ds2_hw4

Minghe Wang

2025-04-21

```
colleges <- read_csv("./College.csv")
```

```
## 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.
```

```
head(colleges)
```

```
## # A tibble: 6 x 18
##   College         Apps Accept Enroll Top10perc Top25perc F.Undergrad P.Undergrad
##   <chr>          <dbl>  <dbl>  <dbl>     <dbl>     <dbl>       <dbl>       <dbl>
## 1 Abilene Chris~  1660   1232    721        23        52        2885         537
## 2 Adelphi Unive~  2186   1924    512        16        29        2683        1227
## 3 Adrian College  1428   1097    336        22        50        1036          99
## 4 Agnes Scott C~   417    349    137        60        89         510          63
## 5 Alaska Pacifi~   193    146     55        16        44         249         869
## 6 Albertson Col~   587    479    158        38        62         678          41
## # i 10 more variables: Outstate <dbl>, Room.Board <dbl>, Books <dbl>,
## #   Personal <dbl>, PhD <dbl>, Terminal <dbl>, S.F.Ratio <dbl>,
## #   perc.alumni <dbl>, Expend <dbl>, Grad.Rate <dbl>
```

1

```
data_split <- initial_split(colleges, prop = 0.8)
training_data <- training(data_split)
testing_data <- testing(data_split)
training_data <- training_data[, -1]
test_dataing <- testing_data[, -1]
```

a

```
ctrl <- trainControl(method = "cv")

set.seed(1)
rpart.fit <- train(Outstate ~ .,
                   training_data,
                   method = "rpart",
```

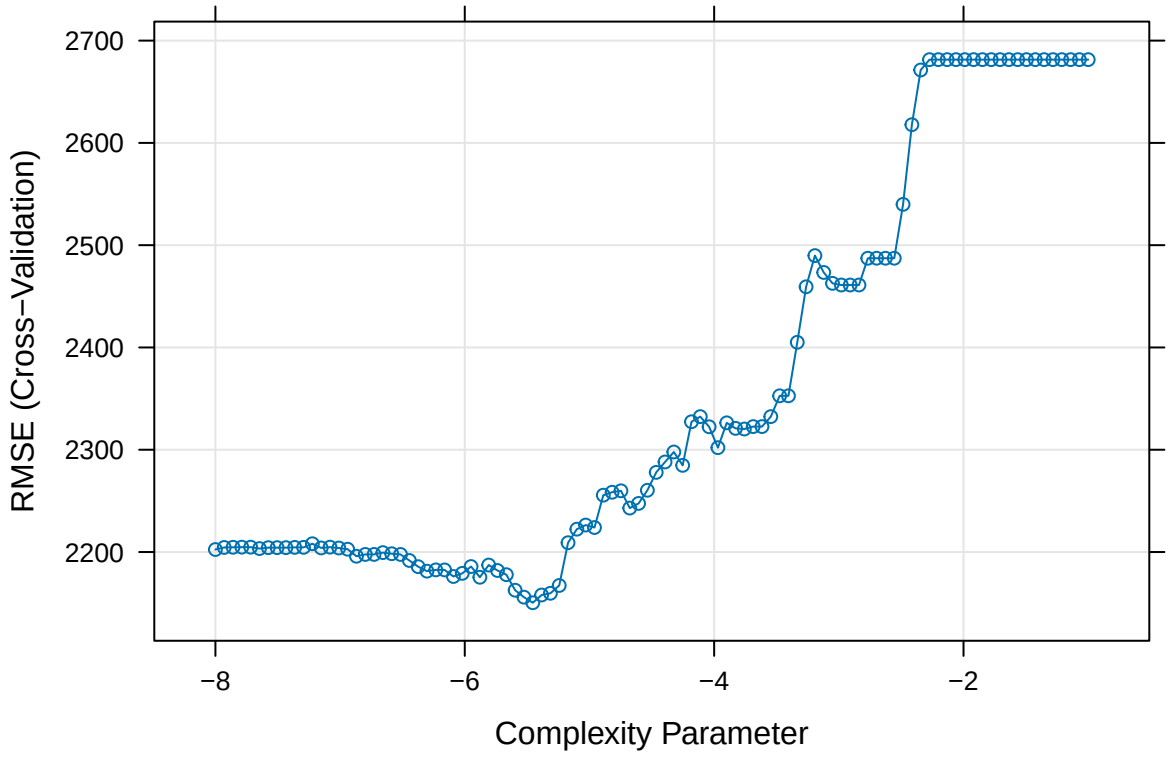
```
                  tuneGrid = data.frame(cp = exp(seq(-8,-1, length = 100))),
                  trControl = ctrl)

pred_rpart <- predict(rpart.fit, newdata = testing_data)
summary(pred_rpart)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6802    9357   11197   11583   12526   18486
```

