Computational Statistics II

Lab 8 Homework

Subash Kharel

April 17, 2020

1. **This problem involves the OJ data set which is part of the ISLR package**
   a. **Create a training set having a random sample of 800 observations, and a test set containing the remaining observations.**

```
library(ISLR)
library(tree)
attach(OJ)

set.seed(1000)
train_sample = sample(1:nrow(OJ), 800)
OJ.test = OJ[-train,]
OJ.train = OJ[train,]
Purchase.test = Purchase[-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~., OJ)
summary(tree.oj)
```

```
Classification tree:
tree(formula = Purchase ~ ., data = OJ.train)
Variables actually used in tree construction:
[1] "LoyalCH"      "PriceDiff"    "WeekofPurchase"
Number of terminal nodes:  7
Residual mean deviance:  0.7848 = 622.4 / 793
Misclassification error rate: 0.175 = 140 / 800
```

Training error rate is 17.5% and number of terminal nodes is 7

   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
```

```
1) root 800 1069.000 CH ( 0.61125 0.38875 )
  2) LoyalCH < 0.482389 297  319.600 MM ( 0.22896 0.77104 )
    4) LoyalCH < 0.0356415 55    9.996 MM ( 0.01818 0.98182 ) *
    5) LoyalCH > 0.0356415 242  285.500 MM ( 0.27686 0.72314 )
     10) PriceDiff < 0.31 188  197.200 MM ( 0.21809 0.78191 )
       20) WeekofPurchase < 274.5 166  185.600 MM ( 0.24699 0.75301 ) *
       21) WeekofPurchase > 274.5 22    0.000 MM ( 0.00000 1.00000 ) *
     11) PriceDiff > 0.31 54   74.790 MM ( 0.48148 0.51852 ) *
  3) LoyalCH > 0.482389 503  447.300 CH ( 0.83698 0.16302 )
    6) LoyalCH < 0.753545 235  284.500 CH ( 0.70638 0.29362 )
     12) PriceDiff < 0.015 72   98.420 MM ( 0.43056 0.56944 ) *
     13) PriceDiff > 0.015 163  149.500 CH ( 0.82822 0.17178 ) *
    7) LoyalCH > 0.753545 268  104.000 CH ( 0.95149 0.04851 ) *
```

Interpreting the terminal node 21: Week of Purchase > 274.5 22 0.00 MM (0.00000 1.00000)

Number of items = 22

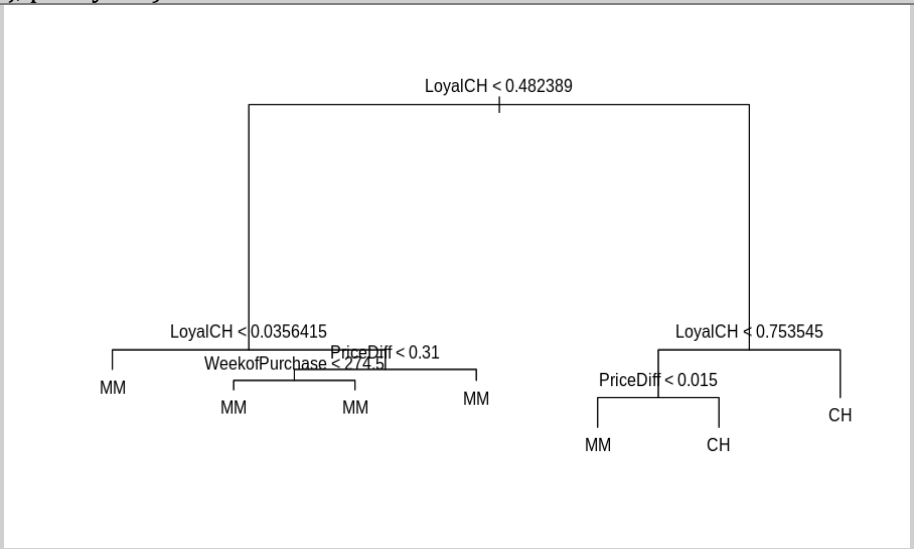
Deviance = 0.00

Overall prediction = MM

Fraction of observation taking values CH and MM = (0, 1)

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

```
plot(tree.oj)
text(tree.oj, pretty = 1)
```



Looking at the tree, we can say that only three variables can be used to decide the class for Purchase variable. The most dominating variable in the data is the LoyalCH. If the value of LoyalCH becomes less than 0.4823 for a data, it has high probability of being classified as MM. If its value is > 0.753545, it will be classified as CH.

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?

```
pred = predict(tree.oj,OJ.test, type="class")
table(pred, Purchase.test)
test.error = round(mean(pred != Purchase.test)*100,2)
test.error
```

