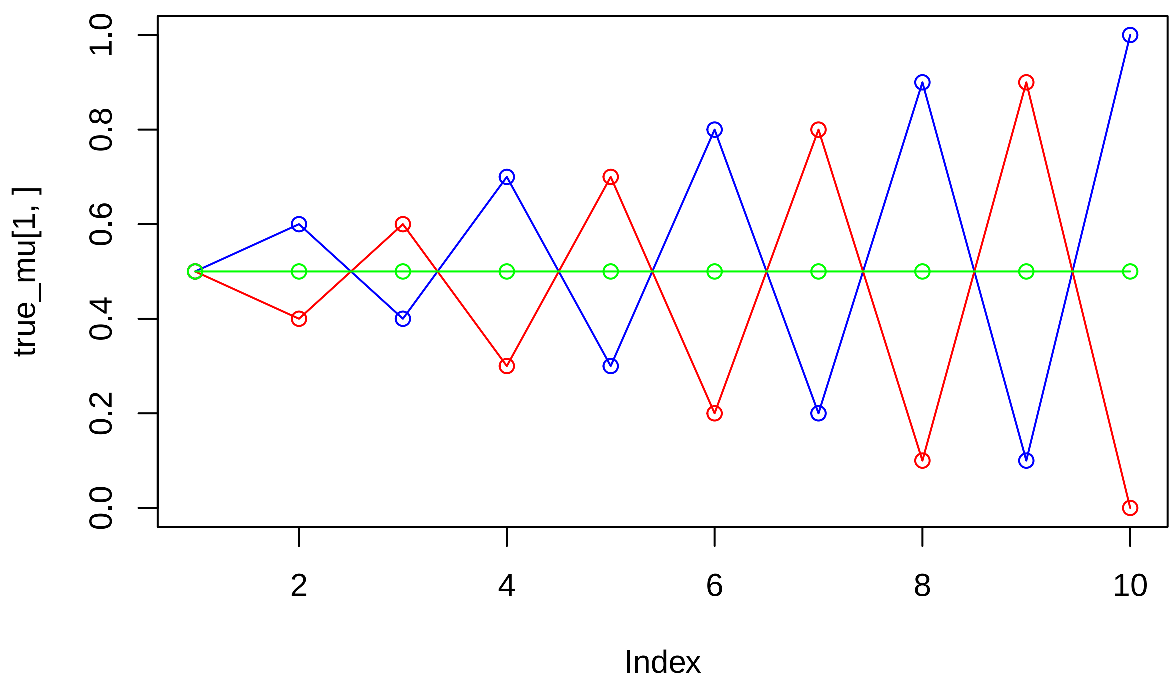
Rasmus Holm

2016-12-01

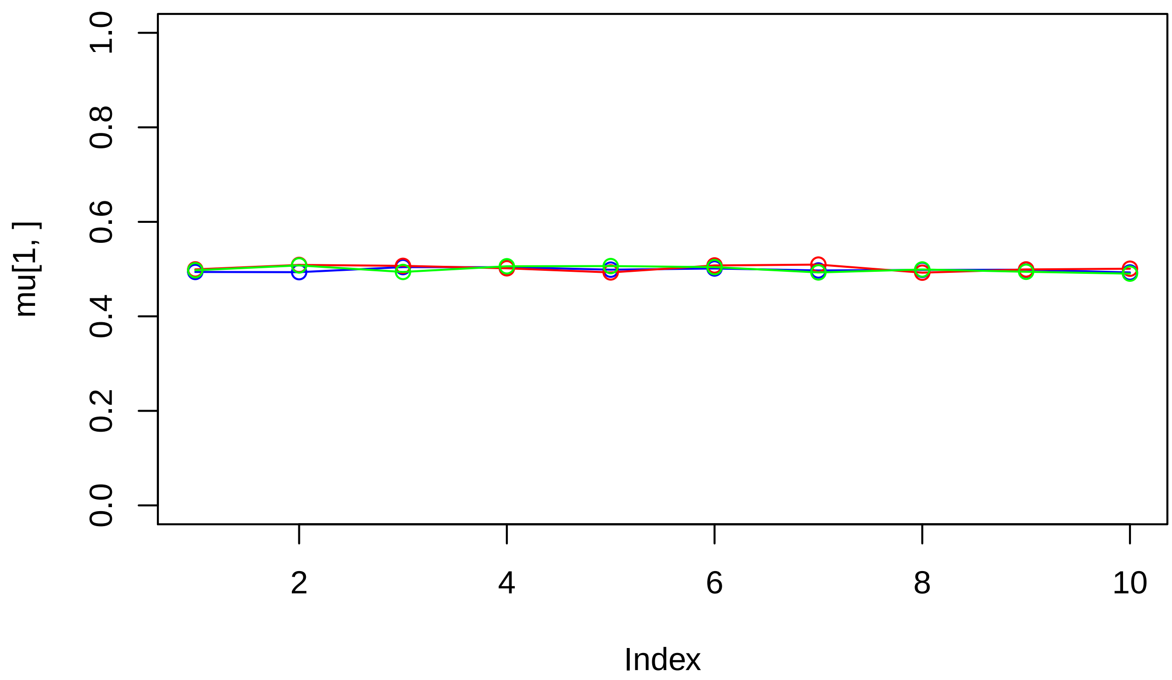
Contents

Assignment 1a

Assignment 1b



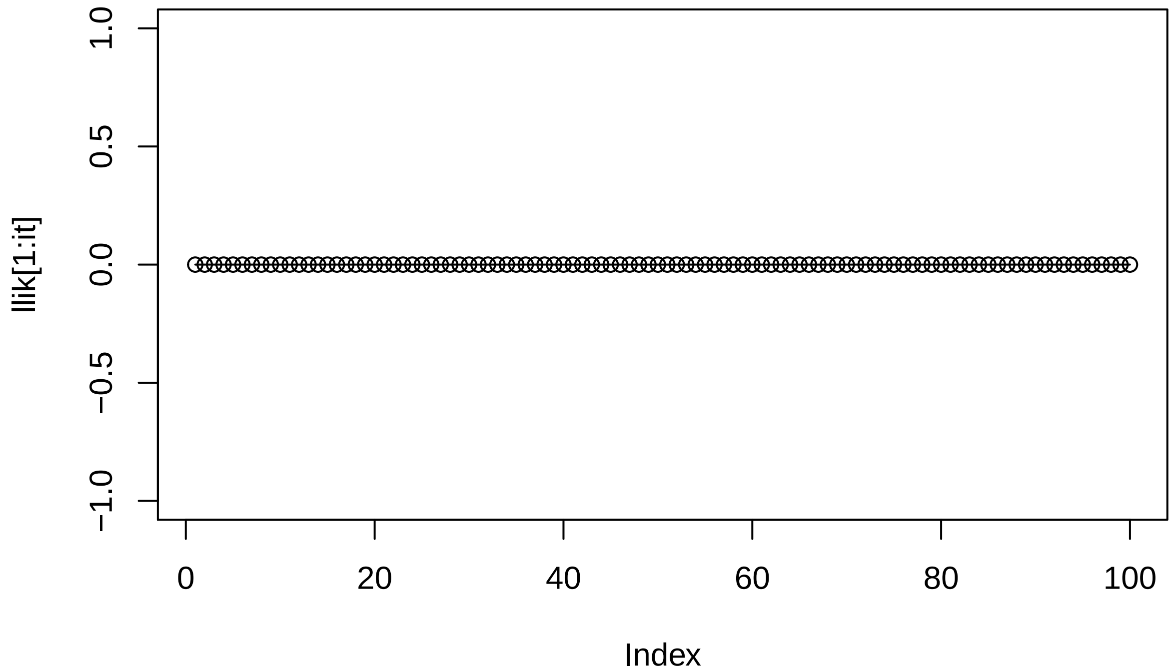

```
#> [1] 0.3326090 0.3336558 0.3337352
#>           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
#> [1,] 0.4939877 0.4935375 0.5042511 0.5040286 0.4987810 0.5012754 0.4971036
#> [2,] 0.4993719 0.5088453 0.5068730 0.5016720 0.4929275 0.5077146 0.5095075
#> [3,] 0.4975302 0.5077926 0.4939841 0.5059821 0.5063490 0.5041462 0.4929400
#>           [,8]      [,9]     [,10]
#> [1,] 0.4982144 0.4987654 0.4929075
#> [2,] 0.4924574 0.4992470 0.5008651
#> [3,] 0.4992362 0.4943482 0.4903974
```




```
#> iteration:  1 log likelihood:  FALSE
#> iteration:  2 log likelihood:  FALSE
#> iteration:  3 log likelihood:  FALSE
#> iteration:  4 log likelihood:  FALSE
#> iteration:  5 log likelihood:  FALSE
#> iteration:  6 log likelihood:  FALSE
#> iteration:  7 log likelihood:  FALSE
#> iteration:  8 log likelihood:  FALSE
#> iteration:  9 log likelihood:  FALSE
#> iteration:  10 log likelihood:  FALSE
#> iteration:  11 log likelihood:  FALSE
#> iteration:  12 log likelihood:  FALSE
#> iteration:  13 log likelihood:  FALSE
#> iteration:  14 log likelihood:  FALSE
#> iteration:  15 log likelihood:  FALSE
#> iteration:  16 log likelihood:  FALSE