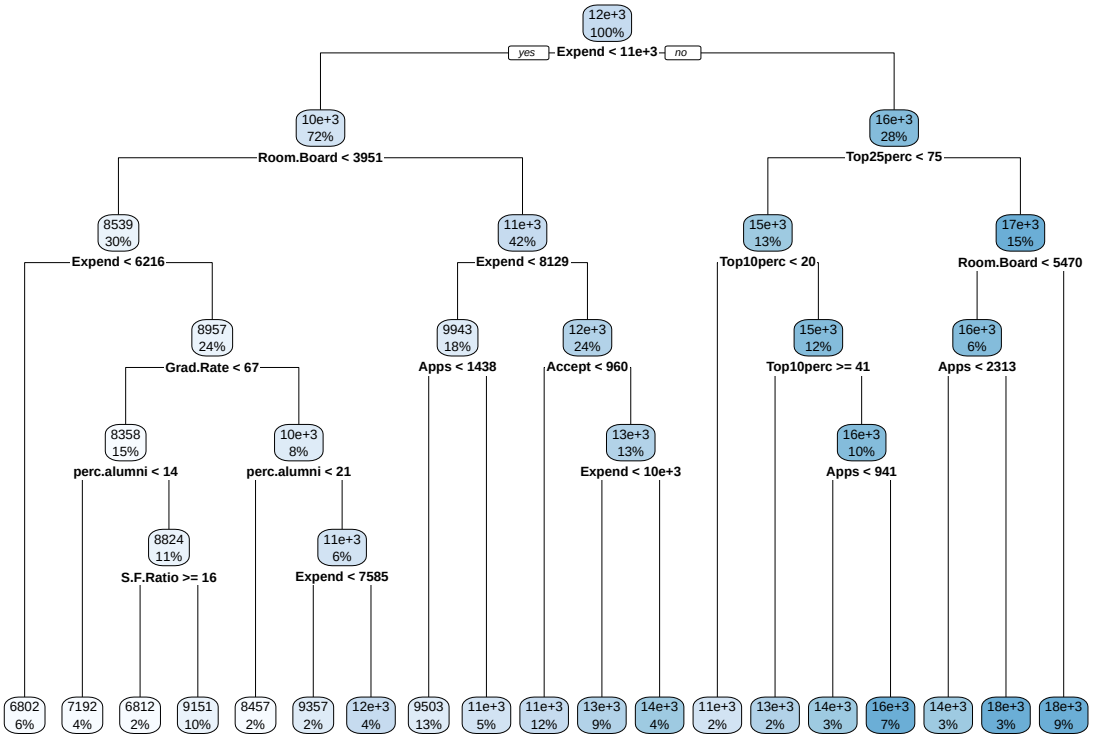
```
head(pred_rpart)
```

```
##         1         2         3         4         5         6
##  7192.200 14394.353  9151.023 11100.682 14394.353  9151.023
```

