Chapter 8 Tree-Based Methods

Jishen Yin

2020/5/13

```
knitr::opts_chunk$set(echo = TRUE)
library(ISLR)
library(MASS)
library(tree)
library(gbm)
library(randomForest)
library(tidyverse)
```

Problem 8

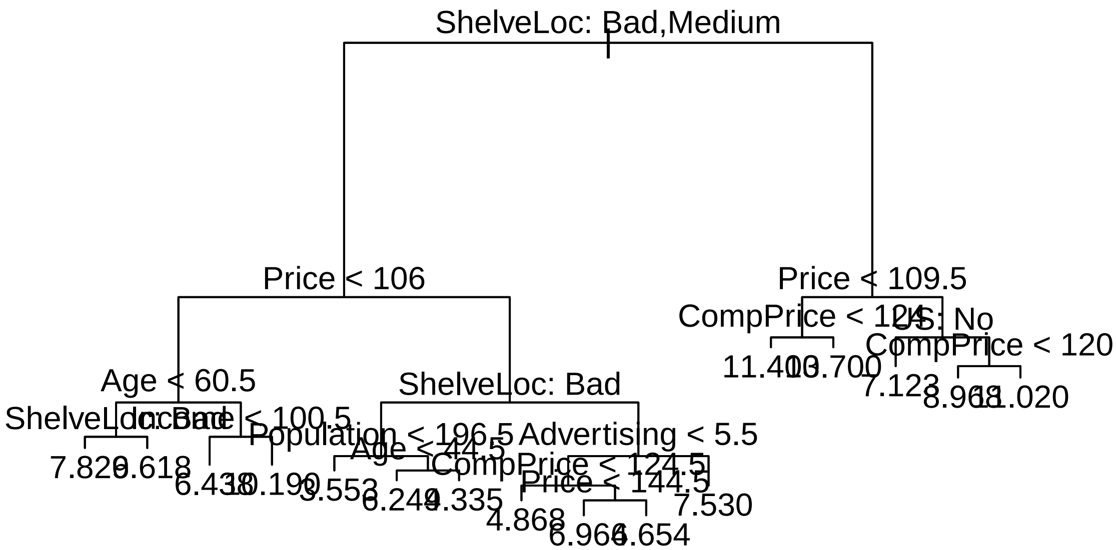
We seek to predict `Sales` in `Carseats` data set using regression trees and related approaches.

(a) Split the data set into a training set and a test set.

```
data(Carseats)

set.seed(7)
train <- sample(1:400, 300)
Carseats_train <- Carseats[train,]
Carseats_val <- Carseats[-train,]
```

(b) Fit a regression tree to the training set. Plot the tree, and interpret the results. What test MSE do you obtain?