```
>  Purchase.test
pred  CH   MM
 CH  123  12
 MM  41   94
>test.error
 [1]  19.63
```

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

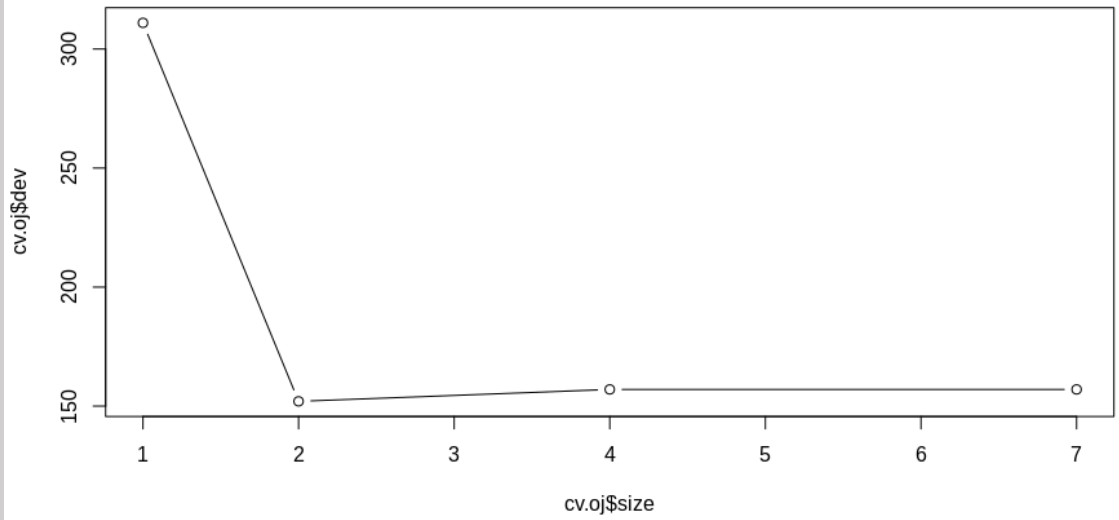
```
cv.oj = cv.tree(tree.oj, FUN=prune.misclass)
cv.oj
```

```
$size
[1] 7 4 2 1

$dev
[1] 157 157 152 311

$k
[1]  -Inf    0    5  161

$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.

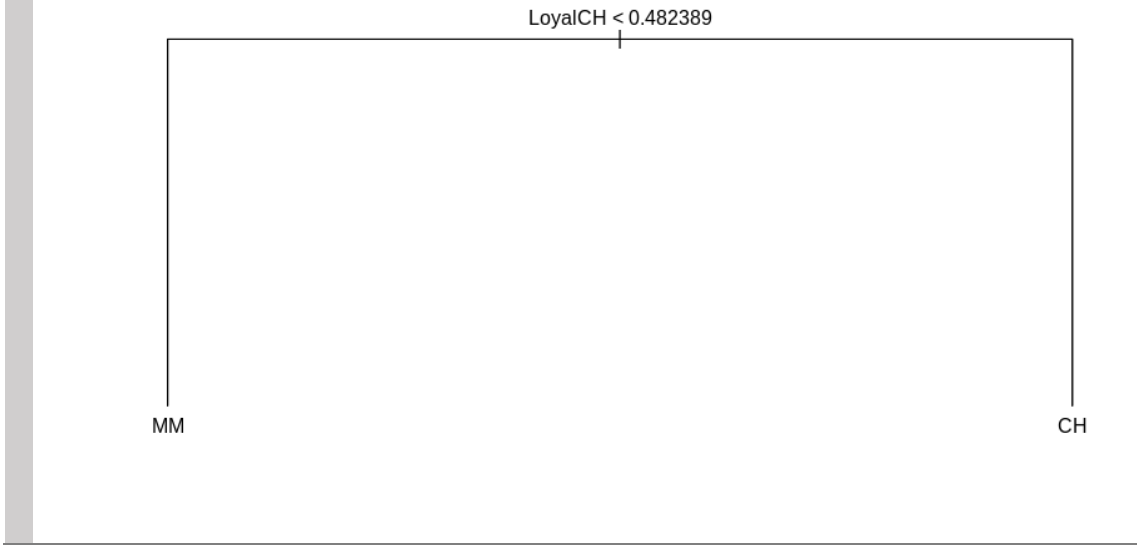
```
plot(cv.oj$size, cv.oj$dev, type="b")
```



h. **Which tree size corresponds to the lowest cross-validated classification error rate?**
Answer: Tree size of 2 corresponds to the lowest cross-validated error rate from the graph above.

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.

```
prune.oj =prune.misclass (tree.oj ,best=2)
plot(prune.oj)
text(prune.oj)
```



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

```
tree.train.pruned = predict(prune.oj, OJ.train, type="class")
table(tree.train.pruned, OJ.train$Purchase)
train.error.pruned = round(mean(tree.train.pruned != OJ.train$Purchase)*100,2)
train.error.pruned
cat("Pruned: Train Error :", train.error.pruned)


tree.train.unpruned = predict(tree.oj, OJ.train, type="class")
table(tree.train.unpruned, OJ.train$Purchase)
train.error.unpruned = round(mean(tree.train.unpruned != OJ.train$Purchase)*100,2)
train.error.unpruned
cat("Unpruned: Train Error :", train.error.unpruned)
```

```
Pruned: Train Error : 18.75
Unpruned: Train Error : 17.5
```

Training error rate for pruned tree is higher.

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

```
tree.test.pruned = predict(prune.oj, OJ.test, type="class")
table(tree.test.pruned, Purchase.test)
test.error.pruned = round(mean(tree.test.pruned != Purchase.test)*100,2)
test.error.pruned
cat("Pruned: Test Error :", test.error.pruned)


tree.test.unpruned = predict(tree.oj, OJ.test, type="class")
table(tree.test.unpruned, Purchase.test)
test.error.unpruned = round(mean(tree.test.unpruned != Purchase.test)*100,2)
test.error.unpruned
cat("Unpruned: Test Error :", test.error.unpruned)
```

```
Pruned: Test Error : 20
Unpruned: Test Error : 19.63
```

Test error rate for pruned tree is higher.