```r
###############################################################################
## Course: Machine Learning for Economists and Business Analysts
## Topic: Trees and Forest

###############################################################################

# Install packages
# install.packages("grf")
# install.packages("rpart")
# install.packages("rpart.plot")
# install.packages("glmnet")
# install.packages("DiagrammeR")

# Load packages
library(grf)
library(rpart)
library(rpart.plot)
library(glmnet)
library(DiagrammeR)

getwd()
setwd("C:/Users/user/Dropbox/PhD Kurs Basel 2020/PC 2/")

# Load data
data_2006 <-read.csv("browser_2006.csv", sep = ",")
data_new <-read.csv("browser_new.csv", sep = ",")

####################################################
### Exercise 1: Data Description and Preparation ###
####################################################

###############################################################################
# Task 1
###############################################################################

# Data preparation
y_2006 <- as.matrix(data_2006[,2])
x_2006 <- as.matrix(data_2006[,c(3:ncol(data_2006))])
x_new <- as.matrix(data_new[,c(2:ncol(data_new))])

id_2006 <- as.matrix(data_2006[,1])
id_new <- as.matrix(data_new[,1])


###############################################################################
# Task 2
###############################################################################

# Average spending
mean(y_2006) # 2064.595

###############################################################################
# Task 3
###############################################################################

# Average grade
x_2006[id_2006==921,x_2006[id_2006==921,] == max(x_2006[id_2006==921,])]
#yahoo.com 21.87436

###############################################################################
# Task 4
###############################################################################

log_y_2006 = as.matrix(log(y_2006))

# Cumulative Distribution of Spending
plot(ecdf(y_2006), xlab = "Spending in US-Dollars", sub = "(Truncated at 20,000 US-Dollars)", ylab = "cdf", main =
"Distribution of Spending", xlim= c(0,20000))

# Cumulative Distribution of Log Spendiung
plot(ecdf(log_y_2006), xlab = "log Spending", ylab = "cdf", main = "Distribution of Log Spending")

###############################################################################
# Task 5
###############################################################################

set.seed(123456789)
# Generate variable with the rows in training data
size <- floor(0.5 * nrow(data_2006))
training_set <- sample(seq_len(nrow(data_2006)), size = size)

##############################################################################
##############################################################################
```

```
#########################
### Exercise 2: Trees ###
#########################

###############################################################################
# Task 1
###############################################################################

# Prepare data for tree estimator
outcome <- log_y_2006[training_set]
tree_data_2006 <-  data.frame(outcome, x_2006[training_set,])

# Build shallow tree
shallow_tree <- rpart(formula = outcome ~., data = tree_data_2006, method = "anova",
                             y = TRUE, control = rpart.control(cp = 0.00002, minbucket=100))
# Note: 'minbucket=100' imposes the restriction that each terminal leave should contain at least 100 used cars.
# The algorithm 'rpart' stops growing trees when either one leave has less than 100 observations or the MSE gain of
addidng one addidtional leave is below cp=0.00002.

