Homework 4

Yiying Wu (yw3996)

R packages

```
library(tidyverse)
library(caret)
library(tidymodels)
library(rpart)
library(rpart.plot)
library(ranger)
library(gbm)
```

1. College data

In this exercise, we will build tree-based models using the College data (see "Col- lege.csv" in Homework 2). The response variable is the out-of-state tuition (Outstate). Partition the dataset into two parts: training data (80%) and test data (20%).

```
dat1<-read_csv("./data/College.csv")
dat1 <- na.omit(dat1)%>% select(-College)
```

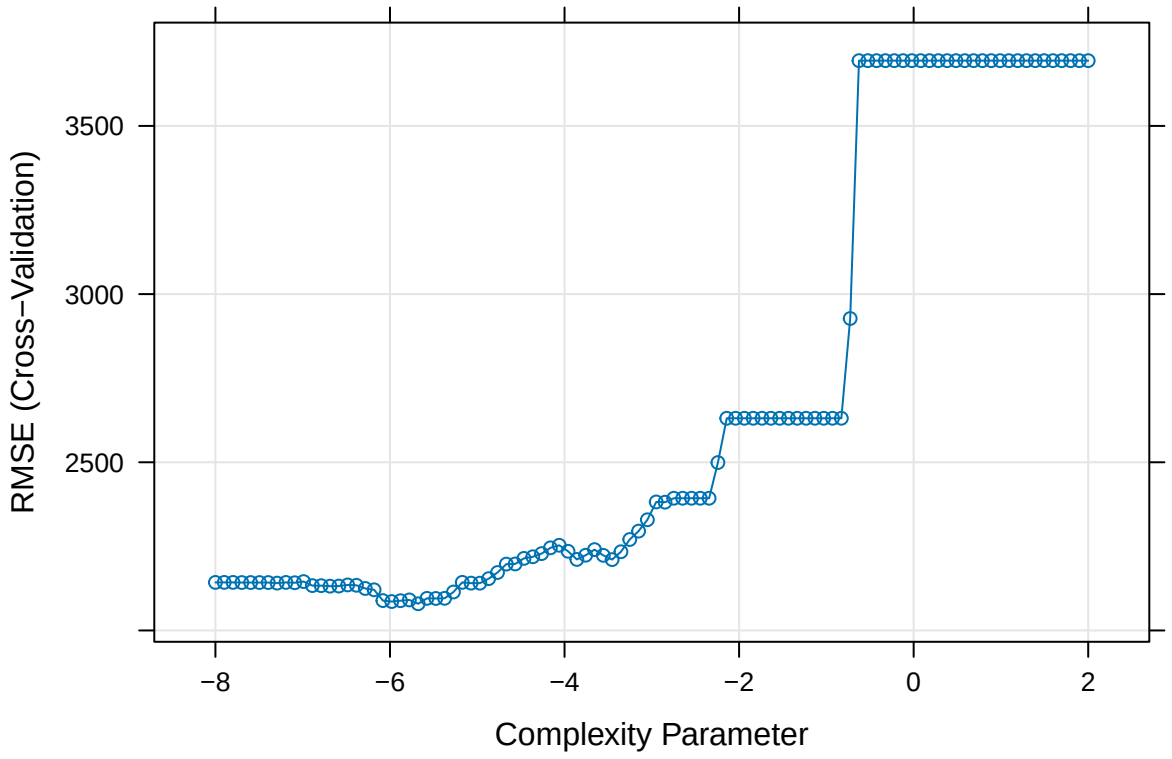
Partition the dataset into two parts: training data (80%) and test data (20%).

```
set.seed(1)
data_split1 <- initial_split(dat1, prop = 0.80)

# Extract the training and test data
training_data1 <- training(data_split1)
x_train1 <- training_data1 %>% select(-Outstate)
y_train1 <- training_data1$Outstate

testing_data1 <- testing(data_split1)
x_test1 <- testing_data1 %>% select(-Outstate)
y_test1 <- testing_data1$Outstate

# ctrl
ctrl1 <- trainControl(method = "cv", number = 10)
```