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




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

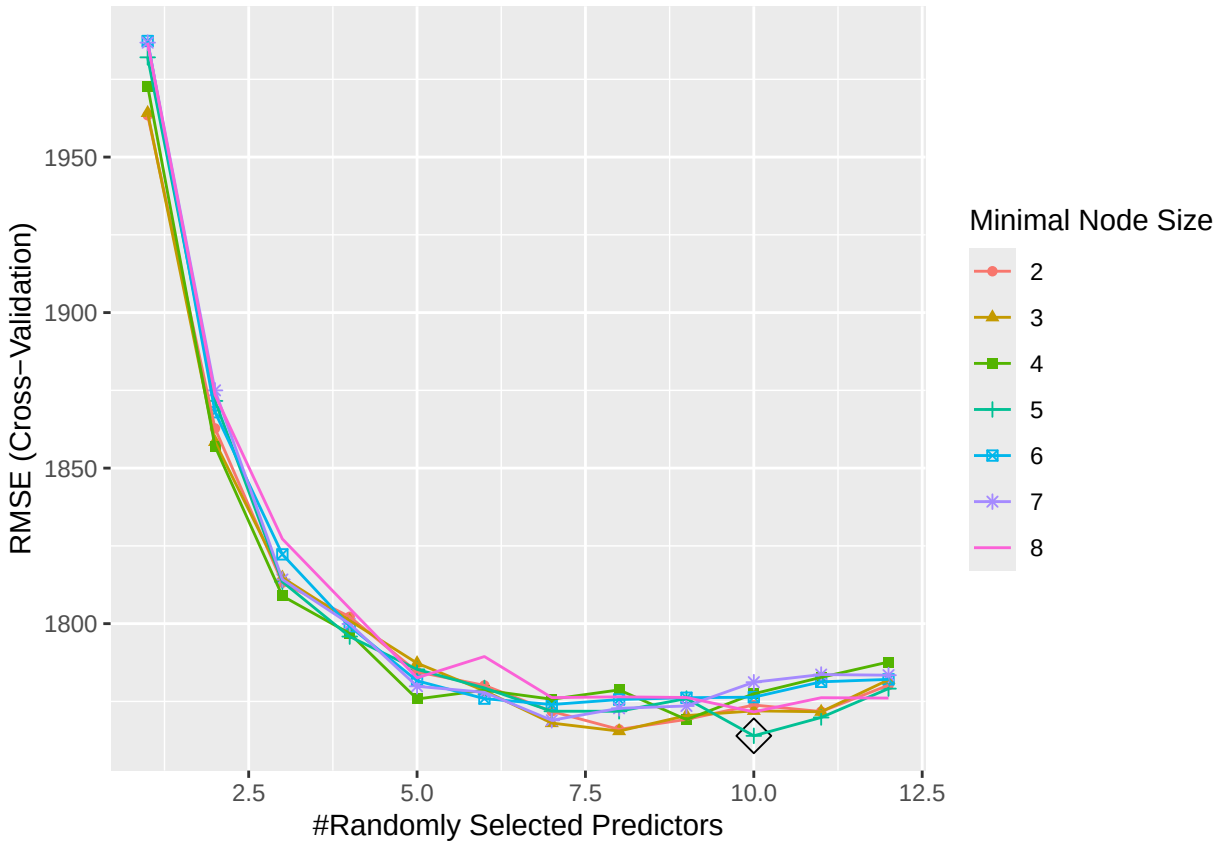


b

```
ctrl <- trainControl(method = "cv")
rf.grid <- expand.grid(mtry = 1:12,
                       splitrule = "variance",
                       min.node.size = 2:8)

set.seed(1)
rf.fit <- train(Outstate ~ . ,
                data = training_data,
                method = "ranger",
                tuneGrid = rf.grid,
                trControl = ctrl)

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

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because more
## than 6 becomes difficult to discriminate
## i you have requested 7 values. Consider specifying shapes manually if you need
##   that many have them.
```

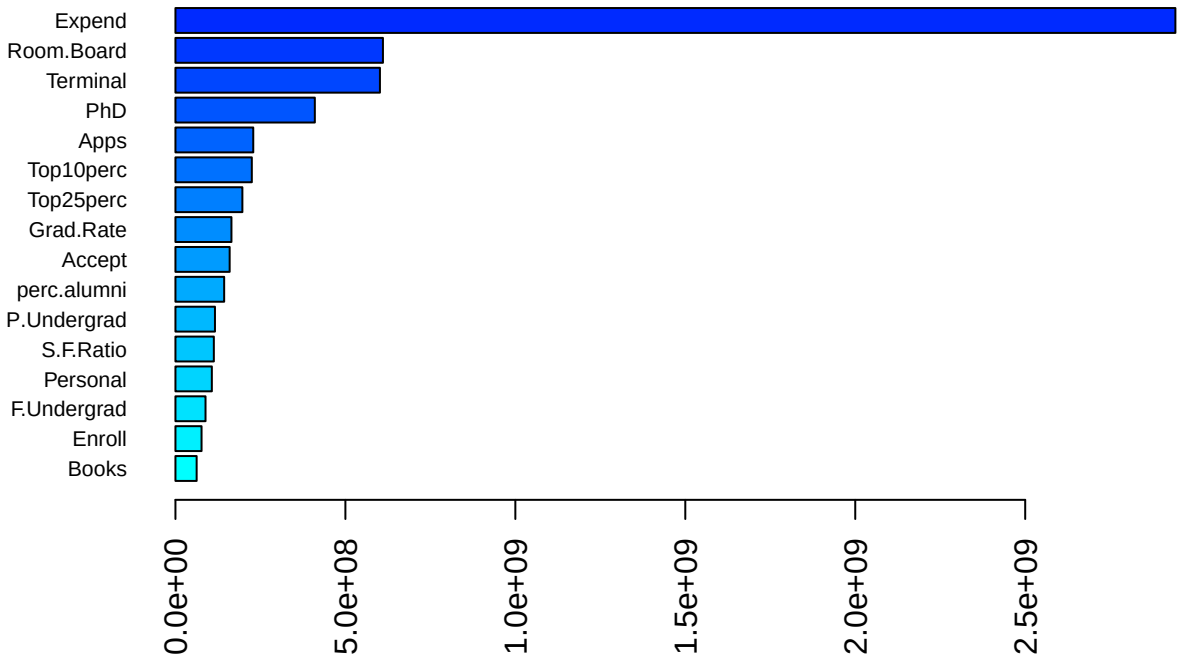
```
## Warning: Removed 12 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
set.seed(1)
rf.final.imp <- ranger(Outstate ~ . ,
                       data = training_data,
                       mtry = rf.fit$bestTune[[1]],
                       splitrule = "variance",
                       min.node.size = rf.fit$bestTune[[3]],
                       importance = "impurity")

ranger::importance(rf.final.imp)
```

```
##        Apps      Accept      Enroll   Top10perc   Top25perc F.Undergrad
##   228914013   159508690    76777390   224734285   196932946    88495717
## P.Undergrad  Room.Board       Books    Personal         PhD    Terminal
##   116308987   610448048    62338338   107076026   410113719   601635903
##   S.F.Ratio perc.alumni      Expend   Grad.Rate
##   112953688   143353956  2941880823   164745165
```

```
barplot(sort(ranger::importance(rf.final.imp), decreasing = FALSE),
        las = 2, horiz = TRUE, cex.names = 0.7,
        col = colorRampPalette(colors = c("cyan","blue"))(19))
```




```
pred_rf <- predict(rf.fit, newdata = testing_data)
rmse_rf <- RMSE(pred_rf, testing_data$Outstate)
rmse_rf
```

