Stat 897 Fall 2017 Data Analysis Assignment 11

Penn State

Due November 19, 2017

In this assignment we will use the OJ data found in the ISLR library.

```
library(ISLR)
library (tree)

data("OJ")
str(OJ)
```

```
## 'data.frame':    1070 obs. of  18 variables:
##  $ Purchase      : Factor w/ 2 levels "CH","MM": 1 1 1 2 1 1 1 1 1 1 ...
##  $ WeekofPurchase: num  237 239 245 227 228 230 232 234 235 238 ...
##  $ StoreID       : num  1 1 1 1 7 7 7 7 7 7 ...
##  $ PriceCH       : num  1.75 1.75 1.86 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
##  $ PriceMM       : num  1.99 1.99 2.09 1.69 1.69 1.99 1.99 1.99 1.99 1.99 ...
##  $ DiscCH        : num  0 0 0.17 0 0 0 0 0 0 0 ...
##  $ DiscMM        : num  0 0.3 0 0 0 0 0.4 0.4 0.4 0.4 ...
##  $ SpecialCH     : num  0 0 0 0 0 0 1 1 0 0 ...
##  $ SpecialMM     : num  0 1 0 0 0 1 1 0 0 0 ...
##  $ LoyalCH       : num  0.5 0.6 0.68 0.4 0.957 ...
##  $ SalePriceMM   : num  1.99 1.69 2.09 1.69 1.69 1.99 1.59 1.59 1.59 1.59 ...
##  $ SalePriceCH   : num  1.75 1.75 1.69 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
##  $ PriceDiff     : num  0.24 -0.06 0.4 0 0 0.3 -0.1 -0.16 -0.16 -0.16 ...
##  $ Store7        : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 2 2 2 2 ...
##  $ PctDiscMM     : num  0 0.151 0 0 0 ...
##  $ PctDiscCH     : num  0 0 0.0914 0 0 ...
##  $ ListPriceDiff : num  0.24 0.24 0.23 0 0 0.3 0.3 0.24 0.24 0.24 ...
##  $ STORE         : num  1 1 1 1 0 0 0 0 0 0 ...
```

(a) Create a training set containing a random sample of 800 observations, and a test set containing the remaining observations. This is already done. You can use the same sets from Assignment 10.

```
set.seed(35)
train=sample(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 except for Buy 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?

```
# there is no parameter called Buy
dtree <- tree(Purchase ~ ., data = OJ.train)
summary(dtree)
```

```
##
## Classification tree:
## tree(formula = Purchase ~ ., data = OJ.train)
## Variables actually used in tree construction:
## [1] "LoyalCH"       "PriceDiff"     "SpecialCH"     "ListPriceDiff"
## Number of terminal nodes:  9
## Residual mean deviance:  0.7128 = 563.8 / 791
## Misclassification error rate: 0.14 = 112 / 800
```

The tree has the following characteristics: - 9 terminal nodes - Training error rate of 0.14 (112/800)

(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.

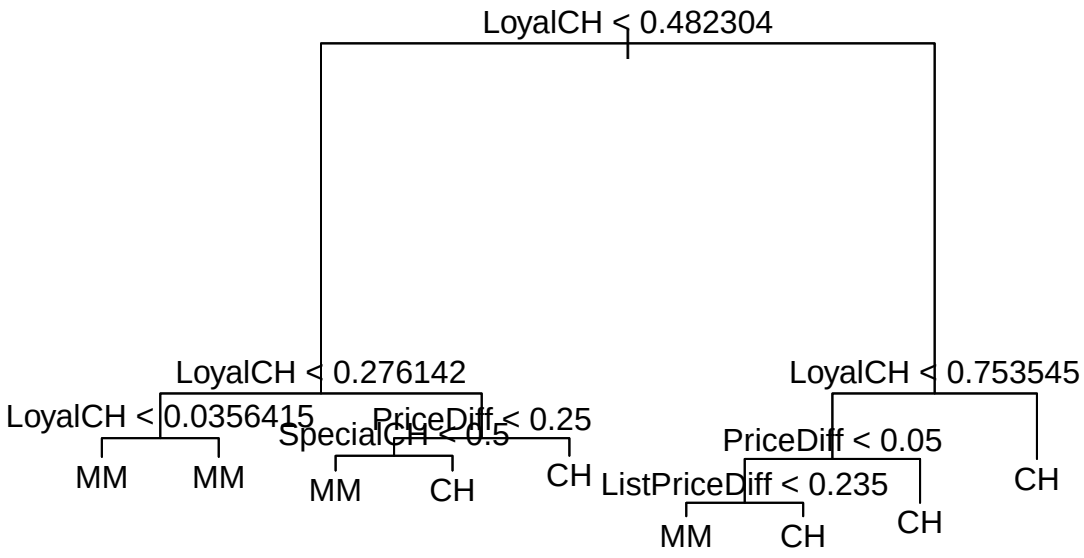
```
dtree
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 800 1066.00 CH ( 0.61500 0.38500 )
##    2) LoyalCH < 0.482304 306  336.20 MM ( 0.23856 0.76144 )
##      4) LoyalCH < 0.276142 172  115.30 MM ( 0.10465 0.89535 )
##        8) LoyalCH < 0.0356415 60    0.00 MM ( 0.00000 1.00000 ) *
##        9) LoyalCH > 0.0356415 112   98.75 MM ( 0.16071 0.83929 ) *
##      5) LoyalCH > 0.276142 134  181.40 MM ( 0.41045 0.58955 )
##       10) PriceDiff < 0.25 87  104.40 MM ( 0.28736 0.71264 )
##         20) SpecialCH < 0.5 77   81.30 MM ( 0.22078 0.77922 ) *
##         21) SpecialCH > 0.5 10   10.01 CH ( 0.80000 0.20000 ) *
##       11) PriceDiff > 0.25 47   61.51 CH ( 0.63830 0.36170 ) *
##    3) LoyalCH > 0.482304 494  420.70 CH ( 0.84818 0.15182 )
##      6) LoyalCH < 0.753545 229  271.30 CH ( 0.72052 0.27948 )
##       12) PriceDiff < 0.05 84  116.30 MM ( 0.47619 0.52381 )
##         24) ListPriceDiff < 0.235 59   76.82 MM ( 0.35593 0.64407 ) *
##         25) ListPriceDiff > 0.235 25   27.55 CH ( 0.76000 0.24000 ) *
##       13) PriceDiff > 0.05 145  116.30 CH ( 0.86207 0.13793 ) *
##      7) LoyalCH > 0.753545 265   91.54 CH ( 0.95849 0.04151 ) *
```

Node label: 20, terminal node, split: SpecialCH < 0.5, Number of observations: 77
Deviance: 81.30 Prediction: MM (22% have CH while the remaining almost 78% have MM)

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