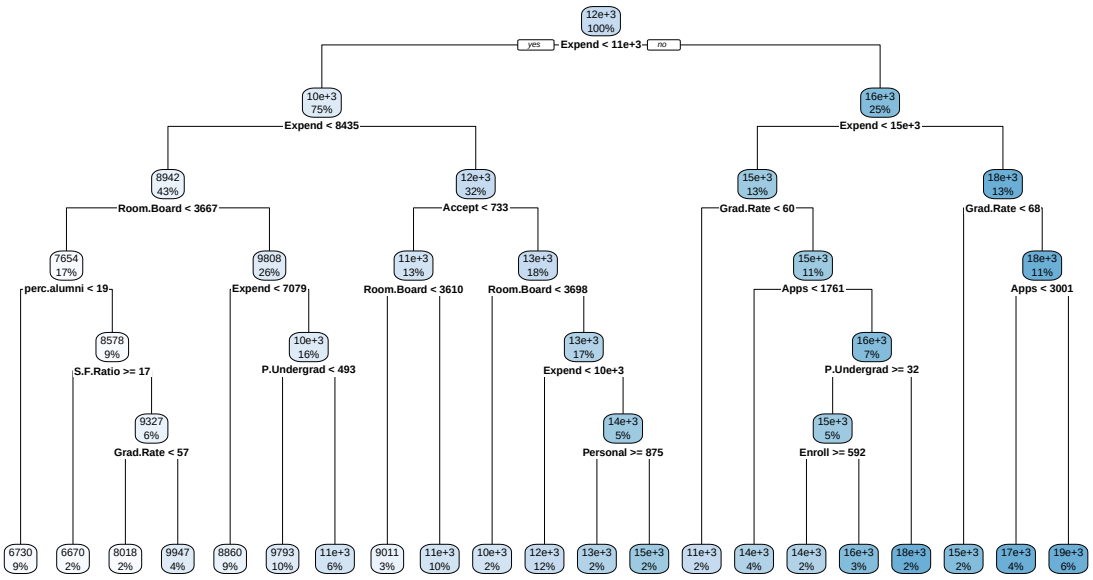
Outcome variable: Outstate

(a) Build a regression tree on the training data to predict the response. Create a plot of the tree.

```
set.seed(1)
rpart.fit <- train(Outstate ~ . ,
                   training_data1,
                   method = "rpart",
                   tuneGrid = data.frame(cp = exp(seq(-8,2, length = 100))),
                   trControl = ctrl1)


plot(rpart.fit, xTrans = log)
```

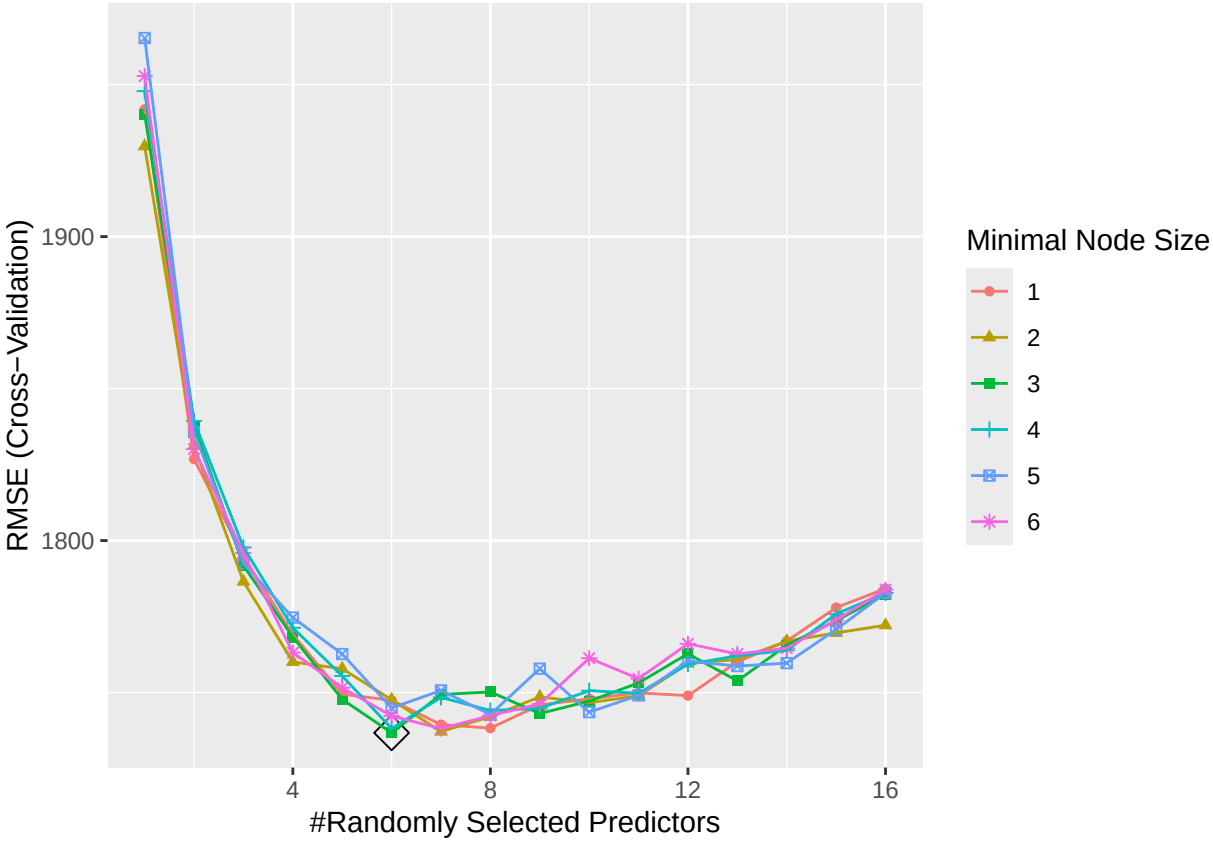



```
rpart.plot(rpart.fit$finalModel)
```