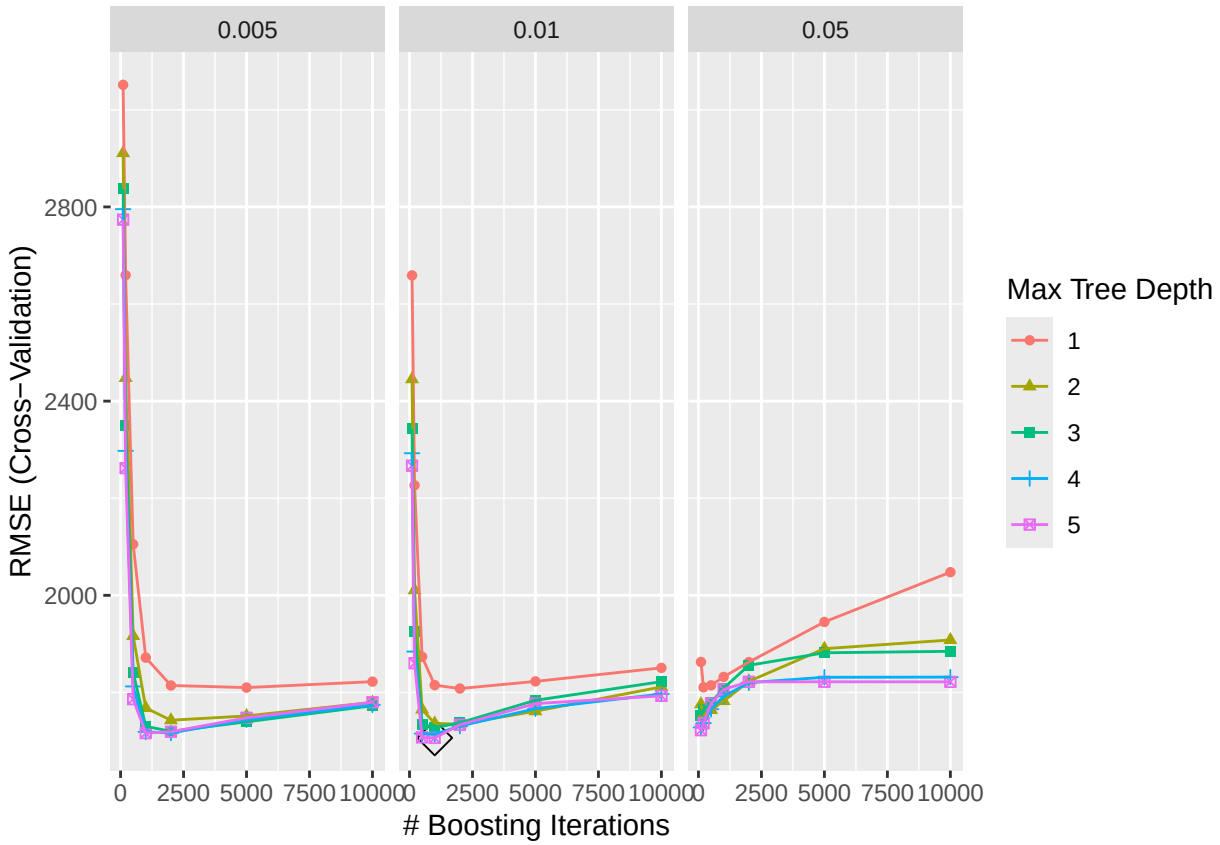
```
## [1] 1742.337
```

Expend is the most importance variable, the Terminal is the second importance, and the rest are odered in a descending way.

