Assignment 7

Tina Roha

4/30/2021

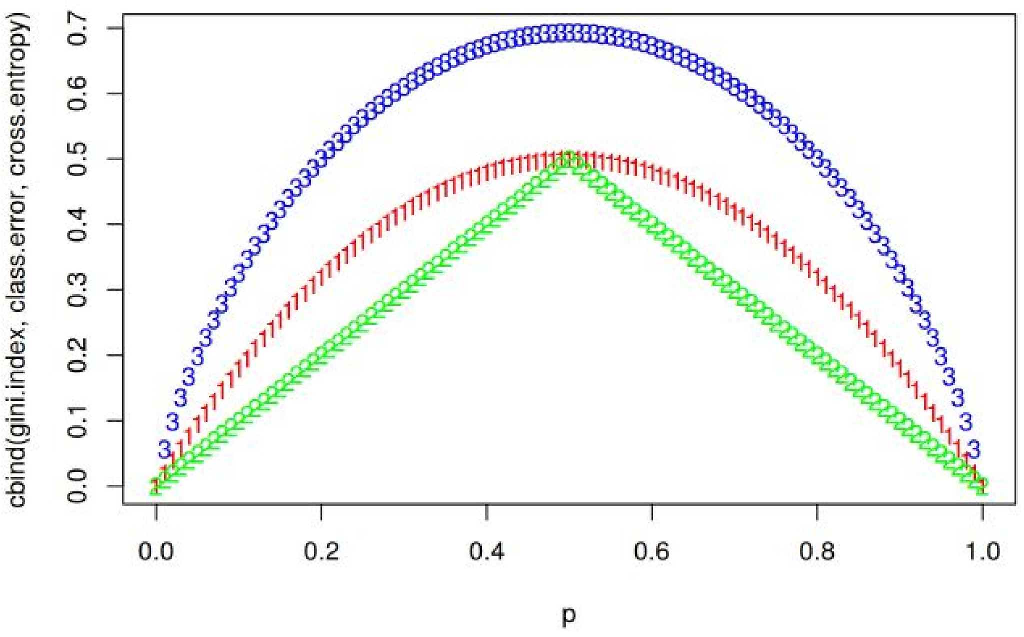
Tree-Based Methods

Chapter 08 (page 332): Questions 3, 8 and 9

Question-3

Consider the Gini index, classification error, and entropy in a simple classification setting with two classes. Create a single plot that displays each of these quantities as a function of ˆpm1. The x-axis should display ˆpm1, ranging from 0 to 1, and the y-axis should display the value of the Gini index, classification error, and entropy. Hint: In a setting with two classes, pˆm1 = 1 − pˆm2. You could make this plot by hand, but it will be much easier to make in R.

```
p <- seq(0, 1, 0.01)
gini.index <- 2 * p * (1 - p)
class.error <- 1 - pmax(p, 1 - p)
cross.entropy <- - (p * log(p) + (1 - p) * log(1 - p))
matplot(p, cbind(gini.index, class.error, cross.entropy), col = c("red", "green", "blue"))
```



Question-8

In the lab, a classification tree was applied to the Carseats data set after converting Sales into a qualitative response variable. Now we will seek to predict Sales using regression trees and related approaches, treating the response as a quantitative variable.