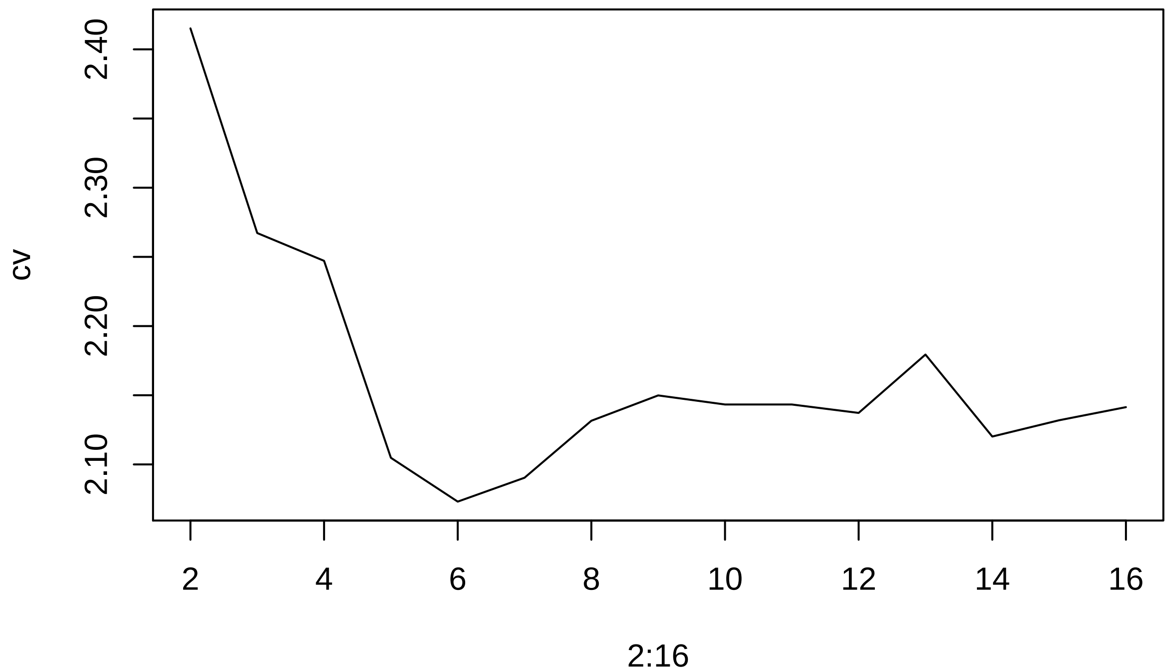
```
set.seed(9)
tree.Carseats <- tree(Sales~., data = Carseats_train)
plot(tree.Carseats)
text(tree.Carseats, pretty = 0)
```



(c) Using cross-validation in order to determine the optimal level of tree complexity. Does pruning the tree improve the test MSE?

```r
RMSE <- function(y_pred, y){
  return(sqrt(mean((y_pred - y)**2)))
}

cv <- sapply(2:16, function(x){
  prune <- prune.tree(tree.Carseats, best = x)
  y_pred <- predict(prune, Carseats_val)
  return(RMSE(y_pred, Carseats_val$Sales))
})
```

```r
plot(2:16, cv, type = "l")
```



When depth is 6, we got the optimal test RMSE.

(d) Use the bagging approach in order to analyze the data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important.

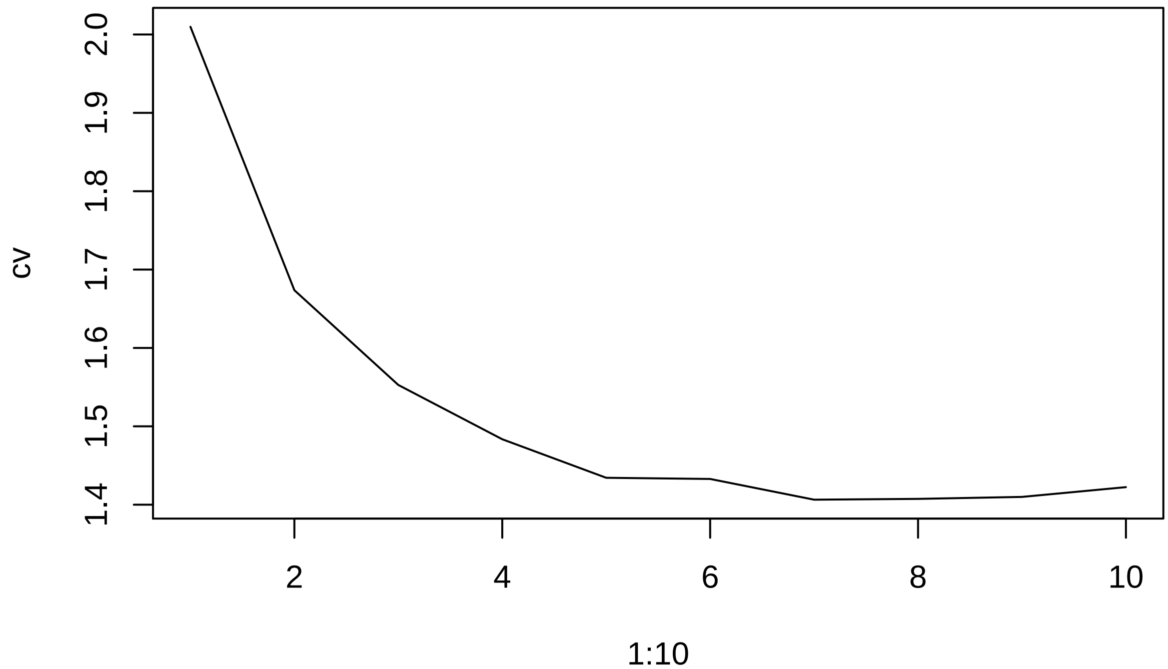
```
set.seed(3)

bag.Carseats <- randomForest(Sales~., data = Carseats_train, mtry = 10, importance = TRUE)

RMSE(predict(bag.Carseats, Carseats_val), Carseats_val$Sales)
```

```
## [1] 1.409598
```