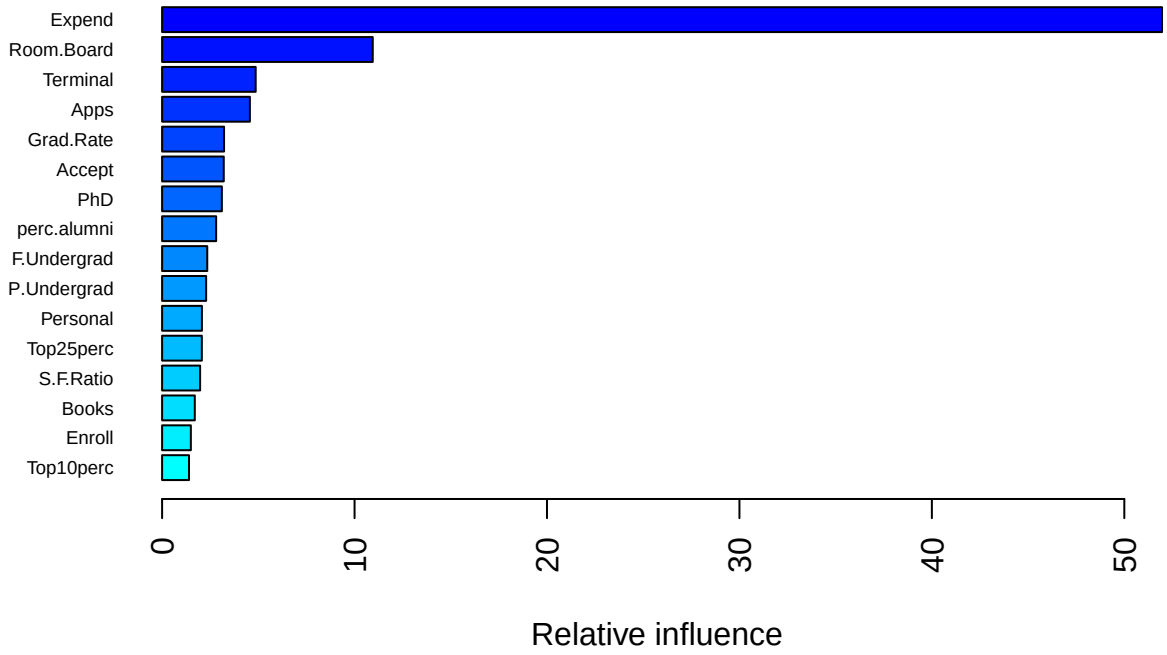
The test error is 1742.3369578.

c

```
gbm.grid <- expand.grid(n.trees = c(100,200,500,1000,2000,5000,10000),
                        interaction.depth = 1:5,
                        shrinkage = c(0.005,0.01,0.05),
                        n.minobsinnode = c(15))

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




```
summary(gbm.fit$finalModel, las = 2, cBars = 19, cex.names = 0.6)
```




```
##                     var   rel.inf
## Expend           Expend 51.959741
## Room.Board   Room.Board 10.946060
## Terminal       Terminal  4.861935
```

```
## Apps               Apps  4.561464
## Grad.Rate     Grad.Rate  3.219611
## Accept           Accept  3.204314
## PhD                 PhD  3.105414
## perc.alumni perc.alumni  2.807528
## F.Undergrad F.Undergrad  2.346381
## P.Undergrad P.Undergrad  2.288423
## Personal       Personal  2.068803
## Top25perc     Top25perc  2.064473
## S.F.Ratio     S.F.Ratio  1.974321
## Books             Books  1.698802
## Enroll           Enroll  1.494286
## Top10perc     Top10perc  1.398445
```

```
pred_gbm <- predict(gbm.fit, newdata = testing_data)
rmse_gbm <- RMSE(pred_rf, testing_data$Outstate)
rmse_gbm
```

```
## [1] 1742.337
```

The expend is the most important variable, and the Room.Board is the second important, the rest are ordered in a descending way.

The test error is 1742.3369578. # 2

```
auto <- read_csv("./auto.csv")
```

```
## 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.
```

```
auto <- na.omit(auto)
auto$origin <- as.factor(auto$origin)
auto$mpg_cat <- factor(auto$mpg_cat, c("high", "low"))

set.seed(1)