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

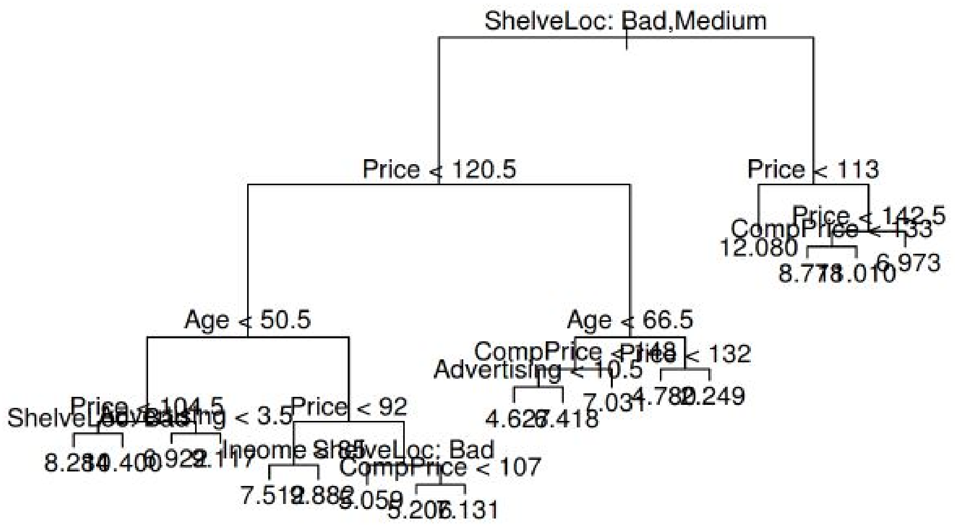
```
library(ISLR)
set.seed(1)
train <- sample(1:nrow(Carseats), nrow(Carseats) / 2)
Carseats.train <- Carseats[train, ]
Carseats.test <- Carseats[-train, ]
```

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

```r
library(tree)
tree.carseats <- tree(Sales ~ ., data = Carseats.train)
summary(tree.carseats)
```

```
##
## Regression tree:
## tree(formula = Sales ~ ., data = Carseats.train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Age"         "Advertising" "Income"
## [6] "CompPrice"
## Number of terminal nodes:  18
## Residual mean deviance:  2.36 = 429.5 / 182
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -4.2570 -1.0360  0.1024  0.0000  0.9301  3.9130
```

```r
plot(tree.carseats)
text(tree.carseats, pretty = 0)
```



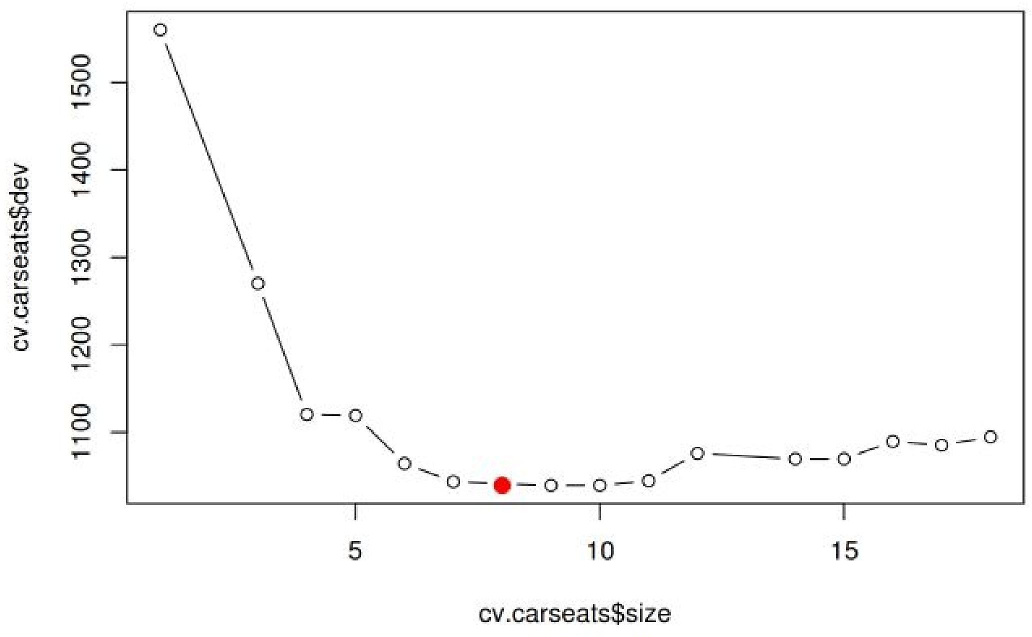
```r
yhat <- predict(tree.carseats, newdata = Carseats.test)
mean((yhat - Carseats.test$Sales)^2)
```

```
## [1] 4.148897
```