#> iteration:  17 log likelihood:  FALSE
#> iteration:  18 log likelihood:  FALSE
#> iteration:  19 log likelihood:  FALSE
#> iteration:  20 log likelihood:  FALSE
#> iteration:  21 log likelihood:  FALSE
#> iteration:  22 log likelihood:  FALSE
#> iteration:  23 log likelihood:  FALSE
#> iteration:  24 log likelihood:  FALSE
#> iteration:  25 log likelihood:  FALSE
#> iteration:  26 log likelihood:  FALSE
```

```
#> iteration:  27 log likelihood:  FALSE
#> iteration:  28 log likelihood:  FALSE
#> iteration:  29 log likelihood:  FALSE
#> iteration:  30 log likelihood:  FALSE
#> iteration:  31 log likelihood:  FALSE
#> iteration:  32 log likelihood:  FALSE
#> iteration:  33 log likelihood:  FALSE
#> iteration:  34 log likelihood:  FALSE
#> iteration:  35 log likelihood:  FALSE
#> iteration:  36 log likelihood:  FALSE
#> iteration:  37 log likelihood:  FALSE
#> iteration:  38 log likelihood:  FALSE
#> iteration:  39 log likelihood:  FALSE
#> iteration:  40 log likelihood:  FALSE
#> iteration:  41 log likelihood:  FALSE
#> iteration:  42 log likelihood:  FALSE
#> iteration:  43 log likelihood:  FALSE
#> iteration:  44 log likelihood:  FALSE
#> iteration:  45 log likelihood:  FALSE
#> iteration:  46 log likelihood:  FALSE
#> iteration:  47 log likelihood:  FALSE
#> iteration:  48 log likelihood:  FALSE
