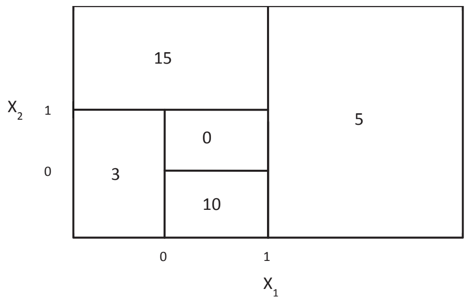
Homework 5 - MATH 4322

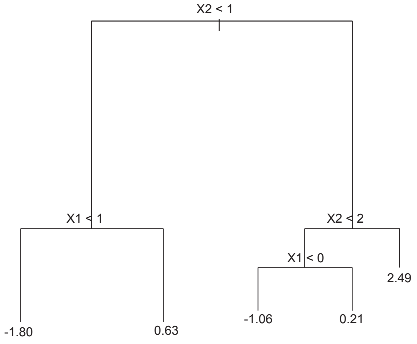
Instructions

1. Due date: April 4, 2024
2. Answer the questions fully for full credit.
3. Scan or Type your answers and submit only one file. (If you submit several files only the recent one uploaded will be graded).
4. Preferably save your file as PDF before uploading.
5. Submit in Canvas.
6. These questions are from *An Introduction to Statistical Learning with Applications* in R by James, et. al., chapter 8.

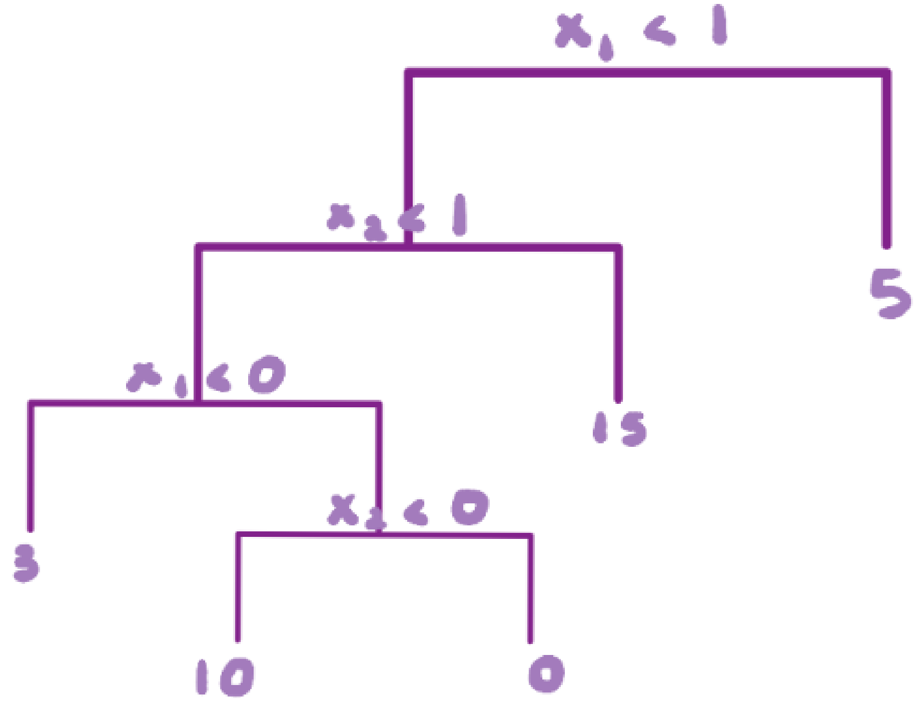
Problem 1

The questions relate to the following plots:

