Homework 4

Zhezheng Jin

Contents

```r
library(tidyverse)
library(caret)
library(mlbench)
library(pROC)
library(pdp)
library(ISLR)
library(caret)
library(rpart)
library(rpart.plot)
library(ranger)
library(tidymodels)
```

1.College Data

```r
# Data Import
data = read_csv("College.csv") %>%
  janitor::clean_names() %>%
  dplyr::select(-college) %>%
  relocate(outstate)
```

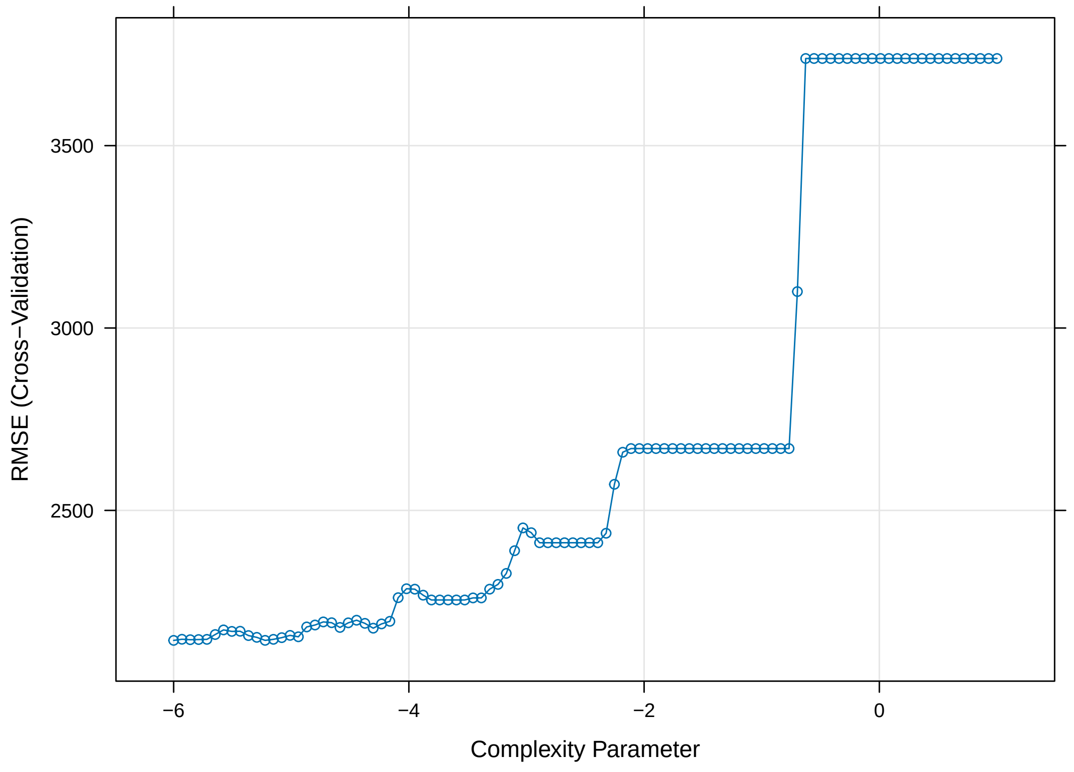
```
## Rows: 565 Columns: 18
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): College
## dbl (17): Apps, Accept, Enroll, Top10perc, Top25perc, F.Undergrad, P.Undergr...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# data partition
set.seed(2358)
data_split <- initial_split(data, prop = 0.8)
train <- training(data_split)
test <- testing(data_split)
```

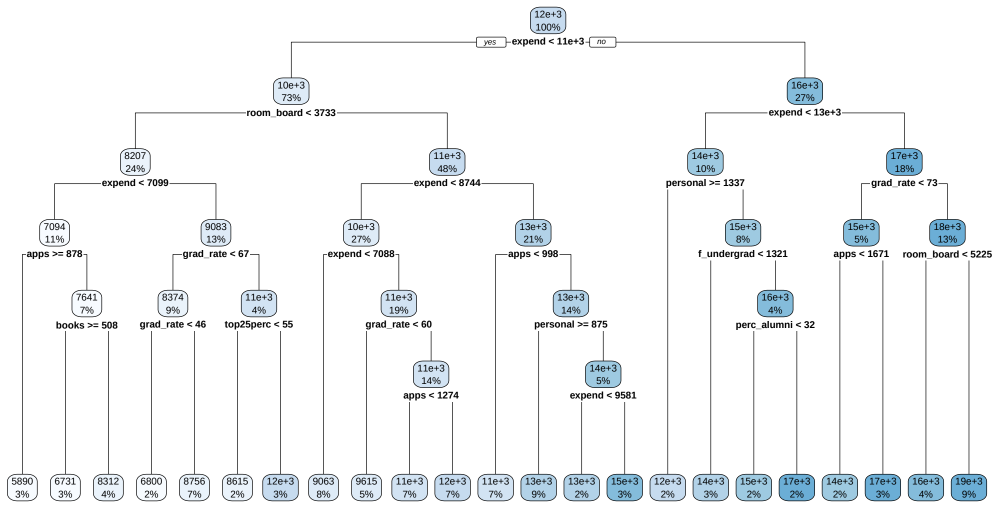
The "College" dataset contains 17 columns and 565 observations after omitting the `college` variable.