#> iteration:  49 log likelihood:  FALSE
#> iteration:  50 log likelihood:  FALSE
#> iteration:  51 log likelihood:  FALSE
#> iteration:  52 log likelihood:  FALSE
#> iteration:  53 log likelihood:  FALSE
#> iteration:  54 log likelihood:  FALSE
#> iteration:  55 log likelihood:  FALSE
#> iteration:  56 log likelihood:  FALSE
#> iteration:  57 log likelihood:  FALSE
#> iteration:  58 log likelihood:  FALSE
#> iteration:  59 log likelihood:  FALSE
#> iteration:  60 log likelihood:  FALSE
#> iteration:  61 log likelihood:  FALSE
#> iteration:  62 log likelihood:  FALSE
#> iteration:  63 log likelihood:  FALSE
#> iteration:  64 log likelihood:  FALSE
#> iteration:  65 log likelihood:  FALSE
#> iteration:  66 log likelihood:  FALSE
#> iteration:  67 log likelihood:  FALSE
#> iteration:  68 log likelihood:  FALSE
#> iteration:  69 log likelihood:  FALSE
#> iteration:  70 log likelihood:  FALSE
#> iteration:  71 log likelihood:  FALSE
#> iteration:  72 log likelihood:  FALSE
#> iteration:  73 log likelihood:  FALSE
#> iteration:  74 log likelihood:  FALSE
#> iteration:  75 log likelihood:  FALSE
#> iteration:  76 log likelihood:  FALSE
#> iteration:  77 log likelihood:  FALSE
#> iteration:  78 log likelihood:  FALSE
#> iteration:  79 log likelihood:  FALSE
#> iteration:  80 log likelihood:  FALSE