(b) Perform random forest on the training data. Report the variable importance and the test error.

```
rf.grid <- expand.grid(mtry = 1:16,
                       splitrule = "variance",
                       min.node.size = 1:6)
set.seed(1)
rf.fit <- train(Outstate ~ . ,
                training_data1,
                method = "ranger",
                tuneGrid = rf.grid,
                trControl = ctrl1)
ggplot(rf.fit, highlight = TRUE)
```

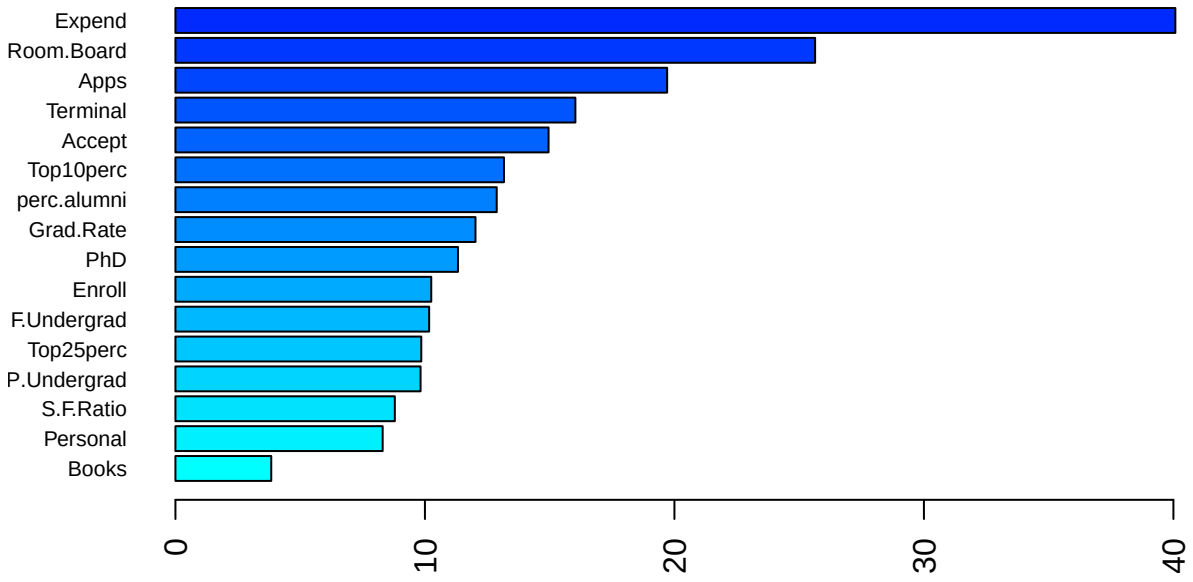



```
rf.fit$bestTune
```

```
##    mtry splitrule min.node.size
## 33    6  variance             3
```