(e) Use random forests to analyze this data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important. Describe the effect of m, the number of variables considered at each split, on the error rate obtained.

```
set.seed(32)
cv <- sapply(1:10, function(x){
  rf.Carseats <- randomForest(Sales~., data = Carseats_train, mtry = x, importance = TRUE)
  y_pred <- predict(rf.Carseats, Carseats_val)
  return(RMSE(y_pred, Carseats_val$Sales))
})
```

```r
plot(1:10, cv, type = "l")
```



When $m = 8$, we got the best RMSE.

```r
rf.Carseats <- randomForest(Sales~., data = Carseats_train, mtry = 8, importance = TRUE)
importance(rf.Carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice   28.293222     238.91283
## Income       9.078488     134.11198
## Advertising 25.407866     202.44837
## Population  -2.695641      83.55657
## Price       63.642372     620.88773
## ShelveLoc   71.894120     777.85546
## Age         19.835362     223.15078
## Education    2.623236      74.83604
## Urban       -0.900073      11.66069
## US           6.426173      21.87461
```

Problem 9

This problem involves the `OJ` data set.

(a) Create a training set containing 800 observations, and a test set containing the remaining observations.

```r
data(OJ)

set.seed(3)
train <- sample(1:nrow(OJ), 800)
OJ_train <- OJ[train,]
OJ_val <- OJ[-train,]
```

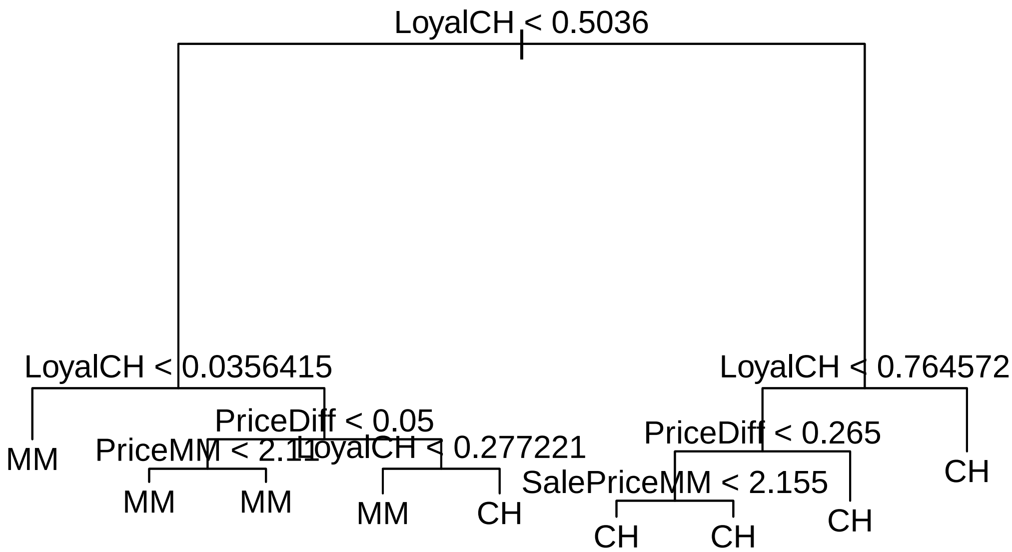
(b) Fit a tree to the training data, with `Purchase` as the response and other variables as predictors.

```r
tree.oj <- tree(Purchase ~., data = OJ_train)
summary(tree.oj)
```

```
##
## Classification tree:
## tree(formula = Purchase ~ ., data = OJ_train)
## Variables actually used in tree construction:
## [1] "LoyalCH"     "PriceDiff"   "PriceMM"     "SalePriceMM"
## Number of terminal nodes:  9
## Residual mean deviance:  0.7247 = 573.2 / 791
## Misclassification error rate: 0.1812 = 145 / 800
```