```
plot(dtree)
text(dtree, pretty = 0)
```



The most important predictor is LoyalCH. The other lesser important predictors are: "PriceDiff" "SpecialCH" "ListPriceDiff"

LoyalCH is the most important and we see that most scenarios for LoyalCH < 0.482304 the prediction is MM whereas for most scenarios for LoyalCH $>= 0.482304$ the prediction is CH.

Predictor ListPriceDiff is important for the case where $0.482304 <= \text{LoyalCH} < 0.753545$ On the other hand SpecialCH is important when $0.276142 < \text{LoyalCH} < 0.482304$

(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?

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

```
##
## dtree.pred  CH  MM
##         CH 143  36
##         MM  18  73
```

```
1-(143+73)/(143+73+18+36)
```

```
## [1] 0.2
```

Test error rate: 20%

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

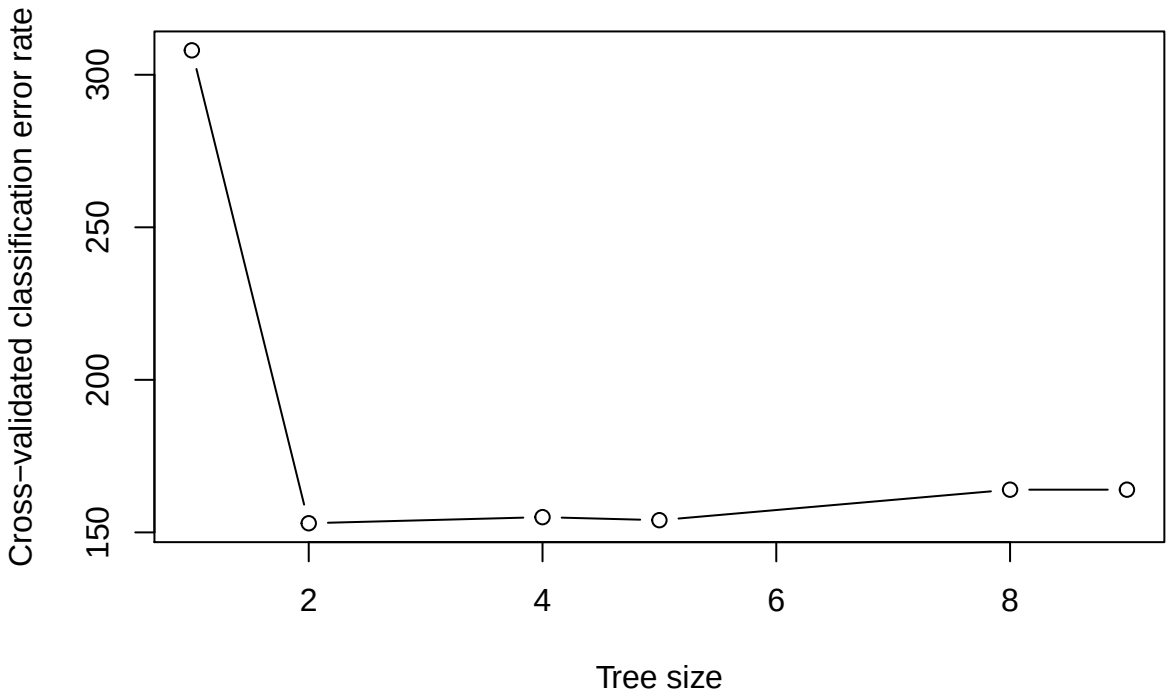
```
set.seed(1)
cv.dtree <- cv.tree(dtree, FUN = prune.misclass)
cv.dtree
```