a. Regression Tree

```r
# using caret
ctrl <- trainControl(method = "cv")
set.seed(2358)
rpart.fit <- train(outstate ~ .,
                   train,
                   method = "rpart",
                   tuneGrid = data.frame(cp =  exp(seq(-6,1, length = 100))),
                   trControl = ctrl)
plot(rpart.fit, xTrans = log)
```



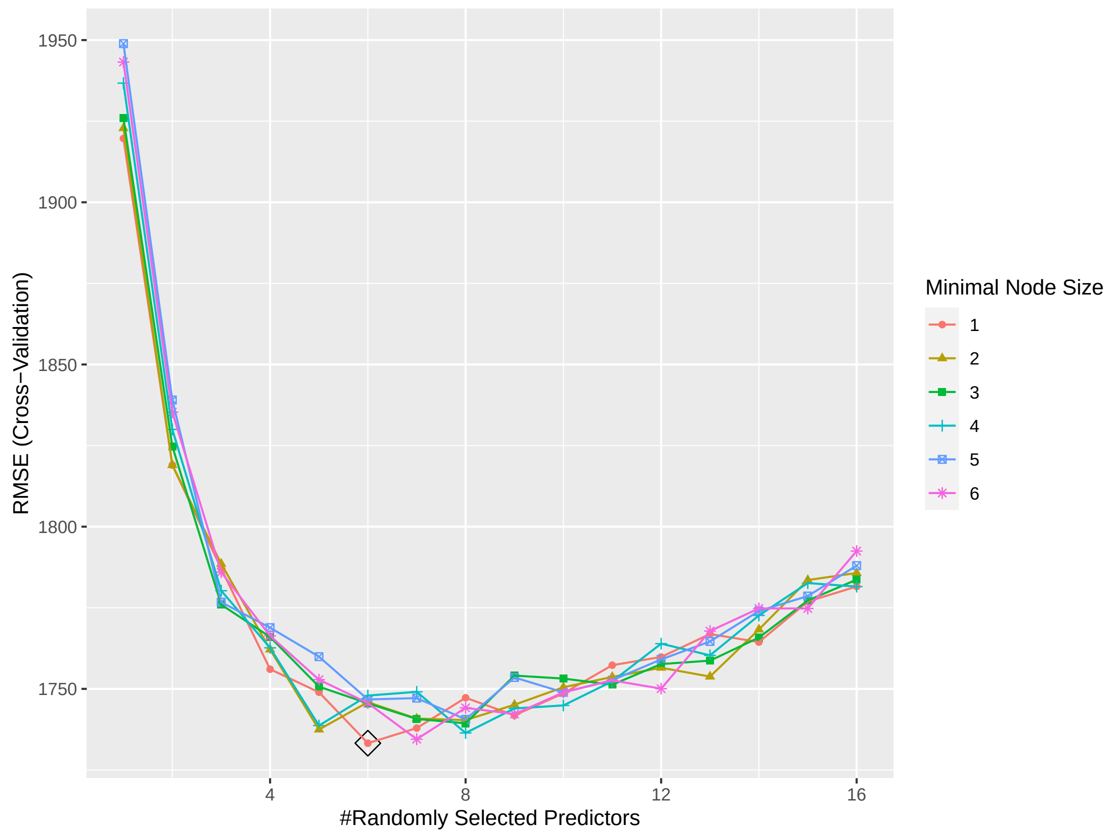
```
# Plot of tree
rpart.plot(rpart.fit$finalModel)
```