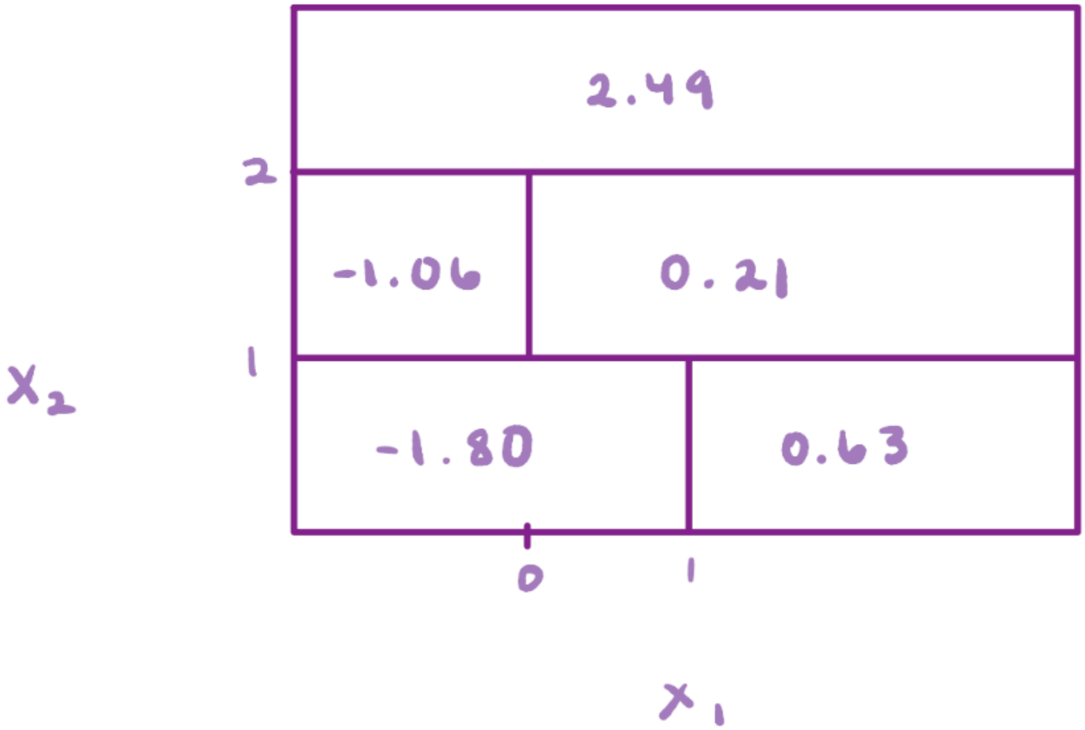




a) Sketch the tree corresponding to the partition of the predictor space illustrated on the left-hand plot. The numbers inside the boxes indicate the mean of Y within each region.



b) Create a diagram similar to the left-hand plot using the tree illustrated in the right-hand plot. You should divide up the predictor space into the correct regions, and indicate the mean for each region.



Problem 2

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 $P(\text{Class is Red}|X)$:

$$0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, \text{ and } 0.75.$$

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?

Answer: Majority Vote Approach
We look at the estimates of P(Class is Red|X), where any value less than 0.5 is Green and any value above or equal to 0.5 is Red.
Values: {0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75}
Results: {Green, Green, Green, Green, Red, Red, Red, Red, Red, Red} According to Majority Vote, since most of the values are red, we will classify these values as Red.

Answer: Average Probability Approach
We look at the estimates of P(Class is Red|X), where any value less than 0.5 is Green and any value above or equal to 0.5 is Red.

```r
values = c(0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75)
mean(values)
```

```
[1] 0.45
```

The mean of these values is 0.45, which means that we will classify these values as Green.

Problem 3

Provide a detailed explanation of the algorithm that is used to fit a regression tree.
Answer
Steps to building a decision tree:
1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations

2. Apply cost complexity pruning to the larger tree in order to obtain a sequence of best subtrees, as a function of α

3. Use K-fold cross validation to choose α (divide the training observations into K folds). For each k = 1, ... , K:

   a) Repeat steps 1 and 2 on all but the kth fold of the training data

   b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α

4. Average the results for each value of α and pick α to minimize the average

5. Return the subtree from Step 2 that corresponds to the chosen value of α

Problem 4

We will use the `Carseats` data set in the `ISLR2` package to predict `Sales`.

a) Treating `Sales` as a quantitative variable, would we use create a regression tree or classification tree?
Regression tree

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

```
library(ISLR2)
```

```
Warning: package 'ISLR2' was built under R version 4.2.3
```