```
## $size
## [1] 9 8 5 4 2 1
##
## $dev
## [1] 164 164 154 155 153 308
##
## $k
## [1]       -Inf   0.000000   5.666667   6.000000   6.500000 160.000000
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

The optimal tree size = 2

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

```
plot(cv.dtree$size, cv.dtree$dev, type = "b", xlab = "Tree size", ylab = "Cross-validated classificatio
```

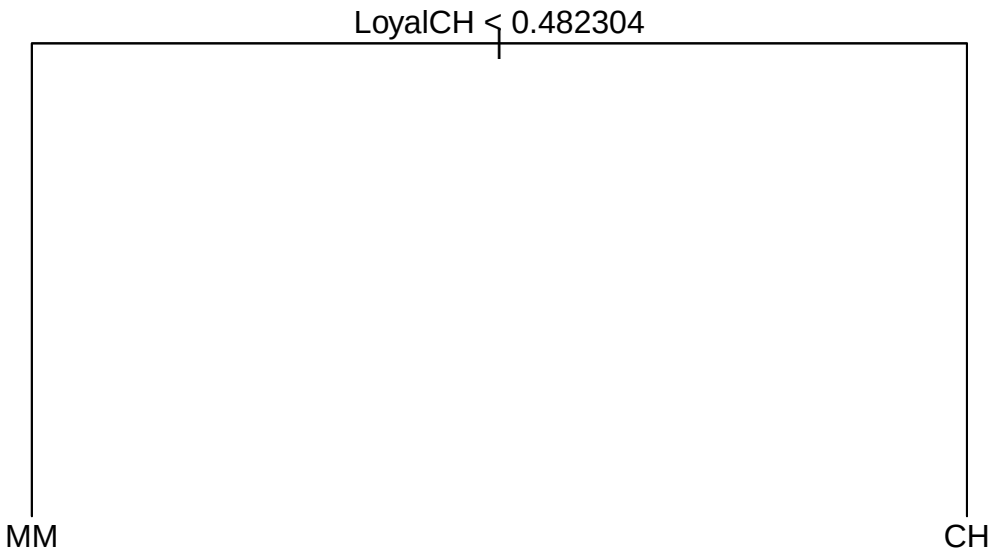


(h) Which tree size corresponds to the lowest cross-validated classification error rate?

Tree size of 2 corresponds to the lowest cross-validated 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 create a pruned tree with five terminal nodes.

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



Plot also indicates clearly that tree size of 2 corresponds to the lowest cross-validated classification error rate.

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

```
summary(dtree)
```

```
##
## Classification tree:
## tree(formula = Purchase ~ ., data = OJ.train)
## Variables actually used in tree construction:
## [1] "LoyalCH"       "PriceDiff"     "SpecialCH"     "ListPriceDiff"
## Number of terminal nodes:  9
## Residual mean deviance:  0.7128 = 563.8 / 791
## Misclassification error rate: 0.14 = 112 / 800
```

```
summary(prune.dtree)
```

```
##
## Classification tree:
## snip.tree(tree = dtree, nodes = c(3L, 2L))
## Variables actually used in tree construction:
## [1] "LoyalCH"
## Number of terminal nodes:  2
## Residual mean deviance:  0.9486 = 757 / 798
## Misclassification error rate: 0.185 = 148 / 800
```

For the training data we find that error rate is slightly higher for the pruned tree ($0.185 > 0.14$)

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

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

```
##
## prune.dtree.pred  CH  MM
##               CH 140  35
##               MM  21  74
```

```
1-(140+74)/(140+74+21+35)
```

```
## [1] 0.2074074
```

The test error rate = 20.74%

With full tree we had got test error rate of 20% and with pruned tree we get a very slight increase of 0.74%. The interpretability of the simple 2 node tree is much greater than the full tree.