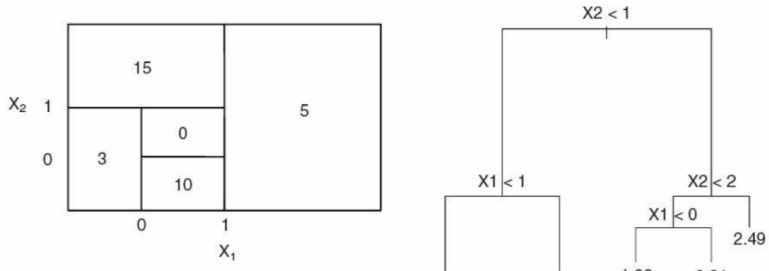
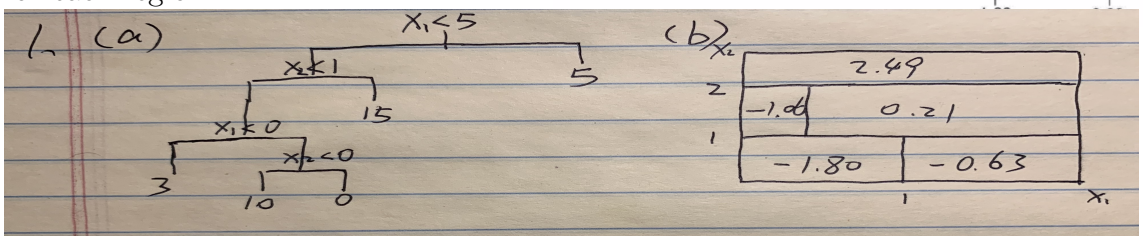
# Homework #10: Trees (Chap. 8)

1. **(Trees; Chap. 8, # 4, p. 332)** Look at the figure below.

   (a) Sketch the tree corresponding to the partition of the predictor space in the left-hand panel of the figure. The numbers inside the boxes indicate the mean of Y within each region.

   (b) Look at the tree in the right-hand panel of the same figure. Based on this tree, create a diagram similar to the left-hand panel of this figure. Divide the predictor space into the correct regions, and indicate the mean for each region.




2. **(Bagging; Chap. 8, # 5, p. 332)** Suppose we produce ten bootstrapped samples from a data set containing red and green classes. We then apply a classification tree to each bootstrapped sample and, for a specific value of $X$, produce 10 estimates of $\boldsymbol{P}\{\text{Class is Red}\,|X\}$ : 0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, and 0.75.

   There are two common ways to combine these results together into a single class prediction. One is the majority vote approach discussed in this chapter. The second approach is to classify based on the average probability. In this example, what is the final classification under each of these two approaches?

```
p <- c(0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75)
  # majority vote approach
summary(p>0.5)

##    Mode   FALSE    TRUE
## logical       4       6

  # average probability
mean(p)

## [1] 0.45
```

   Under the majority vote approach, the classification would be red. Under the average probability approach, the classification would be green.

3. **(Project; Chap. 8, # 9, p. 334)**
   This problem involves the ***OJ*** (orange juice) data set which is part of the `ISLR` package. `library(ISLR)`; `attach(OJ)`; To find its description, type `?OJ`

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

```
set.seed(123)
n <- nrow(OJ)
train <- sample(n, 800)
```

(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?

```
class.tree <- tree(Purchase ~ ., data = OJ[train,])
summary(class.tree)

##
## Classification tree:
## tree(formula = Purchase ~ ., data = OJ[train, ])
## Variables actually used in tree construction:
## [1] "LoyalCH"   "PriceDiff" "SpecialCH" "PctDiscMM"
## Number of terminal nodes:  10
## Residual mean deviance:  0.7289 = 575.8 / 790
## Misclassification error rate: 0.1612 = 129 / 800
```

The training error rate is 0.1612 and the number of terminal nodes is 10.

(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.