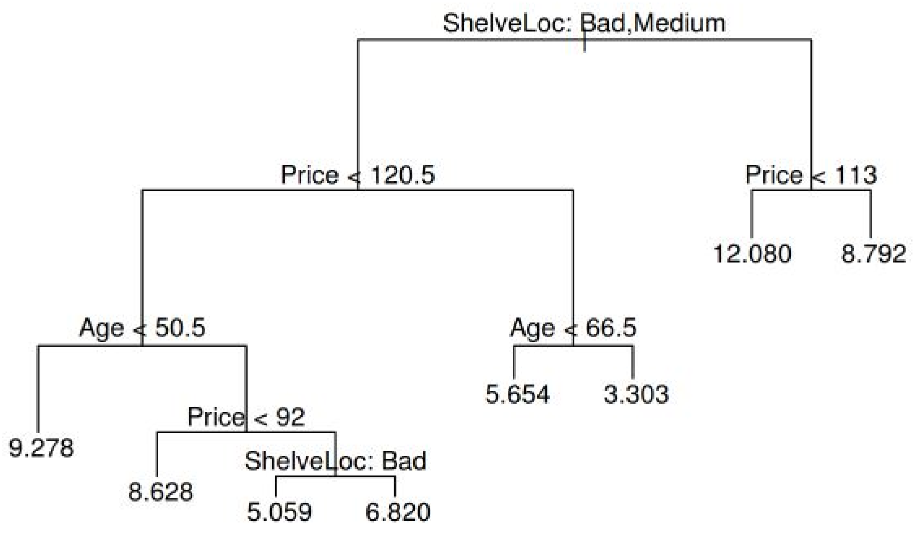
The test MSE obtained is 4.15.

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

```
cv.carseats <- cv.tree(tree.carseats)
plot(cv.carseats$size, cv.carseats$dev, type = "b")
tree.min <- which.min(cv.carseats$dev)
points(tree.min, cv.carseats$dev[tree.min], col = "red", cex = 2, pch = 20)
```



```
prune.carseats <- prune.tree(tree.carseats, best = 8)
plot(prune.carseats)
text(prune.carseats, pretty = 0)
```



```
yhat <- predict(prune.carseats, newdata = Carseats.test)
mean((yhat - Carseats.test$Sales)^2)
```

```
## [1] 5.09085
```

Pruning the tree increases the test MSE to the value of 5.1.

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

```
bag.carseats <- randomForest(Sales ~ ., data = Carseats.train, mtry = 10, ntree = 500, importance = TRUE)
yhat.bag <- predict(bag.carseats, newdata = Carseats.test)
mean((yhat.bag - Carseats.test$Sales)^2)
```

```
## [1] 2.604369
```

```
importance(bag.carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice   14.4124562    133.731797
## Income       6.5147532     74.346961
## Advertising 15.7607104    117.822651
## Population   0.6031237     60.227867
## Price       57.8206926    514.802084
## ShelveLoc   43.0486065    319.117972
## Age         19.8789659    192.880596
## Education    2.9319161     39.490093
## Urban       -3.1300102      8.695529
## US           7.6298722     15.723975
```

Utilizing the bagging approach, the test MSE obtained is 2.6. Furthermore, it can be concluded that the two most important variables are “Price” and “ShelveLoc”.

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.

```
rf.carseats <- randomForest(Sales ~ ., data = Carseats.train, mtry = 3, ntree = 500, importance = TRUE)
yhat.rf <- predict(rf.carseats, newdata = Carseats.test)
mean((yhat.rf - Carseats.test$Sales)^2)
```

```
## [1] 3.296078
```

```
importance(rf.carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice    7.5233429     127.36625
## Income       4.3612064     119.19152
## Advertising 12.5799388     138.13567
## Population  -0.2974474     100.28836
## Price       37.1612032     383.12126
## ShelveLoc   30.3751253     246.19930
## Age         16.0261885     197.44865
## Education    1.7855151      63.87939
## Urban       -1.3928225      16.01173
## US           5.6393475      32.85850
```

Utilizing random forest to analyze the data the test MSE obtained is 3.3. Furthermore, it can be concluded that the two most important variables are also “Price” and “ShelveLoc”.