(d) Create a plot of the tree.

```r
plot(tree.oj)
text(tree.oj)
```



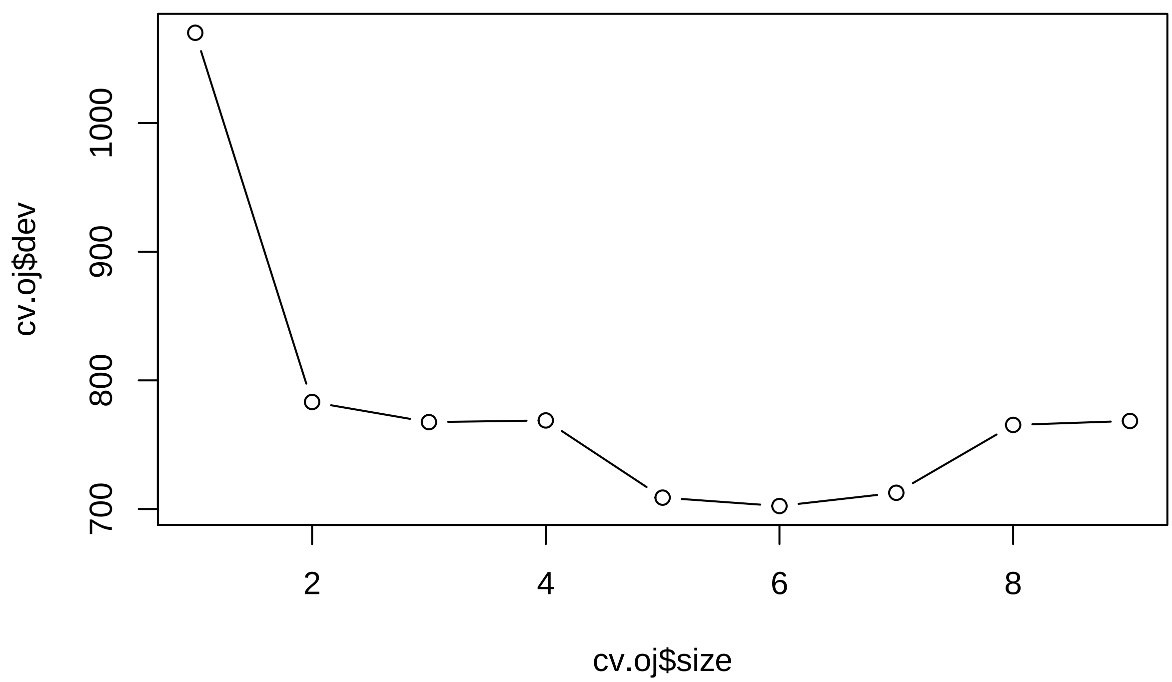
(e) Predict the response on the test data, and produce a confusion matrix.

```
y_pred <- ifelse(predict(tree.oj, OJ_val)[,1] > 0.5, "CH", "MM")
table(y_pred, OJ_val$Purchase)
```

```
##
## y_pred  CH  MM
##     CH 148  31
##     MM  15  76
```