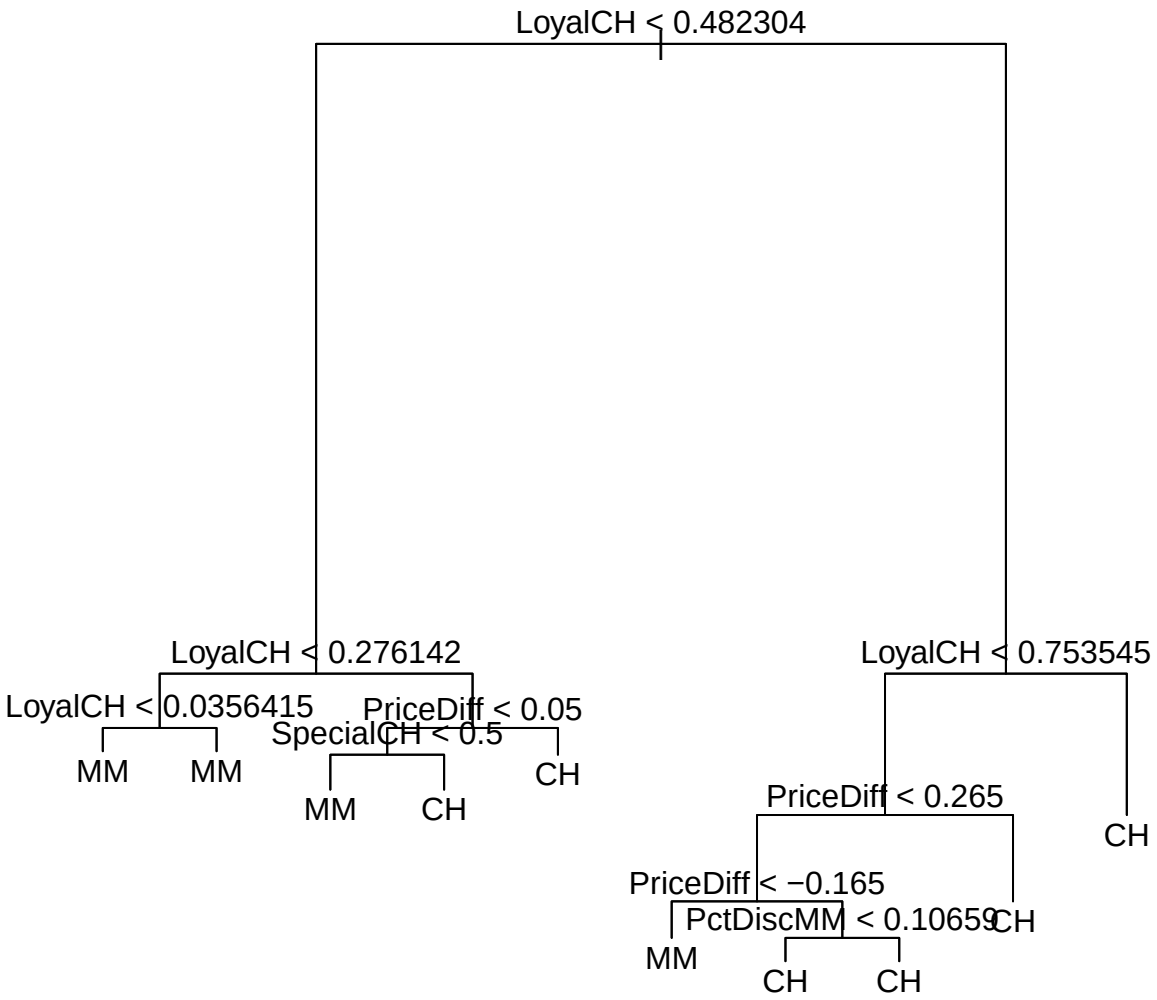
```
class.tree

## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 800 1073.000 CH ( 0.60500 0.39500 )
##    2) LoyalCH < 0.482304 299  320.600 MM ( 0.22742 0.77258 )
##      4) LoyalCH < 0.276142 172  127.600 MM ( 0.12209 0.87791 )
##        8) LoyalCH < 0.0356415 56   10.030 MM ( 0.01786 0.98214 ) *
##        9) LoyalCH > 0.0356415 116  106.600 MM ( 0.17241 0.82759 ) *
##      5) LoyalCH > 0.276142 127  167.400 MM ( 0.37008 0.62992 )
##       10) PriceDiff < 0.05 58   59.140 MM ( 0.20690 0.79310 )
##         20) SpecialCH < 0.5 51   36.950 MM ( 0.11765 0.88235 ) *
##         21) SpecialCH > 0.5 7    5.742 CH ( 0.85714 0.14286 ) *
##       11) PriceDiff > 0.05 69   95.640 CH ( 0.50725 0.49275 ) *
##    3) LoyalCH > 0.482304 501  456.300 CH ( 0.83034 0.16966 )
##      6) LoyalCH < 0.753545 236  292.000 CH ( 0.69068 0.30932 )
##       12) PriceDiff < 0.265 147  202.300 CH ( 0.55102 0.44898 )
##         24) PriceDiff < -0.165 40   47.050 MM ( 0.27500 0.72500 ) *
##         25) PriceDiff > -0.165 107  138.000 CH ( 0.65421 0.34579 )
##           50) PctDiscMM < 0.10659 75  102.900 CH ( 0.56000 0.44000 ) *
##           51) PctDiscMM > 0.10659 32   24.110 CH ( 0.87500 0.12500 ) *
##       13) PriceDiff > 0.265 89   49.030 CH ( 0.92135 0.07865 ) *
```

```
##      7) LoyalCH > 0.753545 265   97.720 CH ( 0.95472 0.04528 ) *
```

For node 8), if Customer brand loyalty for CH is less than 0.0356415, the observation will be classified to MM. The sample size of this box is 56, and the deviance for this node is 10.030. Also in this box, the proportion of MM is 0.98214, and that of CH is 0.01786.