b. Random Forest

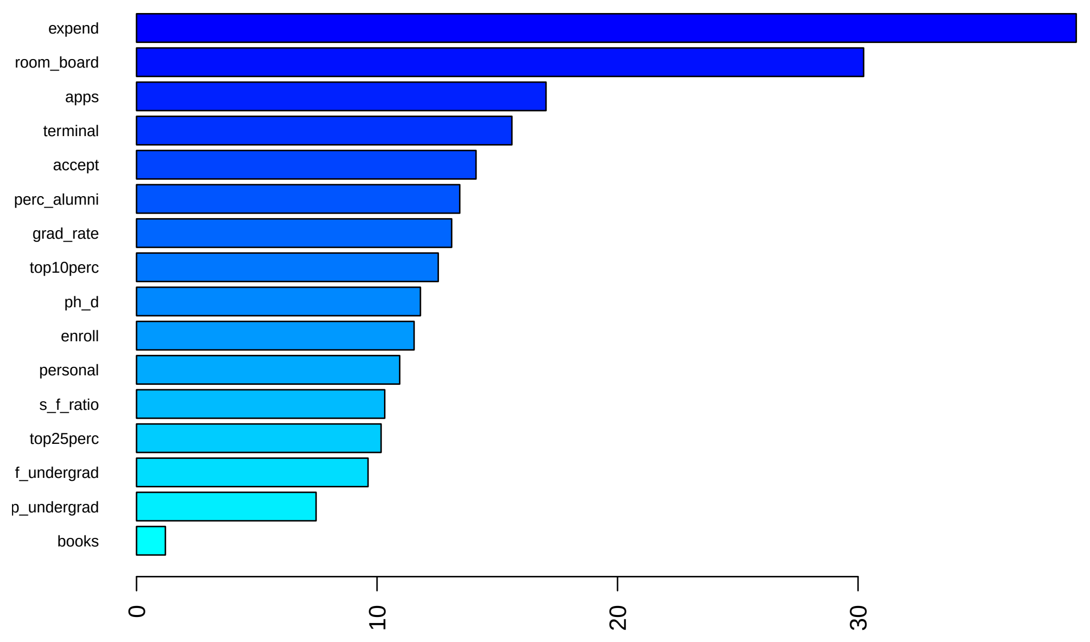
```
# using caret
rf.grid <- expand.grid(mtry = 1:16,
                       splitrule = "variance",
                       min.node.size = 1:6)
set.seed(2358)
rf.fit <- train(outstate ~ .,
                train,
                method = "ranger",
                tuneGrid = rf.grid,
                trControl = ctrl)

ggplot(rf.fit, highlight = TRUE)
```



```r
# Variable importance
set.seed(2358)
rf.final.per <- ranger(outstate ~ .,
                       train,
                       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"))(16))
```



```
# Test error
pred.rf <- predict(rf.fit, newdata = test)
rf.test.error <- RMSE(pred.rf, test$outstate)
rf.test.error
```