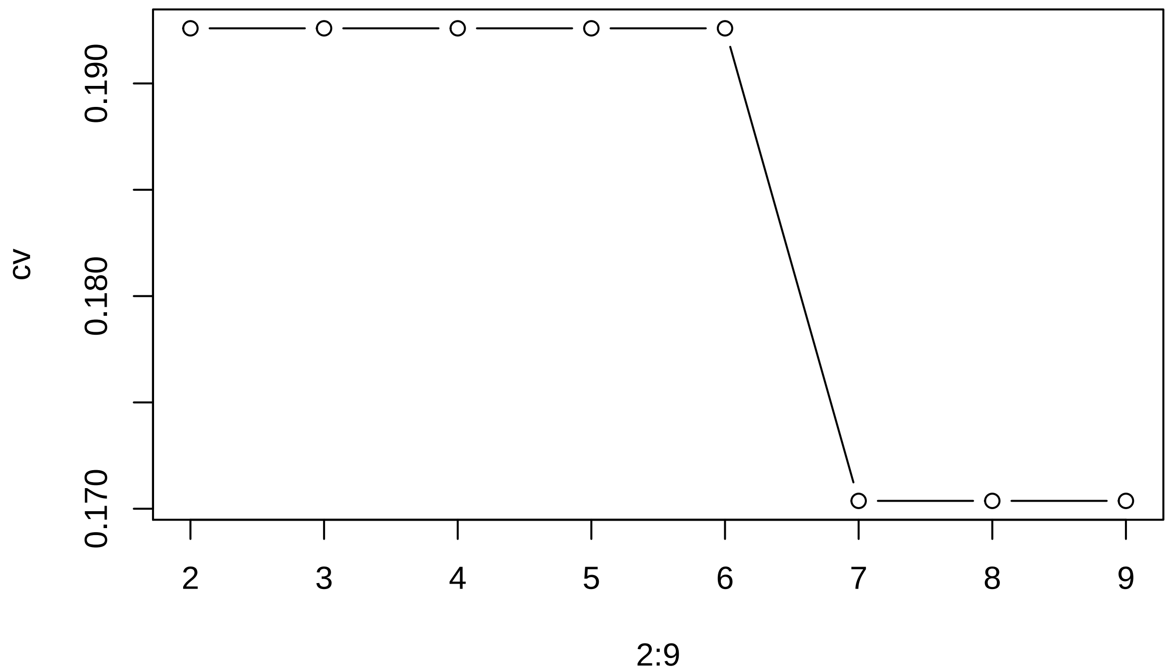
(f) Apply `cv.tree()` function to the training set in order to determine the optimal tree size.

```
set.seed(1)
cv.oj <- cv.tree(tree.oj)
plot(cv.oj$size, cv.oj$dev, type = "b")
```



(g) Produce a plot with tree size on the x-axis and cross-validated classification error rate on the y-axis.

```
cv <- sapply(2:9, function(x){
  prune.oj <- prune.tree(tree.oj, best = x)
  y_pred <- ifelse(predict(prune.oj, OJ_val)[,1] > 0.5, "CH", "MM")
  return(mean(y_pred != OJ_val$Purchase))
})

plot(2:9, cv, type = "b")
```