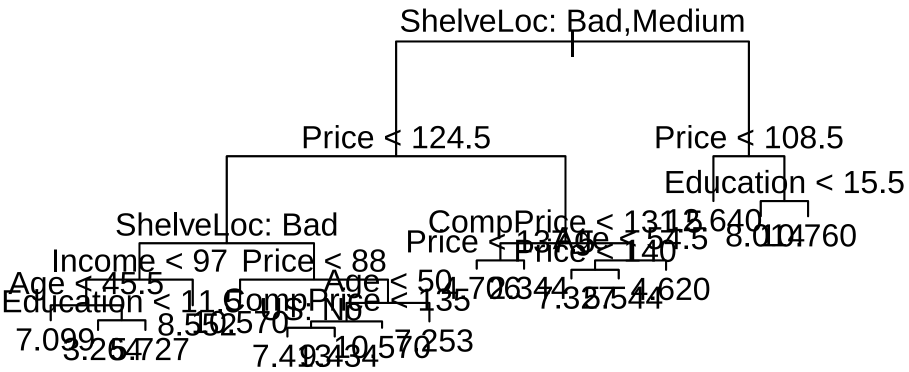
```
library(tree)
```

```
Warning: package 'tree' was built under R version 4.2.3
```

```
attach(Carseats)
set.seed(106)
car.sample = sample(1:nrow(Carseats), nrow(Carseats)/2)
car.train = Carseats[car.sample, ]
car.test = Carseats[-car.sample, ]
```

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

```
car.tree <- tree(Sales ~ ., data = car.train)
plot(car.tree)
text(car.tree, pretty = 0)
```



```
summary(car.tree)
```

```
Regression tree:
tree(formula = Sales ~ ., data = car.train)
Variables actually used in tree construction:
[1] "ShelveLoc" "Price"     "Income"    "Age"       "Education" "CompPrice"
[7] "US"
Number of terminal nodes:  17
Residual mean deviance:  2.157 = 394.7 / 183
Distribution of residuals:
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-5.17400 -0.87900  0.06129  0.00000  0.89830  4.27600
```

Signifcant variables: ShelveLoc, Price, Income, Age, Education, CompPrice
Number of terminal nodes: 17
Residual mean deviance: 2.157

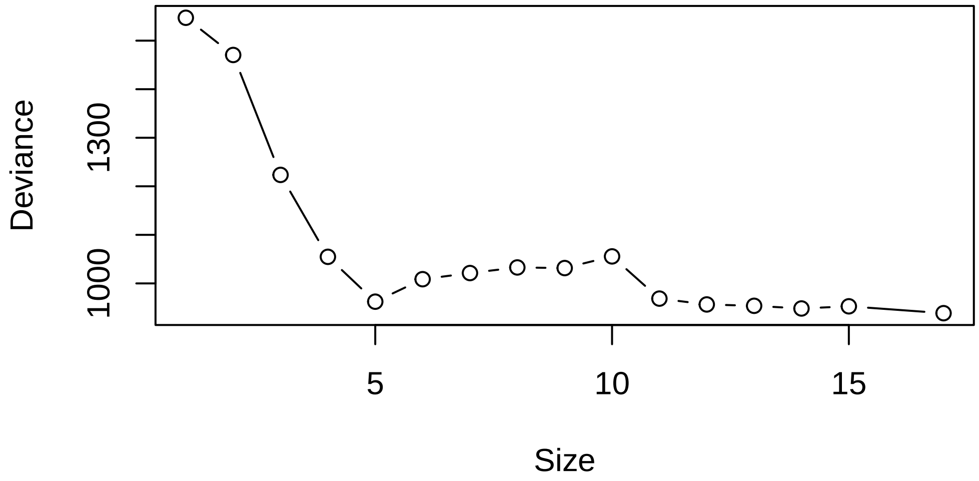
```
predict.car <- predict(car.tree, newdata = car.test)
mean((predict.car - car.test$Sales)^2)
```

```
[1] 4.497297
```

