H4_S21_P1

Naveen

4/25/2021

Problem 1

This problem use the OJ data set (OJ.csv).

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

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?

```
oj.ds.tree.fit<-tree(Purchase~.,oj.ds.train)
summary(oj.ds.tree.fit)
```

```
##
## Classification tree:
## tree(formula = Purchase ~ ., data = oj.ds.train)
## Variables actually used in tree construction:
## [1] "LoyalCH"     "SalePriceMM" "SpecialCH"   "PriceDiff"   "PriceCH"
## [6] "StoreID"
## Number of terminal nodes:  9
## Residual mean deviance:  0.7169 = 567.1 / 791
## Misclassification error rate: 0.1538 = 123 / 800
```

Of all the predictors most important indicators appear to be “LoyalCH” “SalePriceMM” “SpecialCH” “PriceDiff” “PriceCH”. Residual mean deviance is 0.7169. There are 9 terminal nodes. Error rate for the model is 0.1538.

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.

```
oj.ds.tree.fit
```

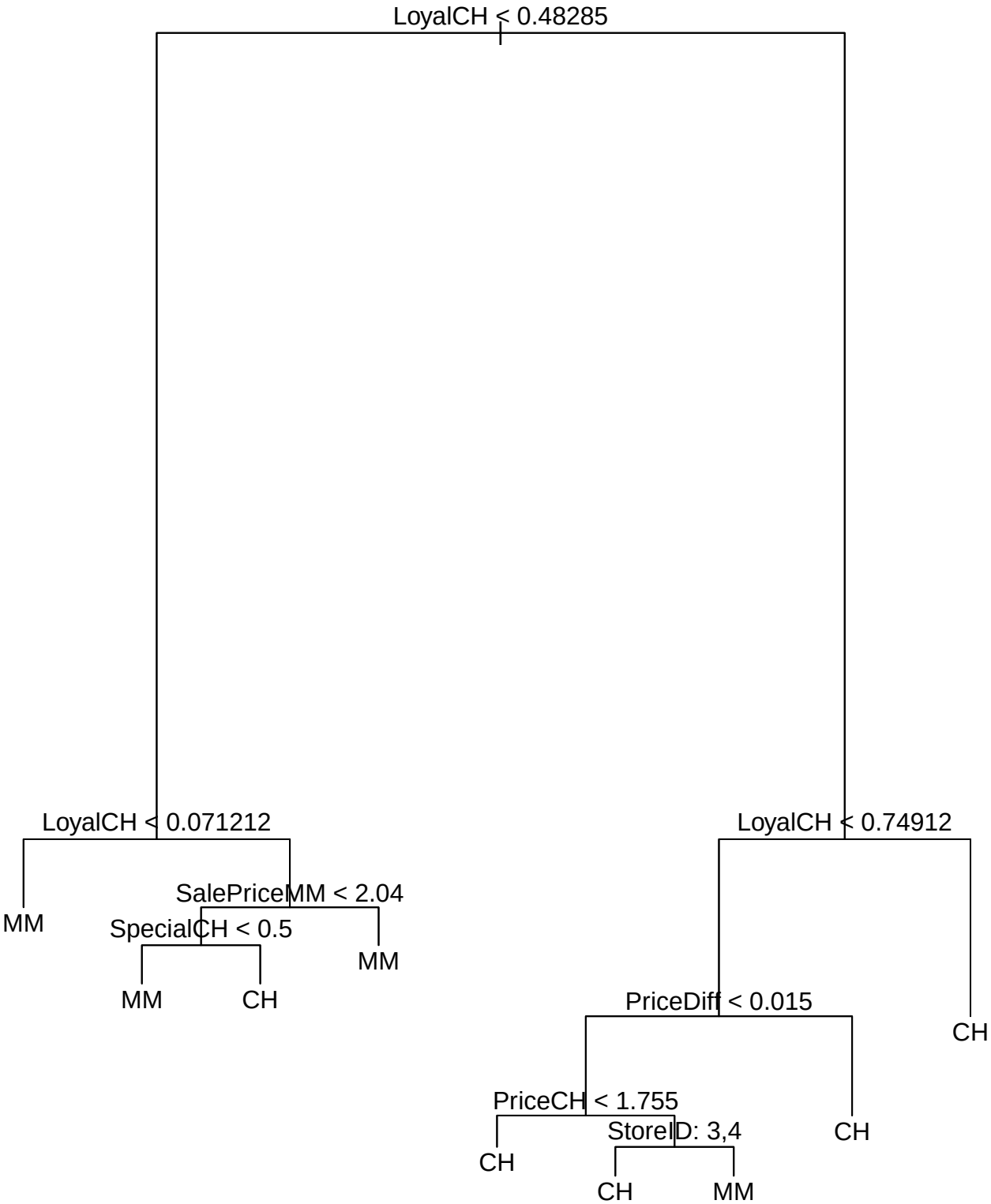
```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 800 1070.00 CH ( 0.61000 0.39000 )