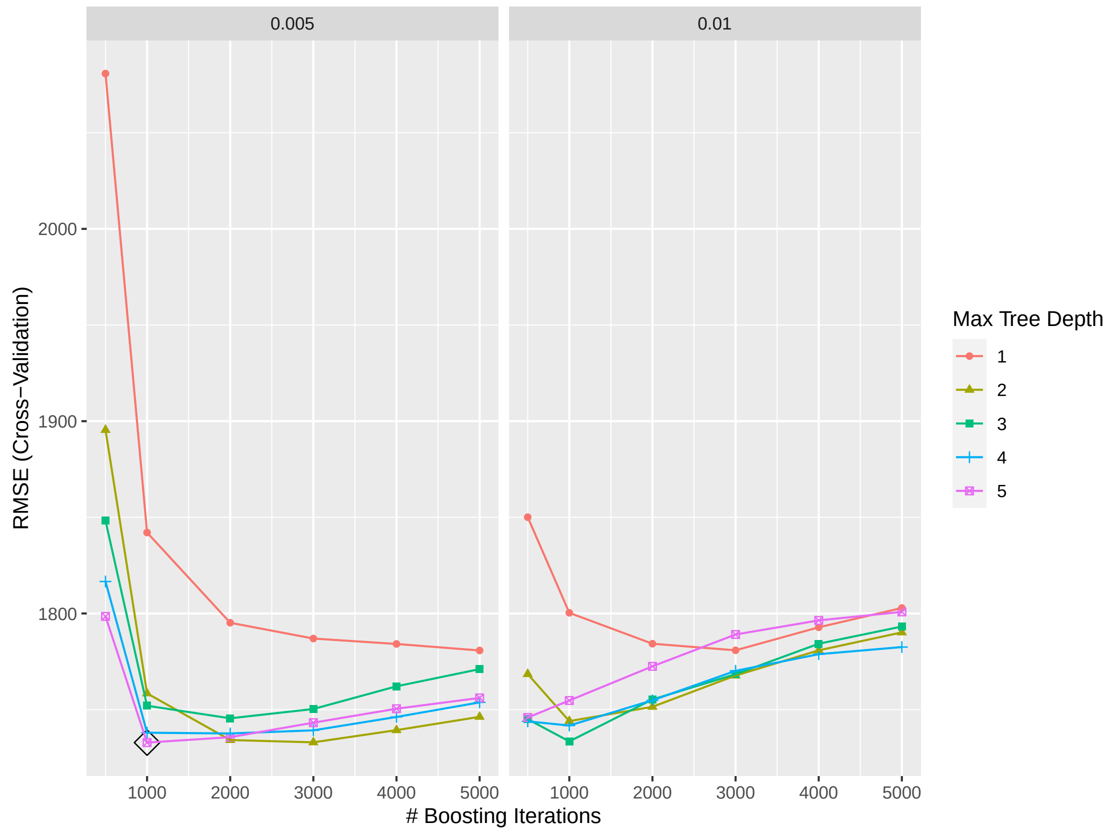
```
## [1] 1764.309
```

expend and room_board are the most important variables, followed by apps, terminal, accept,perc_alumni, grad_rate, etc. The test error(RMSE) is 1764.3089344.

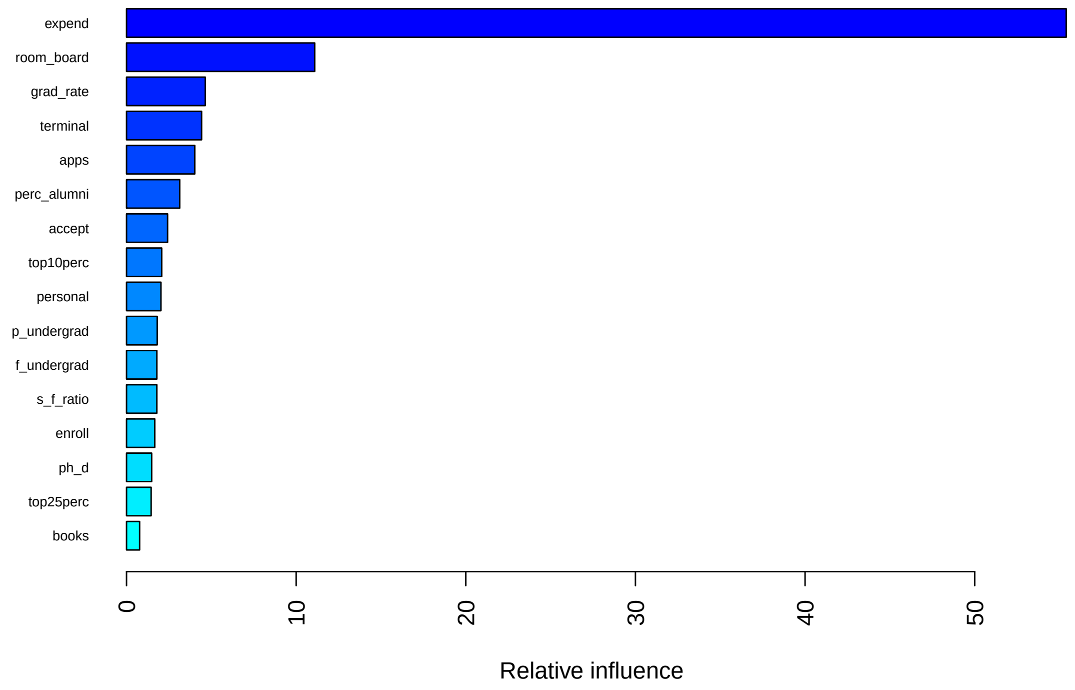
c. Boosting

```
# using caret
gbm.grid <- expand.grid(n.trees = c(500,1000,2000,3000,4000,5000),
                        interaction.depth = 1:5,
                        shrinkage = c(0.005,0.01),
                        n.minobsinnode = 1)
set.seed(2358)
gbm.fit <- train(outstate ~ .,
                 train,
                 method = "gbm",
                 tuneGrid = gbm.grid,
                 trControl = ctrl,
                 verbose = FALSE)
```

```r
ggplot(gbm.fit, highlight = TRUE)
```



```r
# Variable importance
summary(gbm.fit$finalModel, las = 2, cBars = 16, cex.names = 0.6)
```



```
##                     var    rel.inf
## expend           expend 55.3950421
## room_board   room_board 11.0935390
## grad_rate     grad_rate  4.6443246
## terminal       terminal  4.4250604
## apps               apps  4.0239001
## perc_alumni perc_alumni  3.1358692
## accept           accept  2.4211837
## top10perc     top10perc  2.0746045
## personal       personal  2.0296844
## p_undergrad p_undergrad  1.8090414
## f_undergrad f_undergrad  1.7887358
## s_f_ratio     s_f_ratio  1.7874066
## enroll           enroll  1.6653943
## ph_d               ph_d  1.4831469
## top25perc     top25perc  1.4505985
## books             books  0.7724682
```

```
# Test error
pred.gbm <- predict(gbm.fit, newdata = test)
gbm.test.error <- RMSE(pred.gbm, test$outstate)
gbm.test.error
```

```
## [1] 1739.106
```

expend is the most important variables, followed by room_board, grad_rate, terminal, apps, perc_alumni, etc. The test error(RMSE) is 1739.1061213.