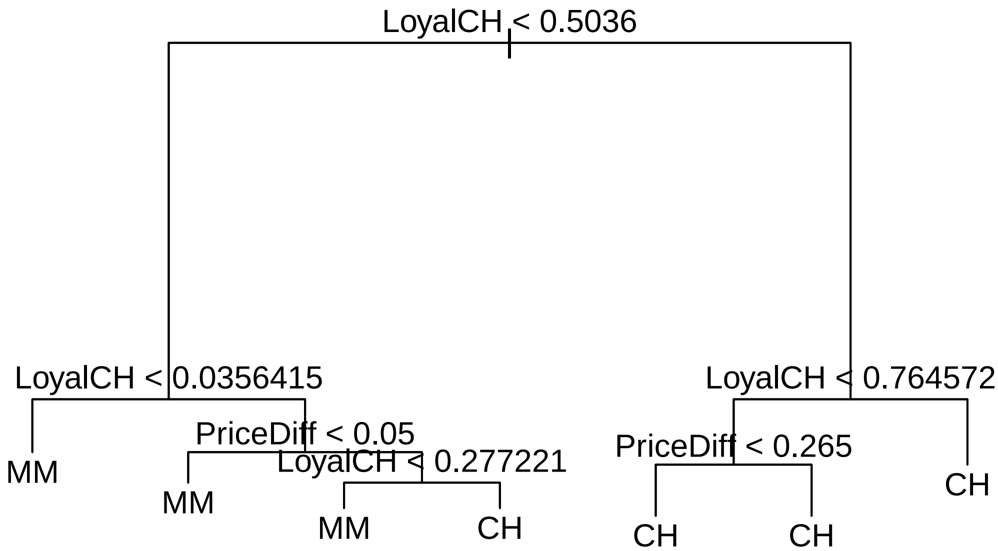


(h) Which tree size corresponds to the lowest cross-validated classification error rate?

When the size is 7, we got the lowest cv classification error rate.

(i) Produce a pruned tree corresponding to the optimal tree size obtained using cross-validation. If cross-validation does not lead to selection of a pruned tree, then created a pruned tree with five terminal nodes.

```
prune.oj <- prune.tree(tree.oj, best = 7)
plot(prune.oj)
text(prune.oj)
```



Problem 10

We now use boosting to predict `Salary` in the `Hitters` data set.

(a) Remove the observations for whom the salary information is unknown, and then log-transform the salaries.

```
data(Hitters)

Hitters <- Hitters %>%
  na.omit(Salary) %>%
  mutate(log_Salary = log(Salary)) %>%
  select(-Salary)
```

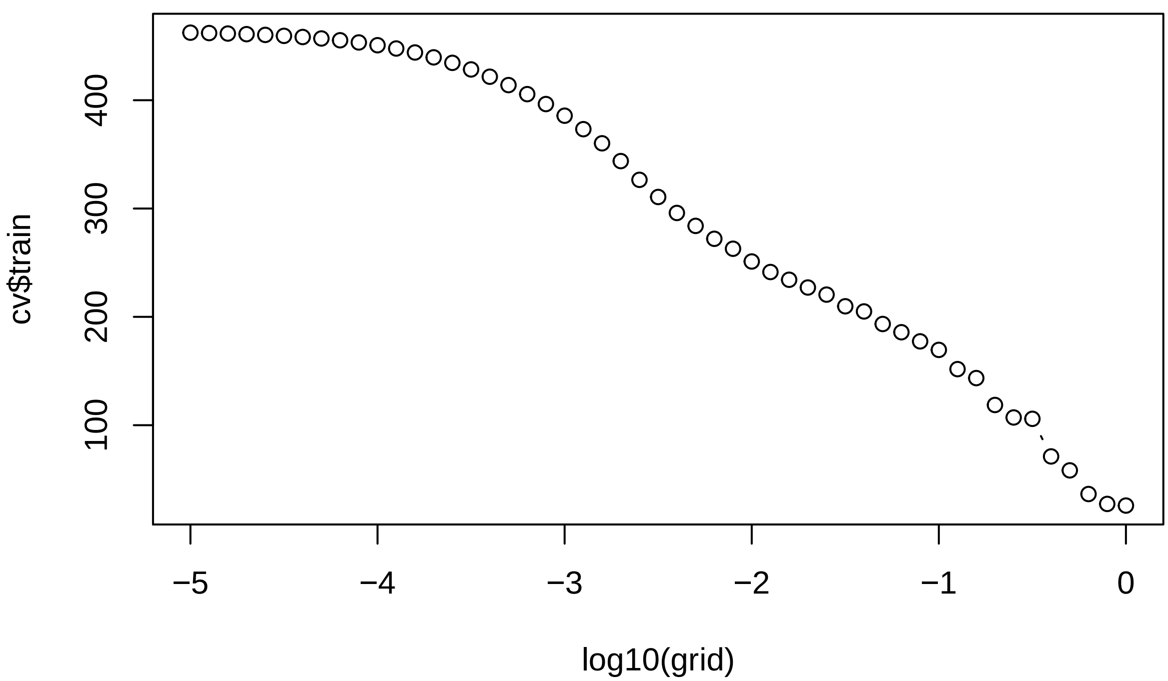
(b) Create a training set consisting of the first 200 observations, and a test set consisting of the remaining observations.

```
set.seed(65)

train <- sample(1:nrow(Hitters), 200)
Hitters_train <- Hitters[train, ]
Hitters_val <- Hitters[-train, ]
```