variable importance

```
set.seed(1)
rf.final.per <- ranger(Outstate ~ . ,
                       training_data1,
                       mtry = rf.fit$bestTune[[1]],
                       splitrule = "variance",
                       min.node.size = rf.fit$bestTune[[3]],
                       importance = "permutation",
                       scale.permutation.importance = TRUE)
barplot(sort(ranger::importance(rf.final.per), decreasing = FALSE),
        las = 2, horiz = TRUE, cex.names = 0.7,
        col = colorRampPalette(colors = c("cyan","blue"))(19))
```



test error

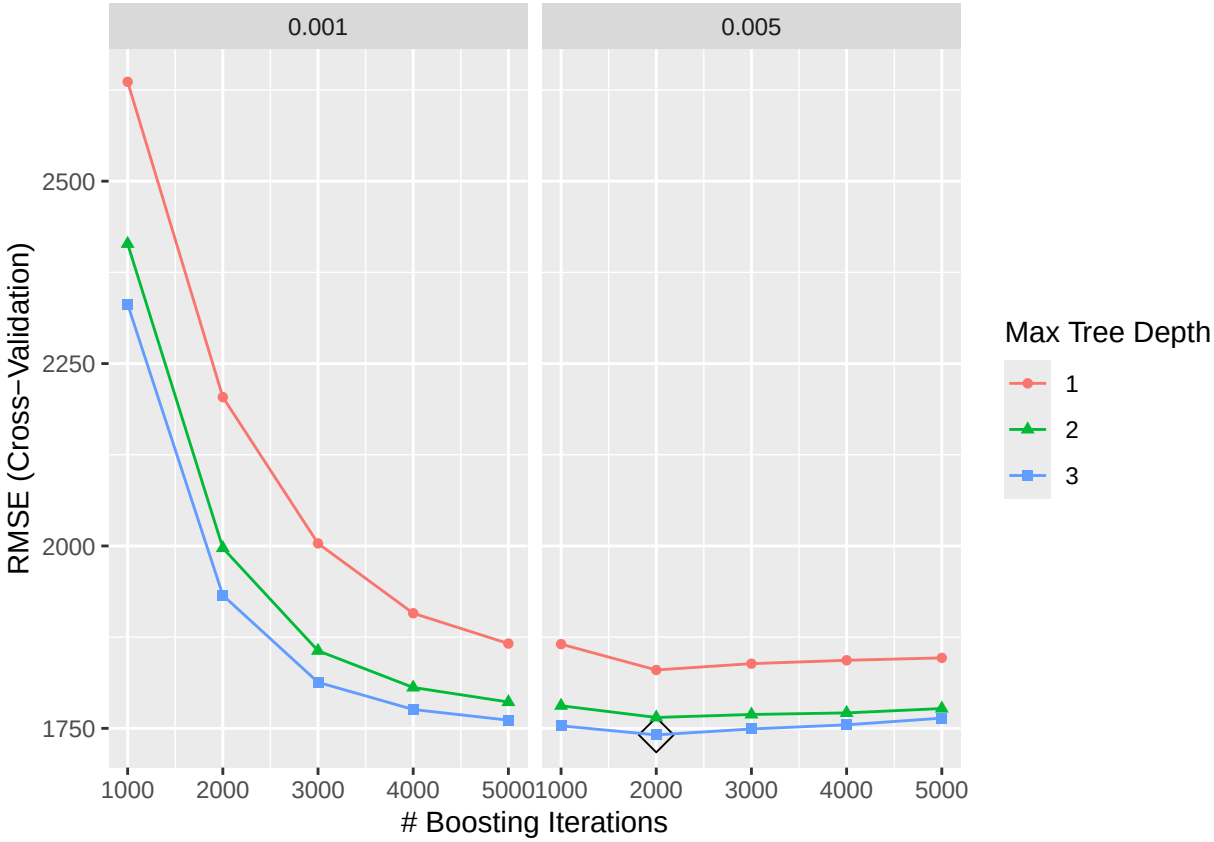
```
rf.predict <- predict(rf.fit, newdata = training_data1)
rf.RMSE <- RMSE(rf.predict, y_test1)
rf.RMSE
```

```
## [1] 5040.468
```

The RMSE for random forest is 5040.468.