```

```
#> iteration:  81 log likelihood:  FALSE
#> iteration:  82 log likelihood:  FALSE
#> iteration:  83 log likelihood:  FALSE
#> iteration:  84 log likelihood:  FALSE
#> iteration:  85 log likelihood:  FALSE
#> iteration:  86 log likelihood:  FALSE
#> iteration:  87 log likelihood:  FALSE
#> iteration:  88 log likelihood:  FALSE
#> iteration:  89 log likelihood:  FALSE
#> iteration:  90 log likelihood:  FALSE
#> iteration:  91 log likelihood:  FALSE
#> iteration:  92 log likelihood:  FALSE
#> iteration:  93 log likelihood:  FALSE
#> iteration:  94 log likelihood:  FALSE
#> iteration:  95 log likelihood:  FALSE
#> iteration:  96 log likelihood:  FALSE
#> iteration:  97 log likelihood:  FALSE
#> iteration:  98 log likelihood:  FALSE
#> iteration:  99 log likelihood:  FALSE
#> iteration:  100 log likelihood:  FALSE
#> [1] 0.3326090 0.3336558 0.3337352
#>           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
#> [1,] 0.4939877 0.4935375 0.5042511 0.5040286 0.4987810 0.5012754 0.4971036
#> [2,] 0.4993719 0.5088453 0.5068730 0.5016720 0.4929275 0.5077146 0.5095075
#> [3,] 0.4975302 0.5077926 0.4939841 0.5059821 0.5063490 0.5041462 0.4929400
#>           [,8]      [,9]     [,10]
#> [1,] 0.4982144 0.4987654 0.4929075
#> [2,] 0.4924574 0.4992470 0.5008651
#> [3,] 0.4992362 0.4943482 0.4903974
```