auto_split <- initial_split(auto, prop = 0.7)
training_dat <- training(auto_split)
testing_dat <- testing(auto_split)
```

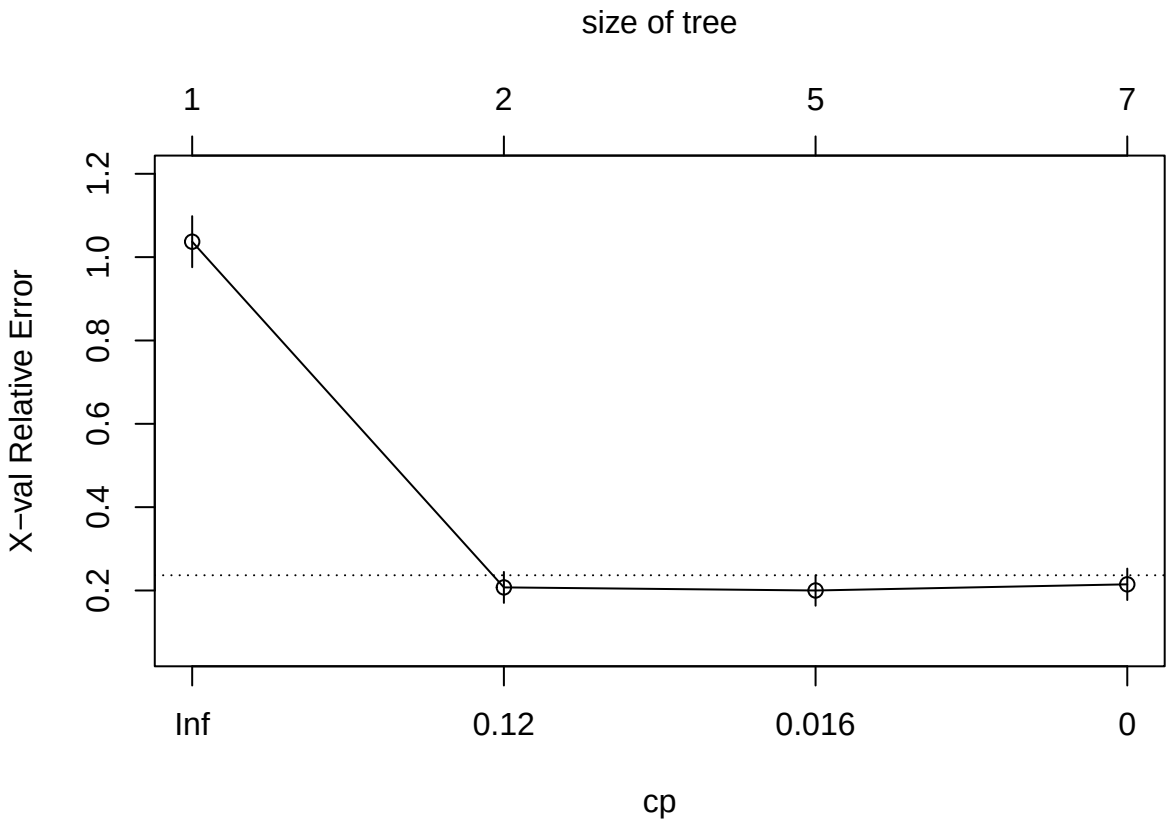
a

```
set.seed(1)
c_tree <- rpart(formula = mpg_cat ~ . ,
                    data = training_dat,
                    control = rpart.control(cp = 0))
cpTable <- printcp(c_tree)
```

```
##
## Classification tree:
## rpart(formula = mpg_cat ~ ., data = training_dat, control = rpart.control(cp = 0))
```

```
##
## Variables actually used in tree construction:
## [1] displacement horsepower   weight       year
##
## Root node error: 135/274 = 0.4927
##
## n= 274
##
##         CP nsplit rel error  xerror     xstd
## 1 0.822222      0  1.000000 1.03704 0.061293
## 2 0.017284      1  0.177778 0.20741 0.037140
## 3 0.014815      4  0.125926 0.20000 0.036544
## 4 0.000000      6  0.096296 0.21481 0.037720
```

```
plotcp(c_tree)
```




```
min_error_ind <- which.min(c_tree$cptable[, "xerror"])
best_size <- c_tree$cptable[min_error_ind, "nsplit"] + 1
best_size
```

```
## [1] 5
```

```
one_se_ind <- which(c_tree$cptable[, "xerror"] <= c_tree$cptable[min_error_ind, "xerror"] + c_tree$cptal