(c) Perform boosting on the training set with 1,000 trees for a range of values of the shrinkage paramter λ. Product a plot with different shrinkage values on the x-axis and the corresponding training set MSE on the y-axis.

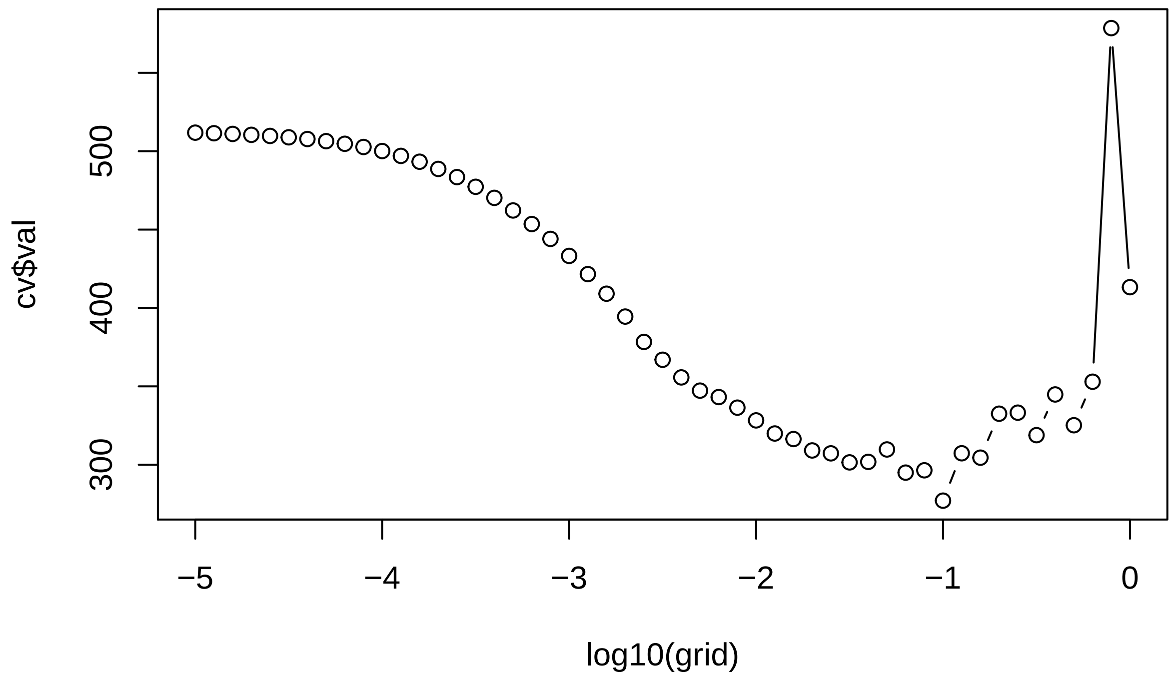
```
grid <- 10^(seq(-5, 0, 0.1))
set.seed(23)

cv <- lapply(grid, function(lambda){
  boost.hitters <- gbm(log_Salary ~., data = Hitters_train, n.trees = 1000,
                       distribution = "gaussian", shrinkage = lambda)
  y_pred_train <- predict(boost.hitters, Hitters_train, n.trees = 1000)
  y_pred_val <- predict(boost.hitters, Hitters_val, n.trees = 1000)
  return(data.frame(train = RMSE(exp(y_pred_train), exp(Hitters_train$log_Salary)),
                    val = RMSE(exp(y_pred_val), exp(Hitters_val$log_Salary))))
})

cv <- do.call(rbind, cv)
```

```
plot(log10(grid), cv$train, type = "b")
```