1.0
0.5
0.0
−0.5
−1.0
llik[1:it]
0
20
40
60
80
100
Index

Assignment 2a

1

```
#> [1] 37.10301
```

2

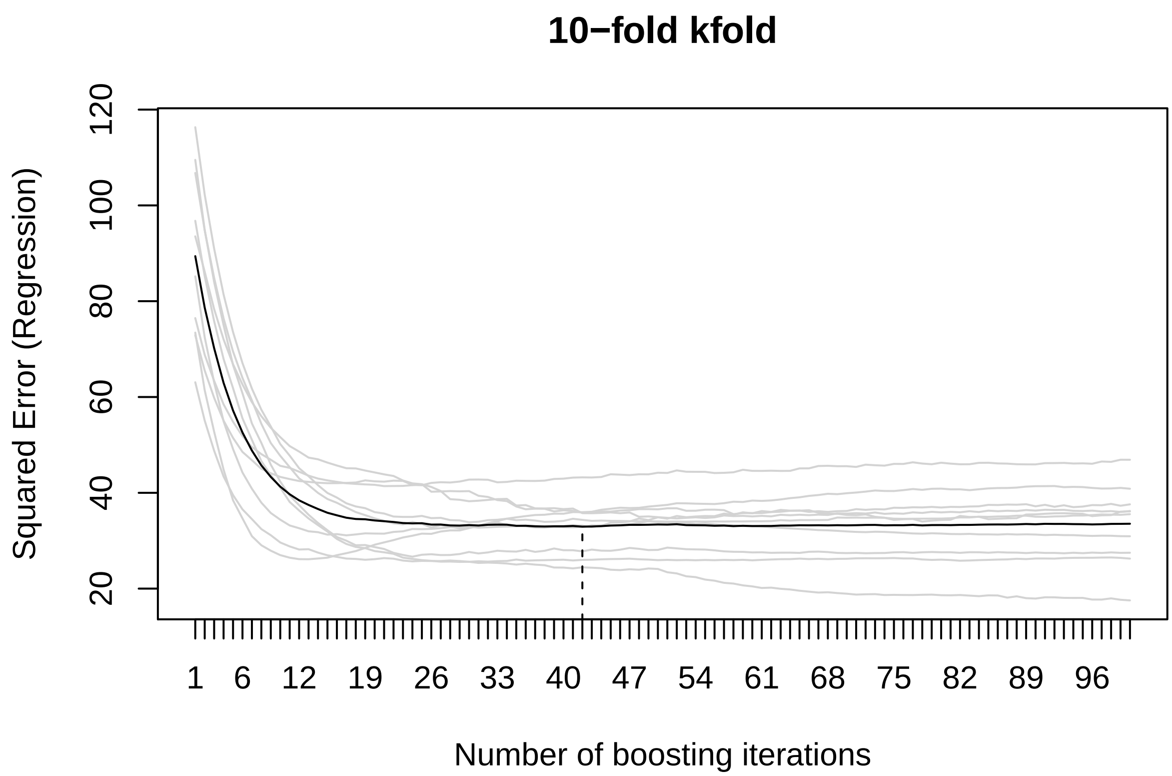
```
#> [1] 30.8038
```

3

Assignment 2b

Assignment 3a

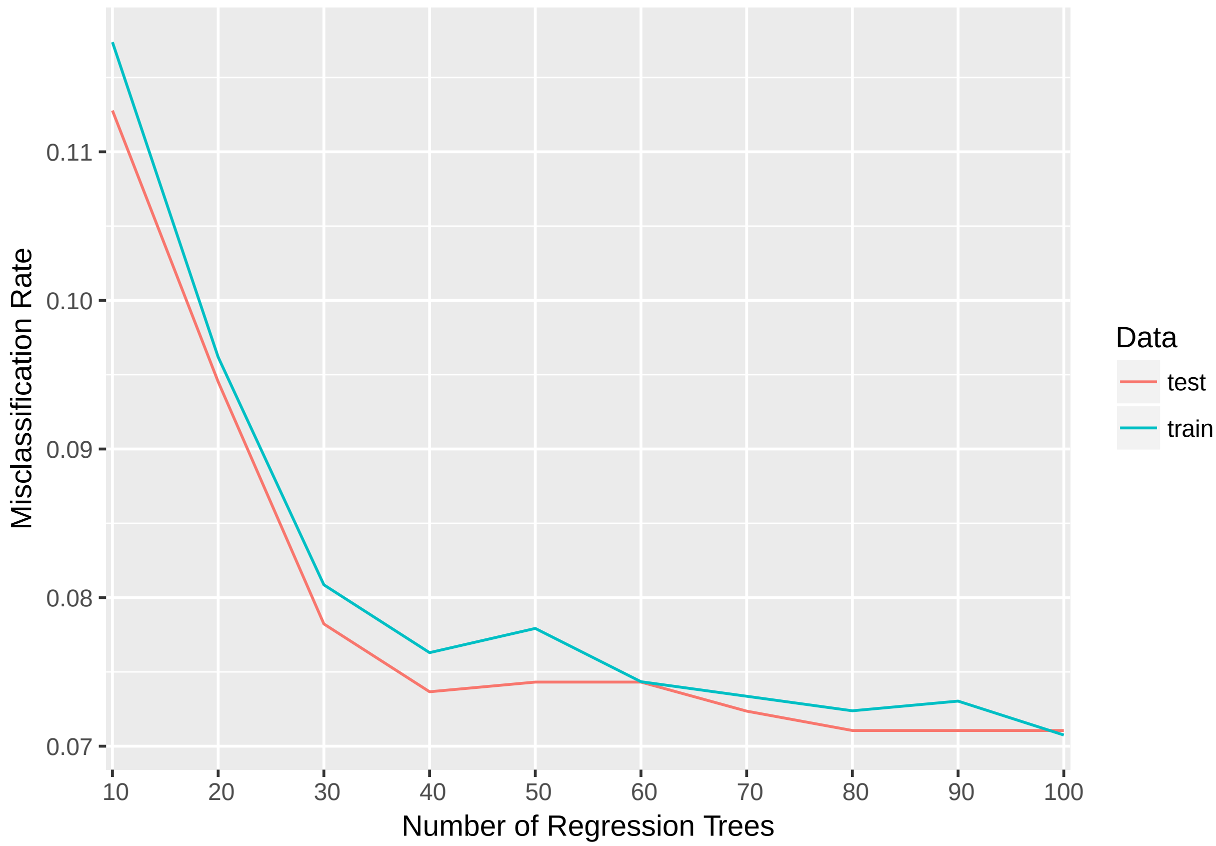
1

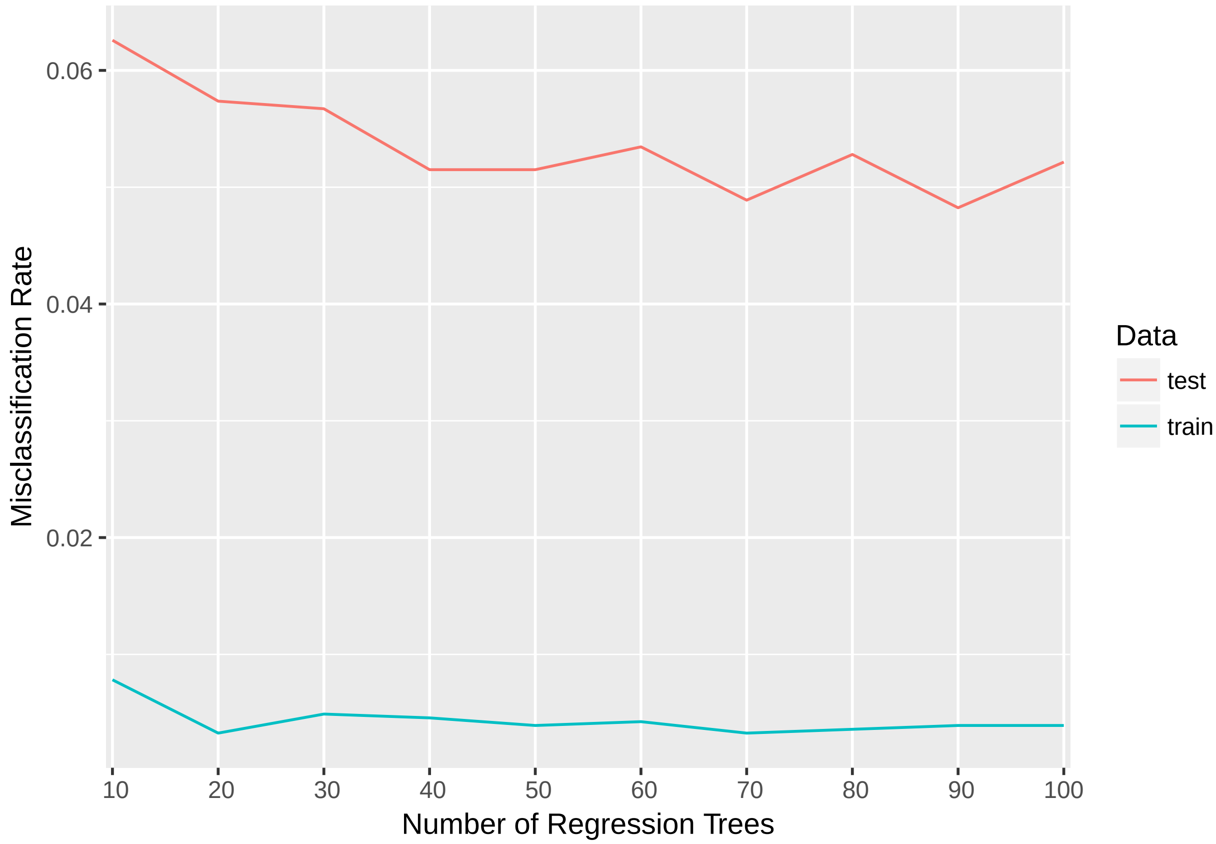


2

```
#> [1] 988.8233
#> [1] 1433.941
```

Assignment 4a




0.06
0.04
0.02
Misclassification Rate
10
20
30
40
50
60
70
80
90
100
Number of Regression Trees
Data
test
train

Appendix

Code for Assignment 1a

Code for Assignment 1b

Code for Assignment 2a

```
library(tree)

data <- read.csv2("../data/bodyfatregression.csv")
names(data) <- c("Waist", "Weight", "Bodyfat")

set.seed(1234567890)
train_idx <- sample(nrow(data), floor(nrow(data) * (2 / 3)))
train <- data[train_idx,]
test <- data[-train_idx,]
set.seed(1234567890)

tree_count <- 100
test_errors <- rep(0, tree_count)

for (i in 1:tree_count) {
    newdata <- train[sample(nrow(train), replace=TRUE),]
    fit <- tree(Bodyfat ~ ., data=newdata, split="deviance")
    test_error <- mean((predict(fit, test) - test$Bodyfat)^2)
    test_errors[i] <- test_error
}

mean(test_errors)
set.seed(1234567890)

tree_count <- 100
k <- 3