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

```
plot(class.tree)
text(class.tree)
```




(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?

```
prediction <- predict(class.tree, newdata = OJ[-train,], type = "class")
table(prediction,OJ$Purchase[-train])

##
## prediction  CH  MM
##         CH 158  37
##         MM  11  64

mean(prediction != OJ$Purchase[-train]) # test error rate

## [1] 0.1777778
```

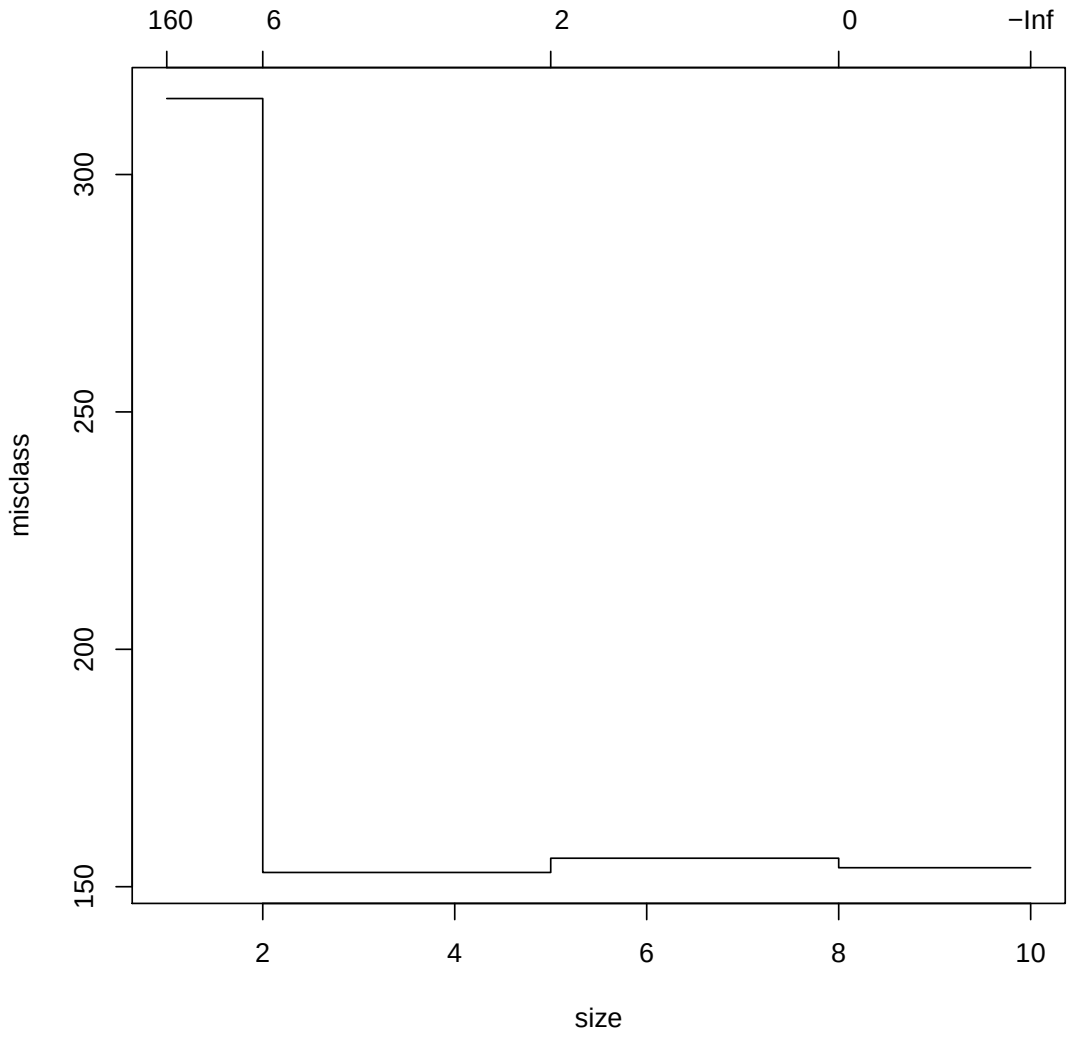
The test error rate is 0.1777778.

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

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

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

```
plot(cv)
```



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

```
cv$size[which.min(cv$dev)]

## [1] 2
```

The lowest cross-validated classification error rate is 2.

(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.

```
tree.pruned <- prune.misclass( class.tree, best= cv$size[which.min(cv$dev)] )
```

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

```
train.pred <- predict(class.tree, newdata = OJ[train,], type = "class")
mean(train.pred != OJ$Purchase[train]) # train error rate

## [1] 0.16125

train.pred.pruned <- predict(tree.pruned, newdata = OJ[train,], type = "class")
mean(train.pred.pruned != OJ$Purchase[train]) # train error rate. pruned

## [1] 0.19125
```

The training error rate of the pruned trees is higher.

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

```
prediction.pruned <- predict(tree.pruned, newdata = OJ[-train,], type = "class")
mean(prediction.pruned != OJ$Purchase[-train])

## [1] 0.1888889
```

The test error rate of the pruned tree is $0.1888889 > 0.1777778$, the test error rate of pruned tree.