Question-9

This problem involves the OJ data set which is part of the ISLR package.

a. Create a training set containing a random sample of 800 observations, and a test set containing the remaining observations.

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

b. Fit a tree to the training data, with Purchase as the response and the other variables as predictors. Use the summary() function to produce summary statistics about the tree, and describe the results obtained. What is the training error rate? How many terminal nodes does the tree have?

```
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"     "SpecialCH"     "ListPriceDiff"
## Number of terminal nodes:  8
## Residual mean deviance:  0.7305 = 578.6 / 792
## Misclassification error rate: 0.165 = 132 / 800
```

In this case, the training error rate is 0.165 and the tree has a total of 8 terminal nodes.

c. Type in the name of the tree object in order to get a detailed text output. Pick one of the terminal nodes, and interpret the information displayed.

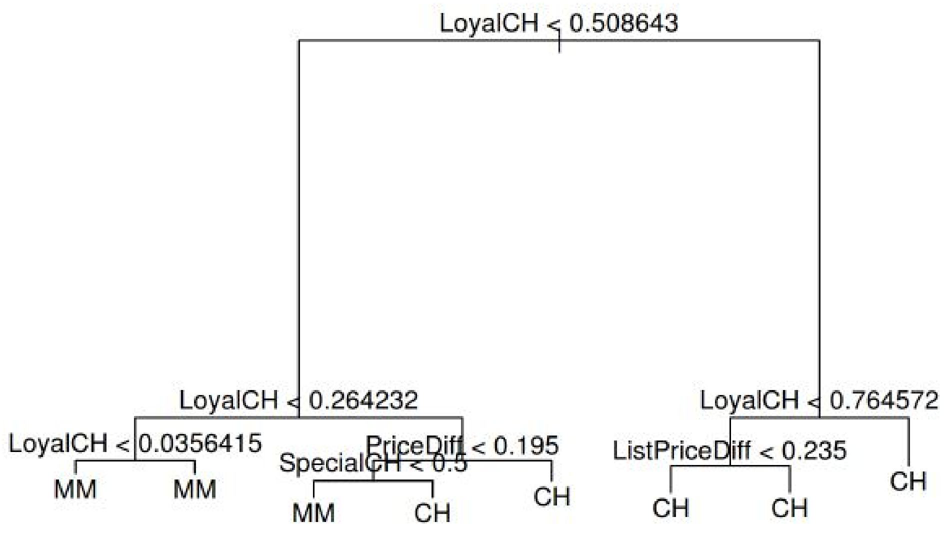
```
tree.oj
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 800 1064.00 CH ( 0.61750 0.38250 )
##    2) LoyalCH < 0.508643 350  409.30 MM ( 0.27143 0.72857 )
##      4) LoyalCH < 0.264232 166  122.10 MM ( 0.12048 0.87952 )
##        8) LoyalCH < 0.0356415 57   10.07 MM ( 0.01754 0.98246 ) *
##        9) LoyalCH > 0.0356415 109  100.90 MM ( 0.17431 0.82569 ) *
##      5) LoyalCH > 0.264232 184  248.80 MM ( 0.40761 0.59239 )
##       10) PriceDiff < 0.195 83   91.66 MM ( 0.24096 0.75904 )
##         20) SpecialCH < 0.5 70   60.89 MM ( 0.15714 0.84286 ) *
##         21) SpecialCH > 0.5 13   16.05 CH ( 0.69231 0.30769 ) *
##       11) PriceDiff > 0.195 101  139.20 CH ( 0.54455 0.45545 ) *
##    3) LoyalCH > 0.508643 450  318.10 CH ( 0.88667 0.11333 )
##      6) LoyalCH < 0.764572 172  188.90 CH ( 0.76163 0.23837 )
##       12) ListPriceDiff < 0.235 70   95.61 CH ( 0.57143 0.42857 ) *
##       13) ListPriceDiff > 0.235 102   69.76 CH ( 0.89216 0.10784 ) *
##      7) LoyalCH > 0.764572 278   86.14 CH ( 0.96403 0.03597 ) *
```