2.Auto Data

```
# Data Import
auto <- read_csv("auto.csv") %>%
  mutate(
    mpg_cat = as.factor(mpg_cat),
    origin = factor(origin, levels = 1:3),
    cylinders = as.factor(cylinders))
```

```
## Rows: 392 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): mpg_cat
## dbl (7): cylinders, displacement, horsepower, weight, acceleration, year, or...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
skimr::skim(auto)
```

Table 1: Data summary

Name	auto
Number of rows	392
Number of columns	8
Column type frequency:	
factor	3
numeric	5
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
cylinders	0	1	FALSE	5	4: 199, 8: 103, 6: 83, 3: 4
origin	0	1	FALSE	3	1: 245, 3: 79, 2: 68
mpg_cat	0	1	FALSE	2	hig: 196, low: 196

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
displacement	0	1	194.41	104.64	68	105.00	151.0	275.75	455.0	
horsepower	0	1	104.47	38.49	46	75.00	93.5	126.00	230.0	
weight	0	1	2977.58	849.40	1613	2225.25	2803.5	3614.75	5140.0	
acceleration	0	1	15.54	2.76	8	13.78	15.5	17.02	24.8	
year	0	1	75.98	3.68	70	73.00	76.0	79.00	82.0	

```r
contrasts(auto$mpg_cat)
```

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

```r
# data partition
set.seed(2358)
data_split2 <- initial_split(auto, prop = 0.7)
train2 <- training(data_split2)
test2 <- testing(data_split2)
```

The "auto" dataset contains 8 variables and 392 observations.

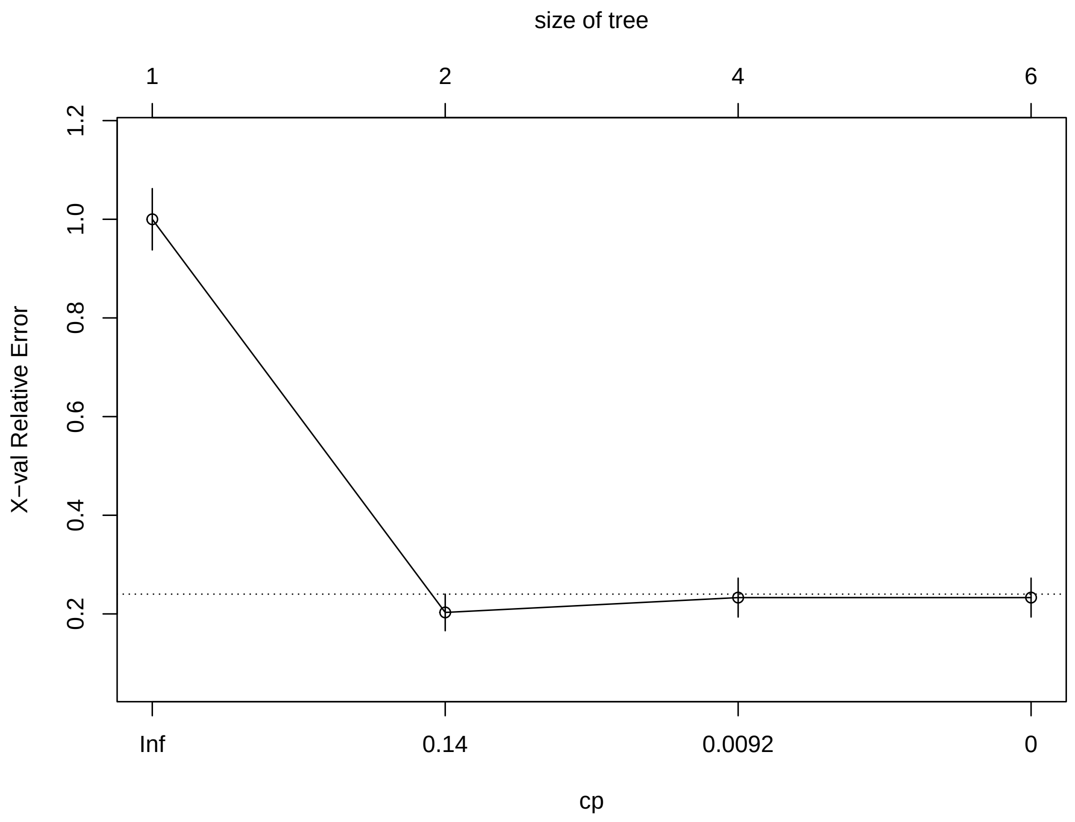
a. Classification Tree

```r
# using rpart
set.seed(2358)
tree1 <- rpart(formula = mpg_cat ~ .,
               data = train2,
               control = rpart.control(cp = 0))

cpTable <- printcp(tree1)
```