errors <- rep(0, tree_count * k)

for (i in 1:tree_count) {
    newdata <- data[sample(nrow(data), replace=TRUE),]
    datasets <- suppressWarnings(split(newdata, 1:k))

    train1 <- rbind(datasets[[1]], datasets[[2]])
    test1 <- datasets[[3]]

    train2 <- rbind(datasets[[1]], datasets[[3]])
    test2 <- datasets[[2]]

    train3 <- rbind(datasets[[2]], datasets[[3]])
    test3 <- datasets[[1]]

    fit1 <- tree(Bodyfat ~ ., data=train1, split="deviance")
    error1 <- mean((predict(fit1, test1) - test1$Bodyfat)^2)
```

```r
    fit2 <- tree(Bodyfat ~ ., data=train2, split="deviance")
    error2 <- mean((predict(fit2, test2) - test2$Bodyfat)^2)

    fit3 <- tree(Bodyfat ~ ., data=train3, split="deviance")
    error3 <- mean((predict(fit3, test3) - test3$Bodyfat)^2)

    errors[(i - 1) * k + 1] <- error1
    errors[(i - 1) * k + 2] <- error2
    errors[(i - 1) * k + 3] <- error3
}

mean(errors)
bagging.regtrees <- function(formula, data, newdata, b) {
    predictions <- matrix(0, nrow=nrow(newdata), ncol=b)

    for (i in 1:k) {
        bootstrap_sample <- data[sample(nrow(data), replace=TRUE),]
        fit <- tree(formula, data=bootstrap_sample, split="deviance")