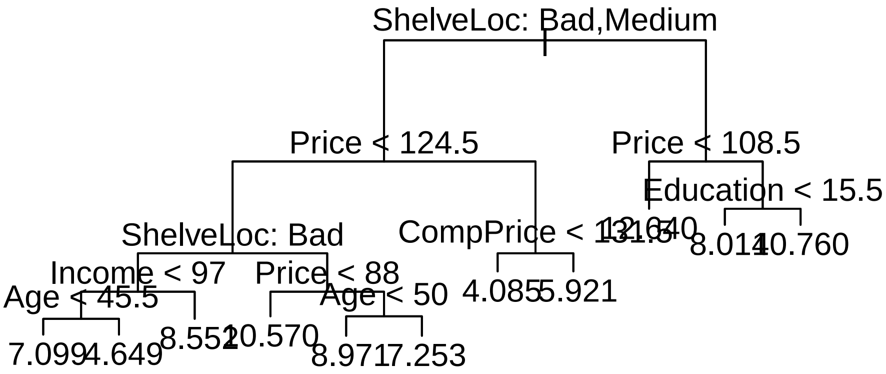
MSE = 4.497297

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

```
set.seed(106)
cv.car <- cv.tree(car.tree)
plot(cv.car$size, cv.car$dev, type = "b", xlab = "Size", ylab = "Deviance")
```



```
prune.car <- prune.tree(car.tree, best = 11)
plot(prune.car)
text(prune.car, pretty = 0)
```



```r
predict.prune.car <- predict(prune.car, newdata = car.test)
mean((predict.prune.car - car.test$Sales)^2)
```

```
[1] 4.737702
```

The MSE is 4.737702, which is slightly higher than the unpruned tree.

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

```r
library(randomForest)
```

```
Warning: package 'randomForest' was built under R version 4.2.3
```

```
randomForest 4.7-1.1
```

```
Type rfNews() to see new features/changes/bug fixes.
```

```
set.seed(106)
bag.car <- randomForest(Sales ~ ., data = car.train, mtry = 10,
ntree = 500, importance = TRUE)
predict.bag.car <- predict(bag.car, newdata = car.test)
mean((predict.bag.car - car.test$Sales)^2)
```

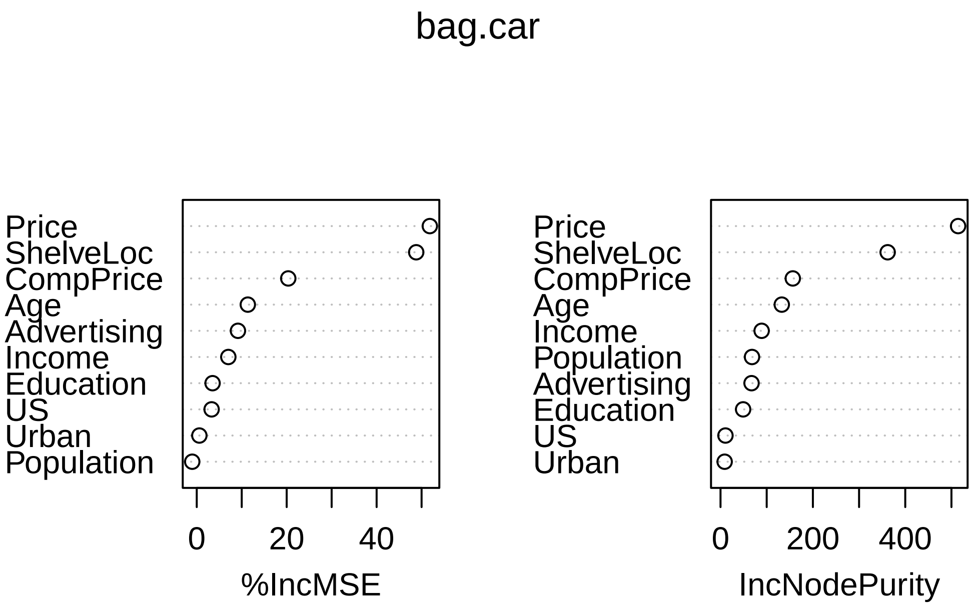
```
[1] 2.442005
```

MSE = 2.442005

```
importance(bag.car)
```

```
               %IncMSE IncNodePurity
CompPrice   20.3500606    156.480083
Income       7.0263844     89.063502
Advertising  9.1629685     67.185291
Population  -1.0008739     68.169340
Price       51.8301078    514.450076
ShelveLoc   48.8048976    361.934489
Age         11.3577292    132.610521
Education    3.5172958     48.917736
Urban        0.5832855      8.947254
US           3.3069884     10.675551
```

```
varImpPlot(bag.car)
```