##    2) LoyalCH < 0.48285 300  318.70 MM ( 0.22333 0.77667 )
##      4) LoyalCH < 0.071212 70   18.16 MM ( 0.02857 0.97143 ) *
```

```
##      5) LoyalCH > 0.071212 230  273.90 MM ( 0.28261 0.71739 )
##       10) SalePriceMM < 2.04 124  115.90 MM ( 0.17742 0.82258 )
##         20) SpecialCH < 0.5 105   74.63 MM ( 0.11429 0.88571 ) *
##         21) SpecialCH > 0.5 19   26.29 CH ( 0.52632 0.47368 ) *
##       11) SalePriceMM > 2.04 106  143.20 MM ( 0.40566 0.59434 ) *
##    3) LoyalCH > 0.48285 500  436.30 CH ( 0.84200 0.15800 )
##      6) LoyalCH < 0.74912 231  281.70 CH ( 0.70130 0.29870 )
##       12) PriceDiff < 0.015 75  102.40 MM ( 0.42667 0.57333 )
##         24) PriceCH < 1.755 32   41.18 CH ( 0.65625 0.34375 ) *
##         25) PriceCH > 1.755 43   48.90 MM ( 0.25581 0.74419 )
##           50) StoreID: 3,4 17   23.51 CH ( 0.52941 0.47059 ) *
##           51) StoreID: 1,2,7 26   14.10 MM ( 0.07692 0.92308 ) *
##       13) PriceDiff > 0.015 156  140.60 CH ( 0.83333 0.16667 ) *
##      7) LoyalCH > 0.74912 269   85.47 CH ( 0.96283 0.03717 ) *
```