one_se_size <- c_tree$cptable[one_se_ind, "nsplit"] + 1
one_se_size
```

```
## [1] 2
```

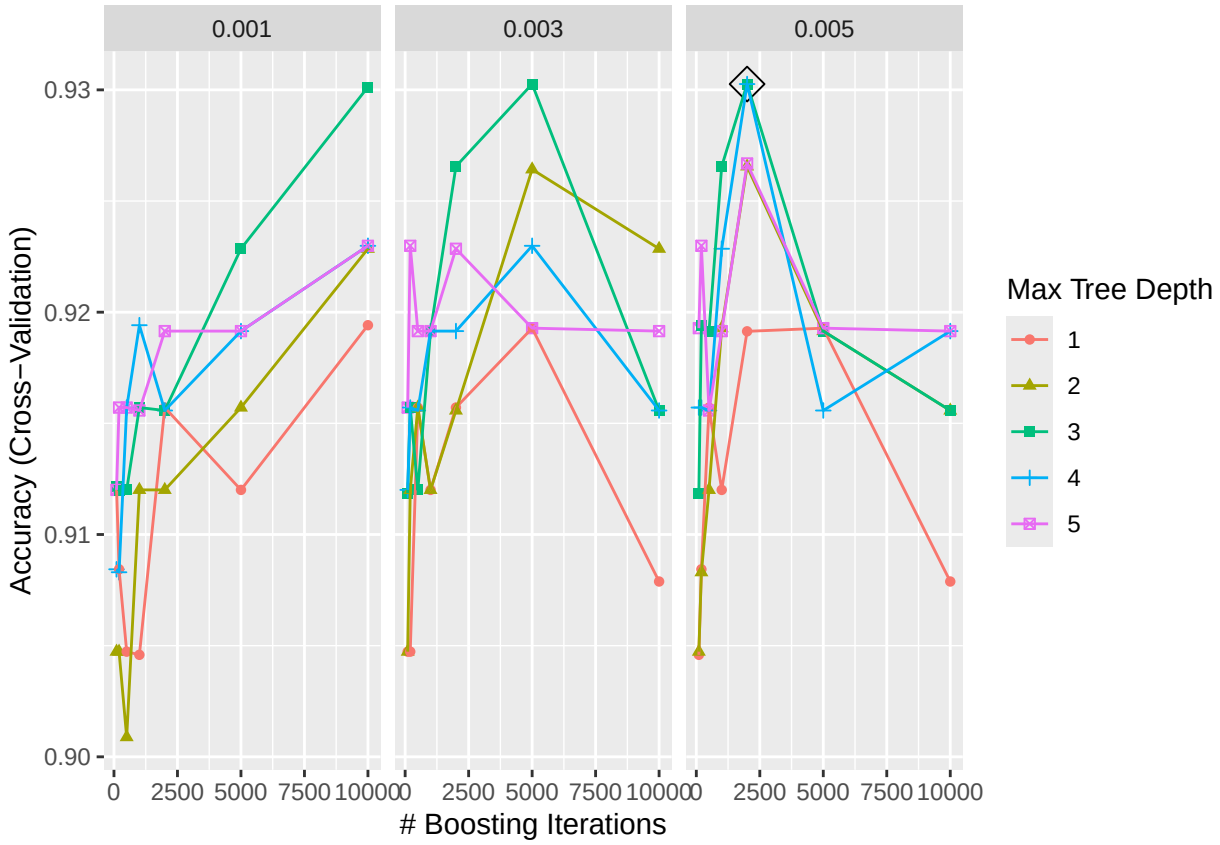
The tree size corresponding to the lowest CV error is 5, which is different from the 1SE CV error's tree size 2.

b

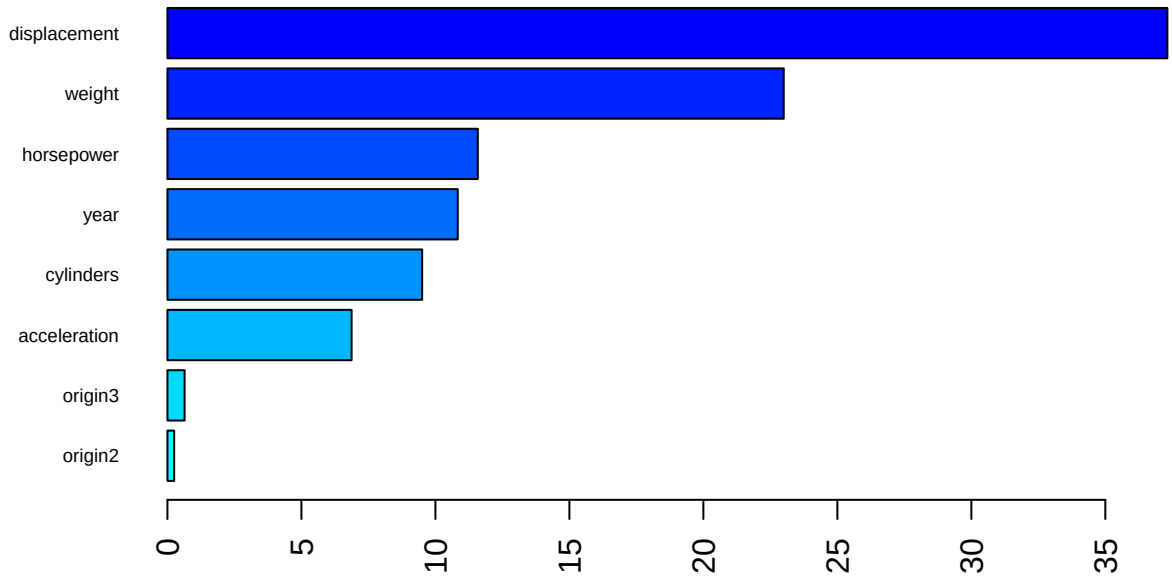
```
bst.grid <- expand.grid(n.trees = c(100,200,500,1000,2000,5000,10000),
                        interaction.depth = 1:5,
                        shrinkage = c(0.001, 0.003, 0.005),
                        n.minobsinnode = 5)
set.seed(1)
ctrl <- trainControl(method = "cv", classProbs = TRUE)
bst.fit <- train(mpg_cat ~ . ,
                 training_dat,
                 tuneGrid = bst.grid,
                 trControl = ctrl,
                 method = "gbm",
                 distribution = "adaboost",
                 metric = "ROC",
                 verbose = FALSE)
```