Our two most important variables are Price and ShelveLoc.

f) 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.

```
library(randomForest)
p = ncol(Carseats) - 1
rf.car <- randomForest(Sales ~ ., data = car.train, mtry = (p/3), ntree = 500,

importance = TRUE)

rf.car
```

```
Call:
 randomForest(formula = Sales ~ ., data = car.train, mtry = (p/3),      ntree = 500, importa
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 3

          Mean of squared residuals: 3.421573
```

```
                    % Var explained: 54.43
```

```
predict.rf <- predict(rf.car, newdata = car.test)
mean((predict.rf - car.test$Sales)^2)
```

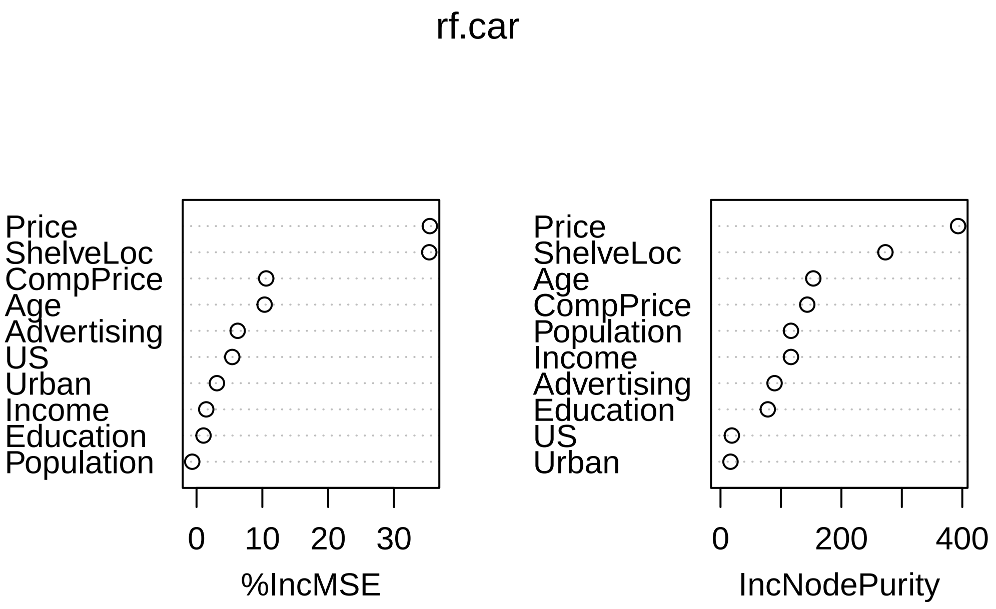
```
[1] 3.099648
```

Each time a tree split is considered, it picks a random subset of m $\approx \frac{p}{3}$ predictors from the full set of p predictors.
Number of variables at each split: 3
MSE = 3.16252

```
importance(rf.car)
```

```
               %IncMSE IncNodePurity
CompPrice   10.5845501     143.50174
Income       1.4746884     116.52794
Advertising  6.2482865      89.43406
Population  -0.6531651     116.53540
Price       35.4451214     393.08066
ShelveLoc   35.3599383     272.77684
Age         10.3401907     153.52630
Education    1.0550229      78.22030
Urban        3.0954920      16.52591
US           5.4261335      18.73908
```

```
varImpPlot(rf.car)
```



The two most important variables are Price and ShelveLoc.

Problem 5

This problem involves the `OJ` data set which is part of the `ISLR2` package.