In this case, the terminal node selected for interpretation is the node labeled 9, which is classified as a node due to the asterisk. Furthermore, for terminal node 9, it can be concluded that the split criterion is LoyalCH > 0.035, the number of observations in the branch is 109 with a deviance of 100.90 and the overall prediction for the branch is MM. Additionally, it can be concluded that only 0.17 of the observations in this branch have CH as its value and the other 0.82 have MM as its value.

d. Create a plot of the tree, and interpret the results.

```
plot(tree.oj)
text(tree.oj, pretty = 0)
```



The results from the output conclude that “LoyalCH” is the greatest indicator of “Purchase”. Furthemore, this is due to the fact that the first branch expresses the strong impact of customer brand loyalty to CH. Additionally, this conclusion can be verified due to the fact that the top three nodes also show “LoyalCH”.

e. Predict the response on the test data, and produce a confusion matrix comparing the test labels to the predicted test labels. What is the test error rate?

```
tree.pred <- predict(tree.oj, OJ.test, type = "class")
table(tree.pred, OJ.test$Purchase)
```

```
##
## tree.pred  CH  MM
##        CH 147  49
##        MM  12  62
```

```
1 - (147 + 62) / 270
```

```
## [1] 0.2259259
```

From the results produced it is evident that the test error rate is approximately 22%.

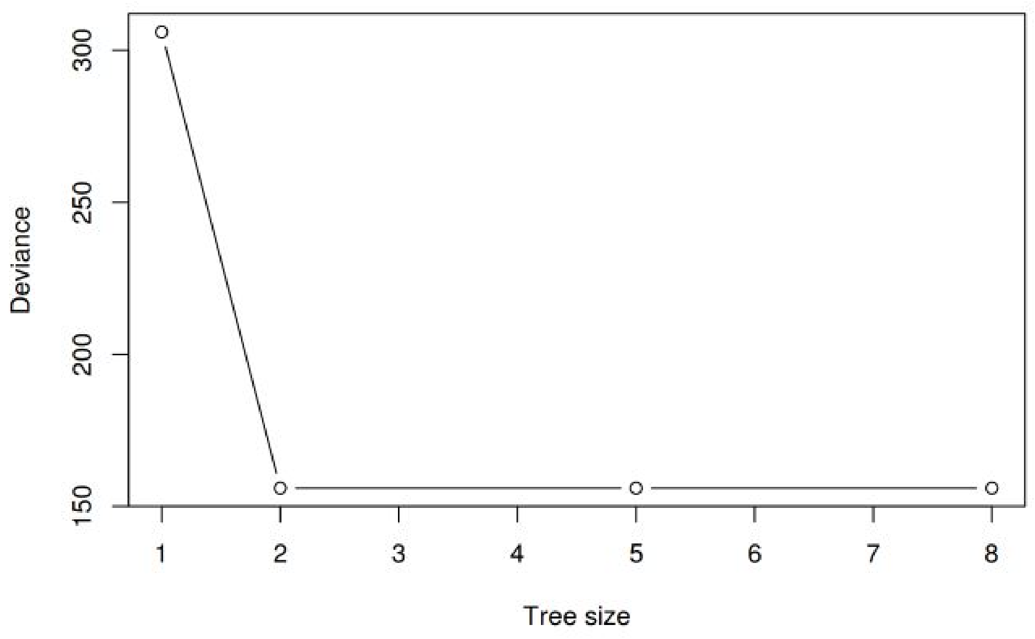
f. Apply the cv.tree() function to the training set in order to determine the optimal tree size.

```
cv.oj <- cv.tree(tree.oj, FUN = prune.misclass)
cv.oj
```

```
## $size
## [1] 8 5 2 1
##
## $dev
## [1] 156 156 156 306
##
## $k
## [1]       -Inf   0.000000   4.666667 160.000000
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

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