        predictions[, i] <- predict(fit, newdata)
    }

    rowMeans(predictions)
}
cv.regtrees <- function(formula, data, newdata, b, k) {
    predictions <- matrix(0, nrow(nrow(newdata)), ncol=b*k)

    for (i in 1:tree_count) {
        bootstrap_sample <- data[sample(nrow(data), replace=TRUE),]
        datasets <- suppressWarnings(split(bootstrap_sample, 1:k))

        train1 <- rbind(datasets[[1]], datasets[[2]])
        test1 <- datasets[[3]]

        train2 <- rbind(datasets[[1]], datasets[[3]])
        test2 <- datasets[[2]]

        train3 <- rbind(datasets[[2]], datasets[[3]])
        test3 <- datasets[[1]]

        fit1 <- tree(Bodyfat ~ ., data=train1, split="deviance")
        prediction1 <- predict(fit1, newdata)

        fit2 <- tree(Bodyfat ~ ., data=train2, split="deviance")
        prediction2 <- predict(fit2, newdata)

        fit3 <- tree(Bodyfat ~ ., data=train3, split="deviance")
        prediction2 <- predict(fit2, newdata)

        predictions[, (i - 1) * k + 1] <- predcition1
        predictions[, (i - 1) * k + 2] <- prediction2
        predictions[, (i - 1) * k + 3] <- prediction3
    }
```

```
    rowMeans(predictions)
}
```