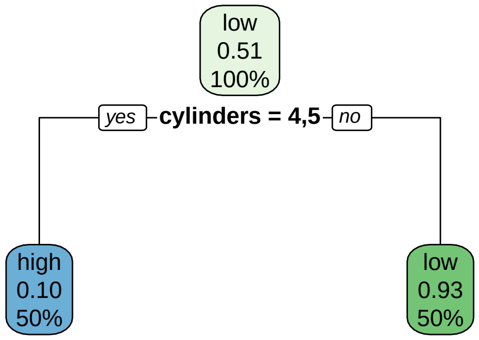
```
##
## Classification tree:
## rpart(formula = mpg_cat ~ ., data = train2, control = rpart.control(cp = 0))
##
## Variables actually used in tree construction:
## [1] acceleration cylinders    horsepower   year
##
## Root node error: 133/274 = 0.4854
##
## n= 274
##
##          CP nsplit rel error  xerror     xstd
## 1 0.8195489      0   1.00000 1.00000 0.062203
## 2 0.0225564      1   0.18045 0.20301 0.037094
## 3 0.0037594      3   0.13534 0.23308 0.039424
## 4 0.0000000      5   0.12782 0.23308 0.039424
```

```r
plotcp(tree1)
```



```
# minimum cross-validation error
minErr <- which.min(cpTable[, "xerror"])
tree2 <- rpart::prune(tree1, cp = cpTable[minErr, "CP"])
rpart.plot(tree2)

# 1SE rule
cp1se <- cpTable[cpTable[, "xerror"]
                 <= cpTable[minErr, "xerror"]
                 + cpTable[minErr, "xstd"], "CP"][1]
tree3 <- rpart::prune(tree1, cp = cp1se)
rpart.plot(tree3)
```



The optimal tree with two terminal nodes (size 2), chosen based on the lowest cross-validation error, contains only one split, which is determined by the cylinder predictor in category 4 and 5.

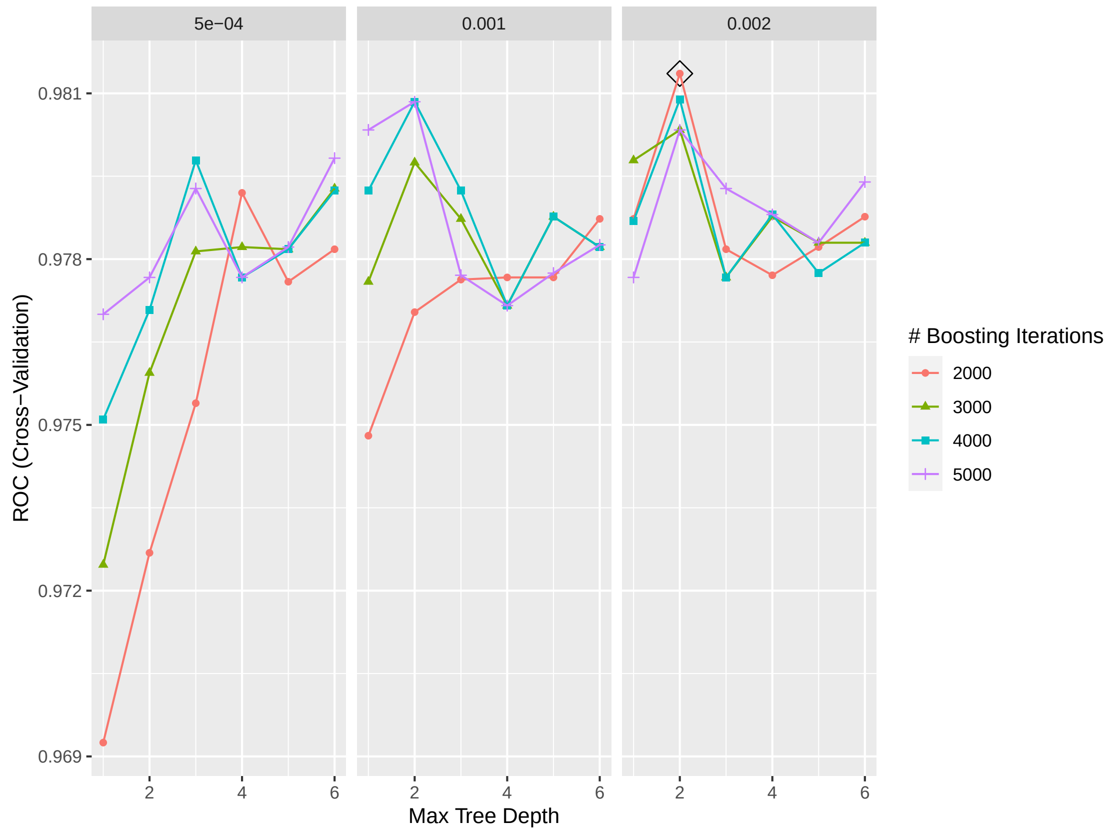
The optimal tree, chosen based on 1SE cross-validation error, is the same as the tree size obtained using the lowest cross-validation error, which is size 2.