(c) Perform boosting on the training data. Report the variable importance and the test error.

```
gbm.grid <- expand.grid(n.trees = c(1000, 2000, 3000, 4000, 5000),
                        interaction.depth = 1:3,
                        shrinkage = c(0.001, 0.005),
                        n.minobsinnode = c(1))

set.seed(1)
gbm.fit <- train(Outstate ~ . ,
                 training_data1,
                 method = "gbm",
                 tuneGrid = gbm.grid,
                 trControl = ctrl1,
                 verbose = FALSE)
ggplot(gbm.fit, highlight = TRUE)
```

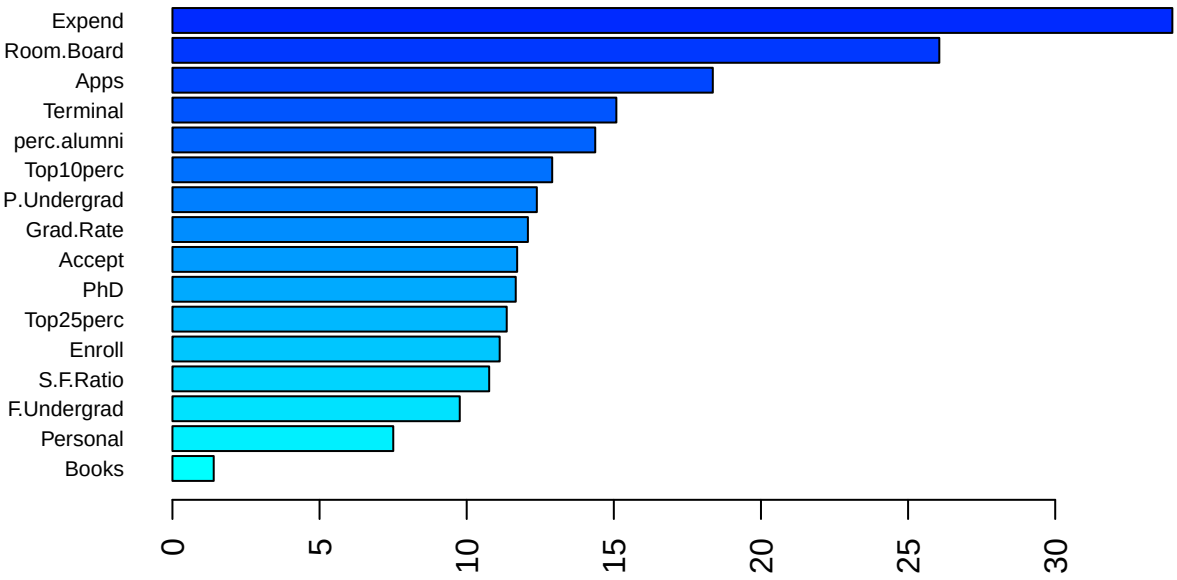



```
gbm.fit$bestTune
```

```
##    n.trees interaction.depth shrinkage n.minobsinnode
## 27    2000                 3     0.005              1
```

variable importance

```
set.seed(1)
gbm.final.per <- ranger(Outstate ~ . ,
                        training_data1,
                        n.trees = gbm.fit$bestTune[[1]],
                        splitrule = "variance",
                        interaction.depth = gbm.fit$bestTune[[2]],
                        shrinkage = gbm.fit$bestTune[[3]],
                        n.minobsinnode = gbm.fit$bestTune[[4]],
                        importance = "permutation",
                        scale.permutation.importance = TRUE)
barplot(sort(ranger::importance(gbm.final.per), decreasing = FALSE),
        las = 2, horiz = TRUE, cex.names = 0.7,
        col = colorRampPalette(colors = c("cyan","blue"))(19))
```



test error

```
gbm.predict <- predict(gbm.fit, newdata = testing_data1)
gbm.RMSE <- RMSE(gbm.predict, y_test1)
gbm.RMSE
```

```
## [1] 1649.232
```

The RMSE for gbm model is 1649.232.

2. auto data

```
dat2<-read_csv("./data/auto.csv")%>%
  mutate(
    mpg_cat = as.factor(mpg_cat),
    origin = as.factor(origin))
dat2 <- na.omit(dat2)
```

Outcome variable: mpg_cat

```
contrasts(dat2$mpg_cat)
```

```
##      low
## high   0
## low    1
```

Split the dataset into two parts: training data (70%) and test data (30%).

```
set.seed(1)
data_split2 <- initial_split(dat2, prop = 0.7)
```

```r
# Extract the training and test data
training_data2 <- training(data_split2)
testing_data2 <- testing(data_split2)

ctrl2 <- trainControl(method = "cv", number = 10,
                      summaryFunction = twoClassSummary,
                      classProbs = TRUE)
```