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

```
library(ISLR2)
set.seed(106)
sample = sample(1:nrow(OJ), 800)
train = OJ[sample, ]
test = OJ[-sample, ]
```

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?
`install.packages("tree")`

Answer

```
library(tree)
attach(OJ)
tree.OJ <- tree(Purchase ~ ., data = train)
summary(tree.OJ)
```

```
Classification tree:
tree(formula = Purchase ~ ., data = train)
Variables actually used in tree construction:
[1] "LoyalCH"       "PriceDiff"     "ListPriceDiff"
Number of terminal nodes:  7
Residual mean deviance:  0.7645 = 606.2 / 793
Misclassification error rate: 0.1488 = 119 / 800
```

The training error rate is 14.88%
The 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.

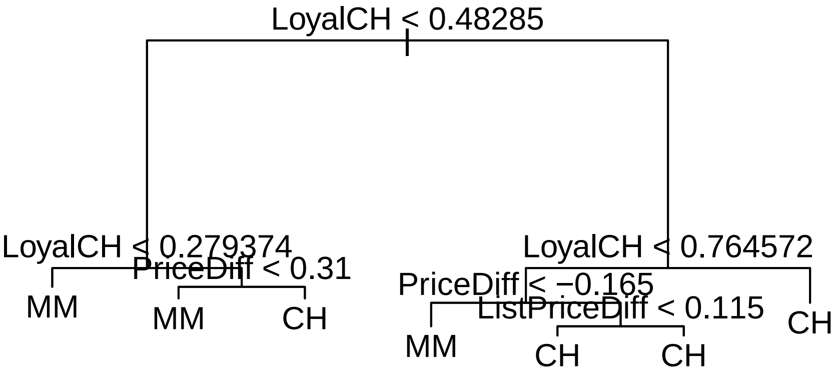
```
tree.OJ
```

```
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 800 1072.00 CH ( 0.60750 0.39250 )
   2) LoyalCH < 0.48285 303  317.60 MM ( 0.21782 0.78218 )
     4) LoyalCH < 0.279374 167  114.20 MM ( 0.10778 0.89222 ) *
     5) LoyalCH > 0.279374 136  176.60 MM ( 0.35294 0.64706 )
      10) PriceDiff < 0.31 102  115.80 MM ( 0.25490 0.74510 ) *
      11) PriceDiff > 0.31 34   44.15 CH ( 0.64706 0.35294 ) *
   3) LoyalCH > 0.48285 497  428.60 CH ( 0.84507 0.15493 )
     6) LoyalCH < 0.764572 243  282.20 CH ( 0.73251 0.26749 )
      12) PriceDiff < -0.165 32   38.02 MM ( 0.28125 0.71875 ) *
      13) PriceDiff > -0.165 211  210.60 CH ( 0.80095 0.19905 )
        26) ListPriceDiff < 0.115 30   41.46 CH ( 0.53333 0.46667 ) *
        27) ListPriceDiff > 0.115 181  155.90 CH ( 0.84530 0.15470 ) *
     7) LoyalCH > 0.764572 254   96.68 CH ( 0.95276 0.04724 ) *
```

Looking at terminal node 12):
Split criterion: PriceDif < -0.165
Number of observations in the branch: 32
Deviance: 38.02
Overall prediction: "MM" (by 71.875%)
Percentage of observations taking "CH": 28.125%

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

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



The most important indicator for Purchase is LoyalCH, as that is the first branch in the tree. The first branch is determined on whether the LoyalCH is lower than 0.48285 or not. It remains important later in the tree, as the next two branches are based of of LoyalCH as well.

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?

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

```
predict.tree  CH  MM
          CH 142  28
          MM  25  75
```

Error Rate Formula: $\frac{misclassified}{total}$

```
test.error.rate = (28 + 25)/(142 + 28 + 25 + 75)
test.error.rate
```

```
[1] 0.1962963
```

The test error rate is about 19.63%.