For 3rd Branch on left : When customer loyalty to Citrus Hill is < 0.48258 and > 0.071212 and Sale Price for Minute maid is less than < 2.04 then the decision tree moves to decide between MM or CH based on SpecialCH if it is < 0.5 then customer is likedy to buy MM else Citurs Hill orange juice

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

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



Tree classification creates binary tree and starts by splitting decision based on LoyalCH if the value is less than 0.48285 then the decision tree moves to left else moves to right node. he tree keeps splitting until it reached the terminal nodes and then the values from the terminal node serves as response based on the predictors.

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?

```
  oj.ds.test.predict <- predict(oj.ds.tree.fit,newdata = oj.ds.test,type="class")
  oj.cm <- table(oj.ds.test.predict,oj.ds.test$Purchase)
  oj.cm
```

```
##
## oj.ds.test.predict  CH  MM
##                 CH 136  29
##                 MM  29  76
```

```
  1 - ( (oj.cm[1,1] + oj.cm[2,2]) / nrow(oj.ds.test))
```

```
## [1] 0.2148148
```

Error rate is 0.2148148

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

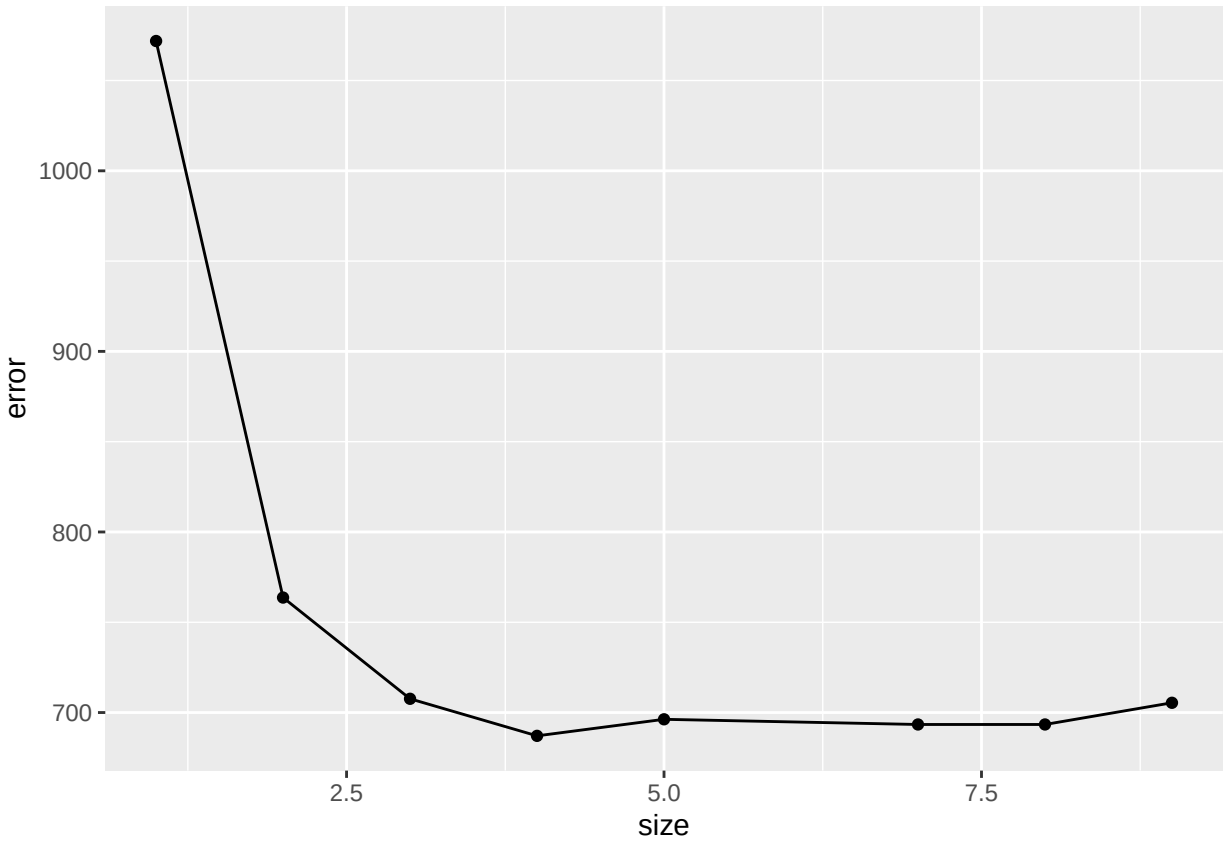
```
oj.ds.train.cv.tree <- cv.tree(oj.ds.tree.fit)
oj.ds.train.cv.tree
```

```
## $size
## [1] 9 8 7 5 4 3 2 1
##
## $dev
## [1]  705.4082  693.3907  693.3907  696.2574  687.0949  707.6130  763.6748
## [8] 1071.8733
##
## $k
## [1]      -Inf  11.29242  12.26703  14.90794  26.60893  38.78107  69.16334
## [8] 315.00169
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

Optimal tree size is 7 with dev of 693.3907

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

```
  ggplot(mapping=aes(oj.ds.train.cv.tree$size, oj.ds.train.cv.tree$dev)) + geom_line() + geom_point() +
```


1000
900
800
700
error
2.5
5.0
7.5
size