b. Boosting

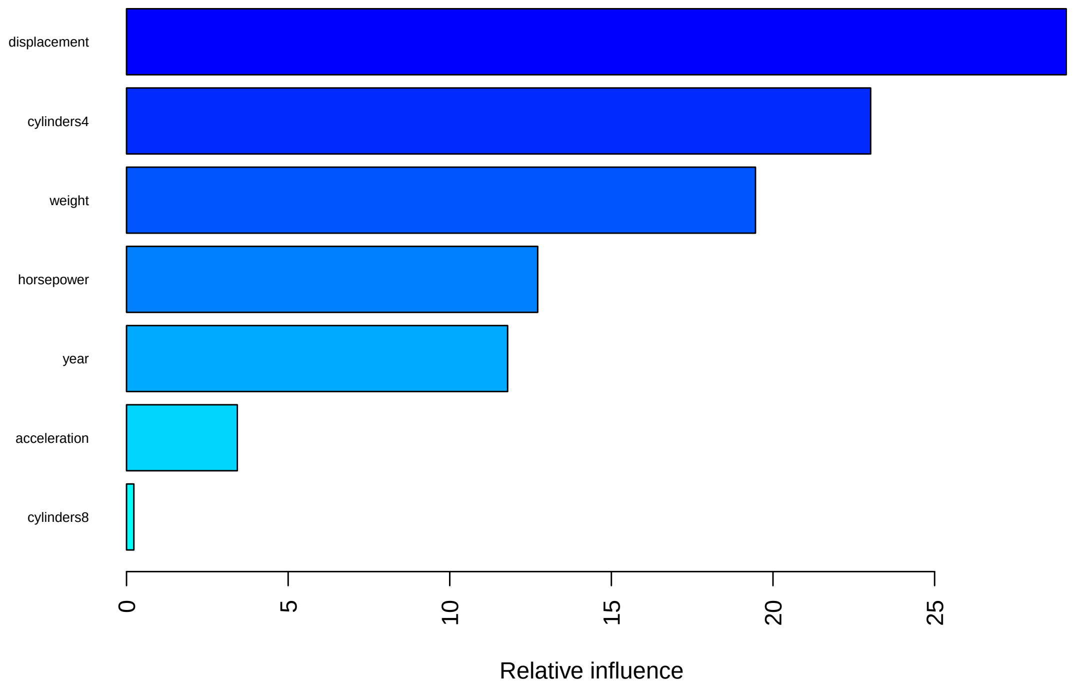
```
# using caret
ctrl2 <- trainControl(method = "cv",
                      classProbs = TRUE,
                      summaryFunction =twoClassSummary)

gbm.grid2 <- expand.grid(n.trees = c(2000,3000,4000,5000),
                         interaction.depth = 1:6,
                         shrinkage = c(0.0005,0.001,0.002),
                         n.minobsinnode = 1)
set.seed(2358)
gbm.fit2 <- train(mpg_cat ~ .,
                  train2,
                  method = "gbm",
                  tuneGrid = gbm.grid2,
                  trControl = ctrl2,
                  distribution = "adaboost",
                  metric = "ROC",
                  verbose = FALSE)
```

```r
ggplot(gbm.fit2, highlight = TRUE)
```



```r
# Variable importance
summary(gbm.fit2$finalModel, las = 2, cBars = 7, cex.names = 0.6)
```



```
##                       var     rel.inf
## displacement displacement 29.07182010
## cylinders4     cylinders4 23.02170771
## weight             weight 19.45775655
## horsepower     horsepower 12.72078613
## year                 year 11.79051661
## acceleration acceleration  3.42740469
## cylinders8     cylinders8  0.22721536
## cylinders6     cylinders6  0.18141569
## origin3           origin3  0.05278769
## origin2           origin2  0.04858946
## cylinders5     cylinders5  0.00000000
```