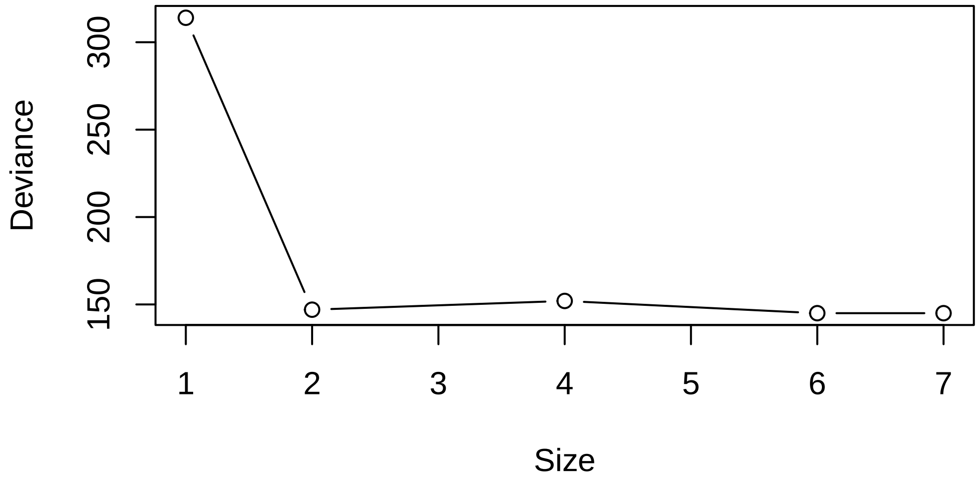
f) Apply the `cv.tree()` function to the training set in order to determine the optimal tree size.

```
set.seed(106)
cv.OJ <- cv.tree(tree.OJ, FUN = prune.misclass)
cv.OJ
```

```
$size
[1] 7 6 4 2 1

$dev
[1] 145 145 152 147 314

$k
[1] -Inf    0    5    7  171

$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.

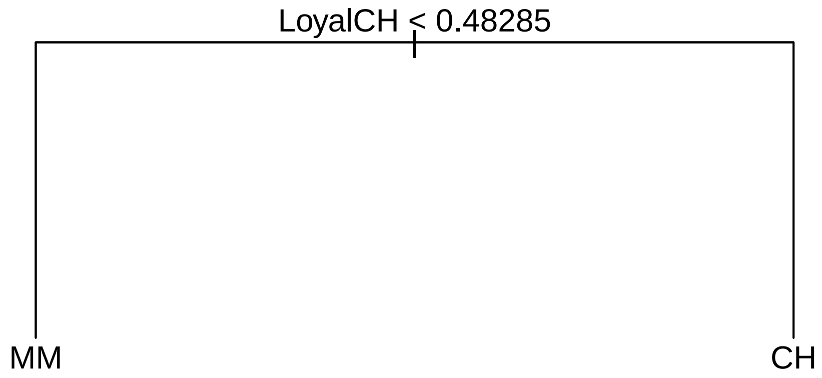
```
plot(cv.OJ$size, cv.OJ$dev, type = "b", xlab = "Size", ylab = "Deviance")
```



h) Which tree size corresponds to the lowest cross-validated classification error rate?
Answer
The tree size that corresponds to the lowest cross-validated classifcation error rate is 6

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, pretty = 0)
```



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

```
summary(tree.OJ)
```

```

Classification tree:
tree(formula = Purchase ~ ., data = train)
Variables actually used in tree construction:
[1] "LoyalCH"       "PriceDiff"     "ListPriceDiff"
Number of terminal nodes:  7 
Residual mean deviance:  0.7645 = 606.2 / 793 
Misclassification error rate: 0.1488 = 119 / 800 
```

```
summary(prune.OJ)
```

```

Classification tree:
snip.tree(tree = tree.OJ, nodes = 2:3)
Variables actually used in tree construction:
[1] "LoyalCH"
```

```
Number of terminal nodes:  2
Residual mean deviance:  0.9351 = 746.2 / 798
Misclassification error rate: 0.1788 = 143 / 800
```

The training error rate of the pruned tree is 17.88% while the training error rate of the unpruned tree is 14.88%. Therefore, the pruned tree is higher.

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

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

```
predict.prune  CH  MM
           CH 139  33
           MM  28  70
```

```
prune.test.error.rate = (33 + 28)/ (139 + 33 + 28 + 70)
prune.test.error.rate
```

```
[1] 0.2259259
```

The test error rate of the pruned tree is about 22.59%, which is higher than the test error rate of the unpruned tree.