Code for Assignment 2b

Code for Assignment 3a

```
library(mboost)

data <- read.csv2("../data/bodyfatregression.csv")

fit <- blackboost(Bodyfat_percent ~ Waist_cm + Weight_kg, data=data)

cvf <- cv(model.weights(fit), type="kfold")
cvm <- cvrisk(fit, folds=cvf, grid=1:100)
plot(cvm)
set.seed(1234567890)
train_idx <- sample(nrow(data), floor(nrow(data) * (2 / 3)))
train <- data[train_idx,]
test <- data[-train_idx,]

fit <- blackboost(Bodyfat_percent ~ Waist_cm + Weight_kg, data=train,
                  control=boost_control(mstop=mstop(cvm)))
test_error <- sum((predict(fit, test) - test$Bodyfat_percent)^2)
train_error <- sum((predict(fit, train) - train$Bodyfat_percent)^2)

test_error
train_error
```

Code for Assignment 4a

```
library(mboost)
library(randomForest)
library(ggplot2)
library(reshape2)

data <- read.csv2("../data/spambase.csv")
data$Spam <- as.factor(data$Spam)

set.seed(1234567890)
train_idx <- sample(nrow(data), floor(nrow(data) * (2 / 3)))
train <- data[train_idx,]
test <- data[-train_idx,]
tree_counts <- seq(10, 100, by=10)
test_errors <- rep(0, length(tree_counts))
train_errors <- rep(0, length(tree_counts))

for (i in 1:length(tree_counts)) {
    fit <- blackboost(Spam ~ ., data=train, family=AdaExp(),
                      control=boost_control(mstop=tree_counts[i]))
```

```r
    test_error <- 1 - (sum(predict(fit, test, type="class") == test$Spam) / nrow(test))
    train_error <- 1 - (sum(predict(fit, train, type="class") == train$Spam) / nrow(train))
    test_errors[i] <- test_error
    train_errors[i] <- train_error
}
plot_data <- data.frame(Trees=tree_counts, test=test_errors, train=train_errors)
plot_data <- melt(plot_data, id="Trees", value.name="Error", variable.name="Data")

ggplot(plot_data) +
    xlab("Number of Regression Trees") +
    ylab("Misclassification Rate") +
    geom_line(aes(x=Trees, y=Error, color=Data)) +
    scale_x_discrete(limits=tree_counts)
test_errors <- rep(0, length(tree_counts))
train_errors <- rep(0, length(tree_counts))

for (i in 1:length(tree_counts)) {
    fit <- randomForest(Spam ~ ., data=train, ntree=tree_counts[i])
    test_error <- 1 - (sum(predict(fit, test, type="class") == test$Spam) / nrow(test))
    train_error <- 1 - (sum(predict(fit, train, type="class") == train$Spam) / nrow(train))
    test_errors[i] <- test_error
    train_errors[i] <- train_error
}
plot_data <- data.frame(Trees=tree_counts, test=test_errors, train=train_errors)
plot_data <- melt(plot_data, id="Trees", value.name="Error", variable.name="Data")

ggplot(plot_data) +
    xlab("Number of Regression Trees") +
    ylab("Misclassification Rate") +
    geom_line(aes(x=Trees, y=Error, color=Data)) +
    scale_x_discrete(limits=tree_counts)
```