## Plot tree structure
rpart.plot(shallow_tree,digits=3)

###############################################################################
# Task 2
###############################################################################

# Build deep tree
set.seed(1001)
deep_tree <- rpart(formula = outcome ~., data = tree_data_2006, method = "anova", xval = 10,
                      y = TRUE, control = rpart.control(cp = 0.00002, minbucket=10))

print('Relative CV-MSE for different tree sizes')
print(deep_tree$cptable)

# Plot CV-MSE
plotcp(deep_tree)

###############################################################################
# Task 3
###############################################################################

op.index <- which.min(deep_tree$cptable[, "xerror"])
print(paste0("Optimal number final leaves: ", op.index))

###############################################################################
# Task 4
###############################################################################

## Select the Tree that Minimises CV-MSE
# Get cp-value that corresponds to optimal tree size
cp.vals <- deep_tree$cptable[op.index, "CP"]

# Prune the deep tree
pruned_tree <- prune(deep_tree, cp = cp.vals)

## Plot tree structure
rpart.plot(pruned_tree,digits=3)

###############################################################################
# Task 5
###############################################################################

## Assess Out-of-Sample Performance
# Predict log online spending
pred_tree <- predict(pruned_tree, newdata= as.data.frame(x_2006))

# R-squared
MSE_tree <- mean(((log_y_2006[-training_set])-pred_tree[-training_set])^2)
r2_tree <-  1- MSE_tree/var(log_y_2006[-training_set])
print(r2_tree)


##############################################################################
##############################################################################


###########################
### Exercise 3: Forests ###
###########################

###############################################################################
# Task 1
###############################################################################
```

```
rep <- 1000 # number of trees
cov <- 1/3 # share of covariates
frac <- 1/2 # fraction of subsample
min_obs <- 100 # max. size of terminal leaves in trees

# Build Forest
set.seed(10001)
forest1 <- regression_forest(x_2006[training_set,],log_y_2006[training_set,],
                            mtry = floor(cov*ncol(x_2006)), sample.fraction = frac, num.trees = rep,
                            min.node.size = min_obs, honesty=FALSE)

# Plot a tree of the forest
tree <- get_tree(forest1,95)
plot(tree)

# Plot the variable importantance
imp1 <- variable_importance(forest1, max.depth = 1)
print(cbind(colnames(x_2006[,imp1>0.02]),imp1[imp1>0.02]))


imp2 <- variable_importance(forest1, decay.exponent = 2, max.depth = 4)
print(cbind(colnames(x_2006[,imp2>0.02]),imp2[imp2>0.02]))

# Prediction
fit <- predict(forest1, newdata = x_2006[-training_set,])$predictions

# R-squared
MSE1 <- mean(((log_y_2006[-training_set,])-fit)^2)
r2_forest1 <-  1- MSE1/var(log_y_2006[-training_set,])
print(r2_tree,r2_forest1)

###############################################################################
# Task 2
###############################################################################

## AUC - area under curve
sizes <- c(1000,500,400,300, 200, 50, 20, 10) # Select a grid of sample sizes
# Prepare matrix to store results
auc <- matrix(NA, nrow = length(sizes), ncol = 3)
colnames(auc) <- c("Trees", "AUC", "Marginal AUC")
auc[,1] <- sizes
# Sum of Squares Total
SST <- mean(((y_2006[-training_set,])-(mean(y_2006[-training_set,])))^2)

set.seed(10001) # set starting value
for (t in sizes){
  print(t)
  # Estimate Forests
  forest <- regression_forest(x_2006[training_set,],(log_y_2006[training_set,]), mtry = floor(cov*ncol(x_2006)),
                              sample.fraction = frac, num.trees = t, min.node.size = min_obs, honesty=FALSE)
  fit <- predict(forest, newdata = x_2006[-training_set,])$predictions # prediction in test sample
  auc[auc[,1]== t,2] <- 1- mean(((log_y_2006[-training_set,])-fit)^2)/var(log_y_2006[-training_set,]) # store R-
squared
}
auc[,3] <- auc[,2] - rbind(as.matrix(auc[-1,2]),auc[nrow(auc),2])

#AUC
plot(auc[,1],auc[,2],type = "o",xlab="Trees", ylab= "R-squared", main = "AUC")
abline(a=auc[1,2],b=0, col="red")

###############################################################################
# Task 3
###############################################################################

## Deep trees
min_obs <- 10
# Build Forest
set.seed(1001)
forest2 <- regression_forest(x_2006[training_set,],log_y_2006[training_set,],
                            mtry = floor(cov*ncol(x_2006)), sample.fraction = frac, num.trees = rep,
                            min.node.size = min_obs, honesty=FALSE)

# Prediction
fit <- predict(forest2, newdata = x_2006[-training_set,])$predictions

# R-squared
MSE2 <- mean((log_y_2006[-training_set,]-fit)^2)
r2_forest2 <-  1- MSE2/var(log_y_2006[-training_set,])
print(cbind(r2_tree,r2_forest1,r2_forest2))

# Plot tree
tree <- get_tree(forest2, 34)
plot(tree)

###############################################################################
```

```
# Task 4
###############################################################################

# Out-of Sample Prediction
fit_new <- predict(forest2, newdata = x_new)$predictions

###############################################################################
# Task 5
###############################################################################

results <- as.matrix(cbind(id_new,fit_new)) # store ID's and predictions in oine matrix
colnames(results) <- c("id","predictions") # label columns

# Store results
write.csv(results, "predictions.csv")
```