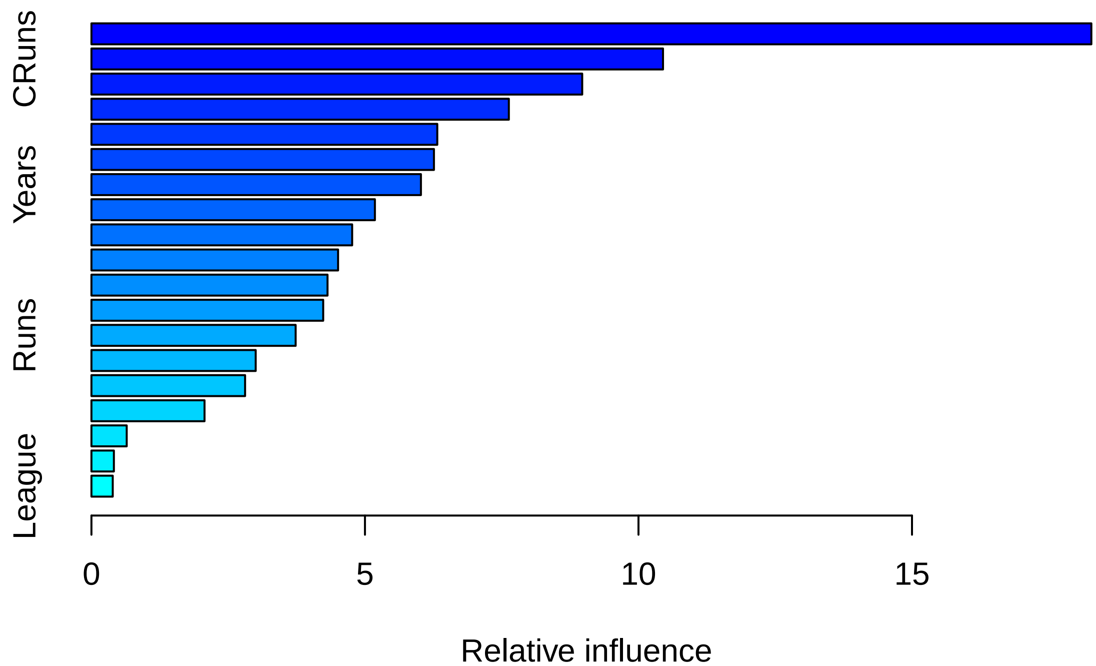
(d) Product a plot with different shrinkage values on the x-axis and the corresponding test set MSE on the y-axis.

```r
plot(log10(grid), cv$val, type = "b")
```



(e) Which variables appear to be the most important predictors in the boosted model?

```r
boost.hitters <- gbm(log_Salary ~., data = Hitters_train, n.trees = 1000,
                     distribution = "gaussian", shrinkage = 0.1)
summary(boost.hitters)
```




```
##                 var    rel.inf
## CAtBat       CAtBat 18.2785765
## CRuns         CRuns 10.4484487
## PutOuts     PutOuts  8.9715869
## CWalks       CWalks  7.6302630
## AtBat         AtBat  6.3219115
## Walks         Walks  6.2609858
## Years         Years  6.0230206
## CHmRun       CHmRun  5.1819895
## CHits         CHits  4.7658215
## Hits           Hits  4.5084582
## HmRun         HmRun  4.3154403
## RBI             RBI  4.2364835
## Runs           Runs  3.7328083
## Assists     Assists  3.0030075
## CRBI           CRBI  2.8086998
## Errors       Errors  2.0673748
## Division   Division  0.6450611
## NewLeague NewLeague  0.4105201
## League       League  0.3895424
```