```
plot(cv.oj$size, cv.oj$dev, type = "b", xlab = "Tree size", ylab = "Deviance")
```

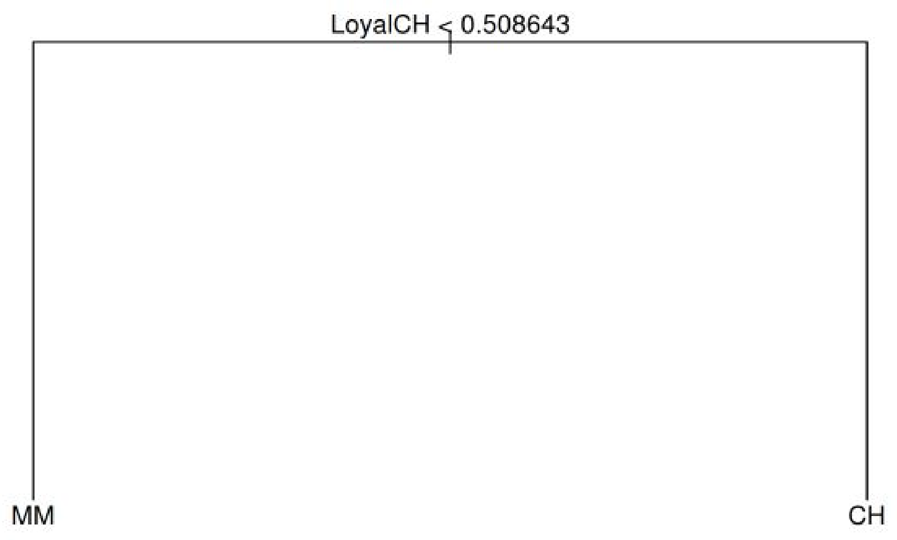


h. Which tree size corresponds to the lowest cross-validated classification error rate?

The tree size that corresponds to the lowest cross-validated classification error rate is the 2-node tree.

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 create a pruned tree with five terminal nodes.

```r
prune.oj <- prune.misclass(tree.oj, best = 2)
plot(prune.oj)
text(prune.oj, pretty = 0)
```



j. Compare the training error rates between the pruned and unpruned trees. Which is higher?

```r
summary(tree.oj)
```

```
##
## Classification tree:
## tree(formula = Purchase ~ ., data = OJ.train)
## Variables actually used in tree construction:
## [1] "LoyalCH"       "PriceDiff"     "SpecialCH"     "ListPriceDiff"
## Number of terminal nodes:  8
## Residual mean deviance:  0.7305 = 578.6 / 792
## Misclassification error rate: 0.165 = 132 / 800
```

```r
summary(prune.oj)
```

```
##
## Classification tree:
## snip.tree(tree = tree.oj, nodes = c(3L, 2L))
## Variables actually used in tree construction:
## [1] "LoyalCH"
## Number of terminal nodes:  2
## Residual mean deviance:  0.9115 = 727.4 / 798
## Misclassification error rate: 0.1825 = 146 / 800
```

By comparing the training error rates between the pruned and unpruned trees, the results show that the pruned trees have a value of 0.1825 and the unpruned trees have a value of 0.165. Therefore, it can be concluded that the pruned trees have a misclassification error rate slightly higher than the unpruned trees.

k. Compare the test error rates between the pruned and unpruned trees. Which is higher?

```
prune.pred <- predict(prune.oj, OJ.test, type = "class")
table(prune.pred, OJ.test$Purchase)
```

```
##
## prune.pred  CH  MM
##         CH 119  30
##         MM  40  81
```

```
1 - (119 + 81) / 270
```

```
## [1] 0.2592593
```

By comparing the test error rates between the pruned and unpruned trees, the results show that the pruned trees caused the test error rate to enlarge to approximately 26%, but overall it produced a more interpretable tree.