```
## Warning in train.default(x, y, weights = w, ...): The metric "ROC" was not in
## the result set. Accuracy will be used instead.
```

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




```
summary(bst.fit$finalModel,las=2,cBars=19,cex.names=0.6)
```



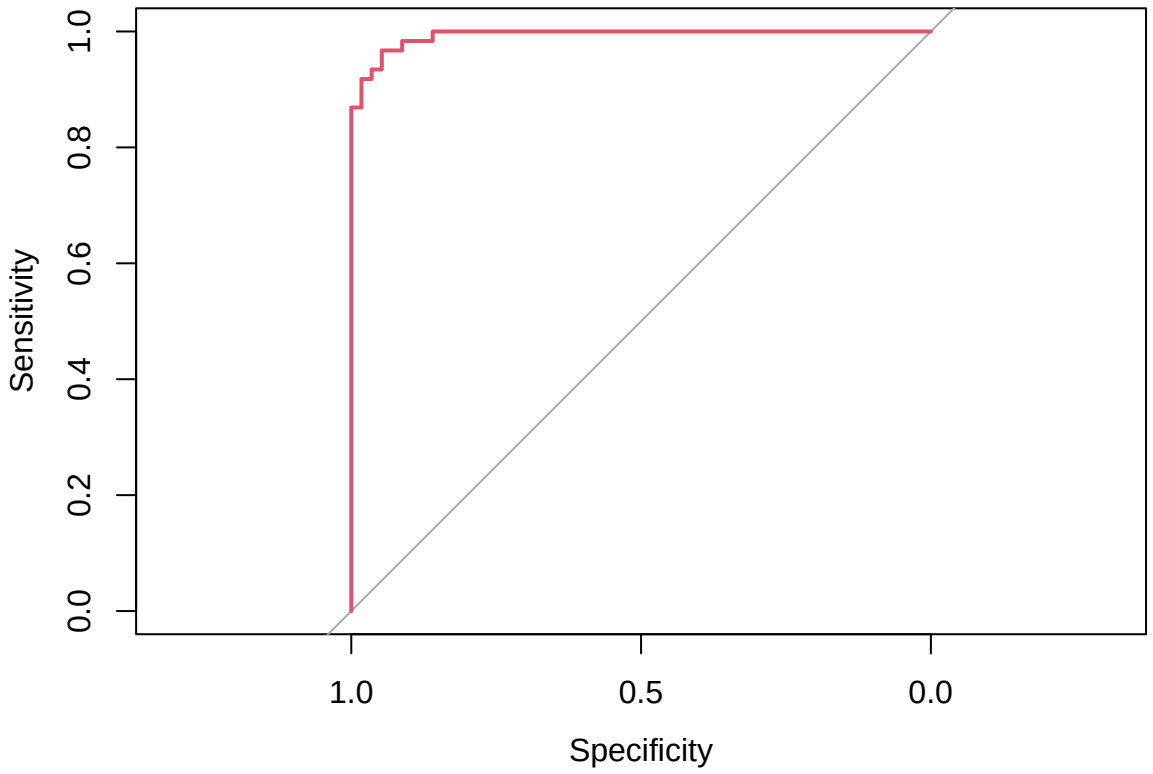

```
##                       var    rel.inf
## displacement displacement 37.3155828
## weight             weight 22.9991686
## horsepower     horsepower 11.5830413
## year                 year 10.8342277
## cylinders       cylinders  9.5051043
## acceleration acceleration  6.8713489
## origin3           origin3  0.6409769
## origin2           origin2  0.2505493
```

```
pred_bst <- predict(bst.fit, newdata = testing_dat, type = "prob")[,1]
roc_bst<-roc(testing_dat$mpg_cat, pred_bst)
```

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

```
## Setting direction: controls > cases
```

```
auc <- roc_bst$auc[1]
plot(roc_bst,col=2)
```




```
pred_bst <- predict(bst.fit, newdata = testing_dat)
confusionMatrix(pred_bst, testing_dat$mpg_cat)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction high low
##       high   55   4
##       low     2  57
##
##                Accuracy : 0.9492
##                  95% CI : (0.8926, 0.9811)
##     No Information Rate : 0.5169
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8983
##
##  Mcnemar's Test P-Value : 0.6831
##
##             Sensitivity : 0.9649
##             Specificity : 0.9344
##          Pos Pred Value : 0.9322
##          Neg Pred Value : 0.9661
##              Prevalence : 0.4831
##          Detection Rate : 0.4661
##    Detection Prevalence : 0.5000
##       Balanced Accuracy : 0.9497
##
##        'Positive' Class : high
```

##

From the variable importance table, the displacement is the most important variable, the second is weight, and the rest are ordered in a descending way.

According to the model performance metrics, auc = 0.9930975, accuracy = 0.9491525, and both sensitivity and sepcificity are high in confusion matrix. Therefore, we believe the model perform well.