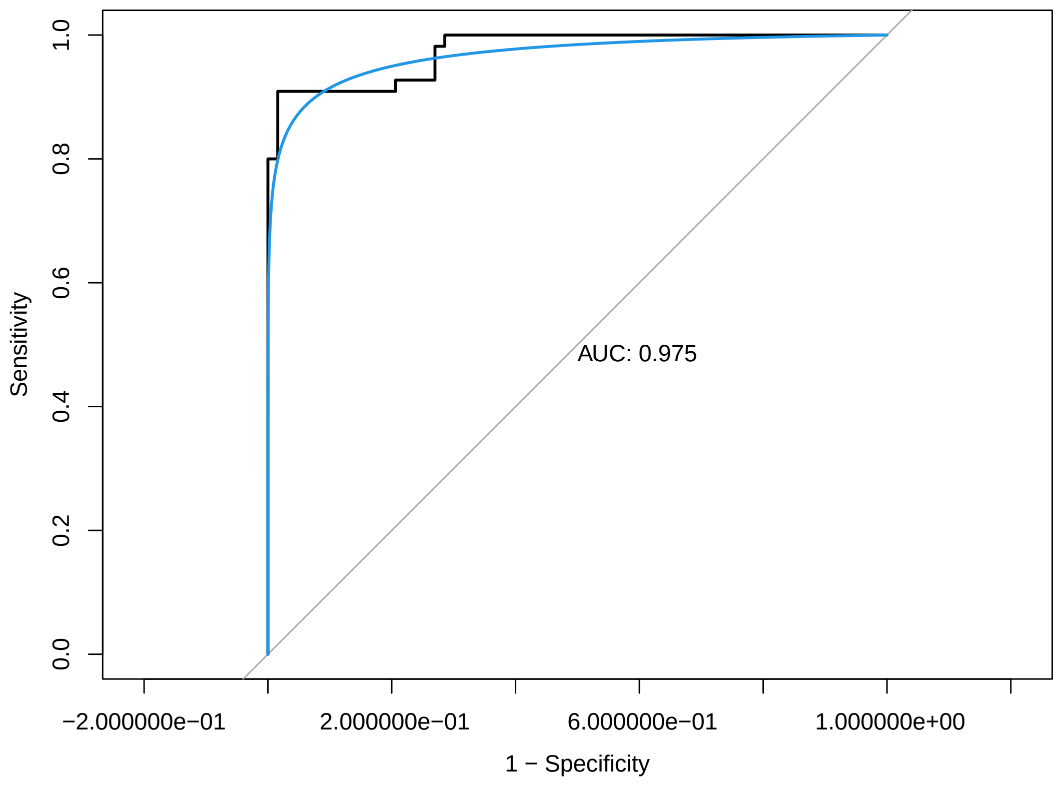
```
# test data performance
gbm2.pred.prob <- predict(
  gbm.fit2, newdata = test2, type = "prob")[,2]
#  retrieve the second column, which is case="low",positive class for the mpg_cat

## ROC plot
roc.gbm2 <- roc(test2$mpg_cat, gbm2.pred.prob)
```

```
## Setting levels: control = high, case = low
```

```
## Setting direction: controls < cases
```

```
plot(roc.gbm2, legacy.axes = TRUE,  print.auc = TRUE)
plot(smooth(roc.gbm2), col = 4, add = TRUE)
```



```
## Confusion Matrix
gbm2.pred <- rep("high", length(gbm2.pred.prob))
gbm2.pred[gbm2.pred.prob>0.5] <- "low"

confusionMatrix(data = as.factor(gbm2.pred),
                reference = test2$mpg_cat,
                positive = "low")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction high low
##       high   62   6
##       low     1  49
##
##                Accuracy : 0.9407
##                  95% CI : (0.8816, 0.9758)
##     No Information Rate : 0.5339
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8801
##
```

```
##  Mcnemar's Test P-Value : 0.1306
##
##             Sensitivity : 0.8909
##             Specificity : 0.9841
##          Pos Pred Value : 0.9800
##          Neg Pred Value : 0.9118
##              Prevalence : 0.4661
##          Detection Rate : 0.4153
##    Detection Prevalence : 0.4237
##       Balanced Accuracy : 0.9375
##
##        'Positive' Class : low
##
```

displacement is the most important variables, followed by cylinders4, weight, horsepower, year, etc.

Test data performance:

The AUC value is 0.975, which means the model's performance on the test data can be considered to be very high.

Based on our confusion matrix analysis, our model's accuracy when applied to test data is 94.07% (95% CI: 88.16% to 97.58%). No information rate is 53.39%, which represents the accuracy if we made the same class prediction for all observations without any information. The p-value is close to 0 which means the accuracy is statistically significantly better than our no information rate.