HW5_Tree_Shuting_Li

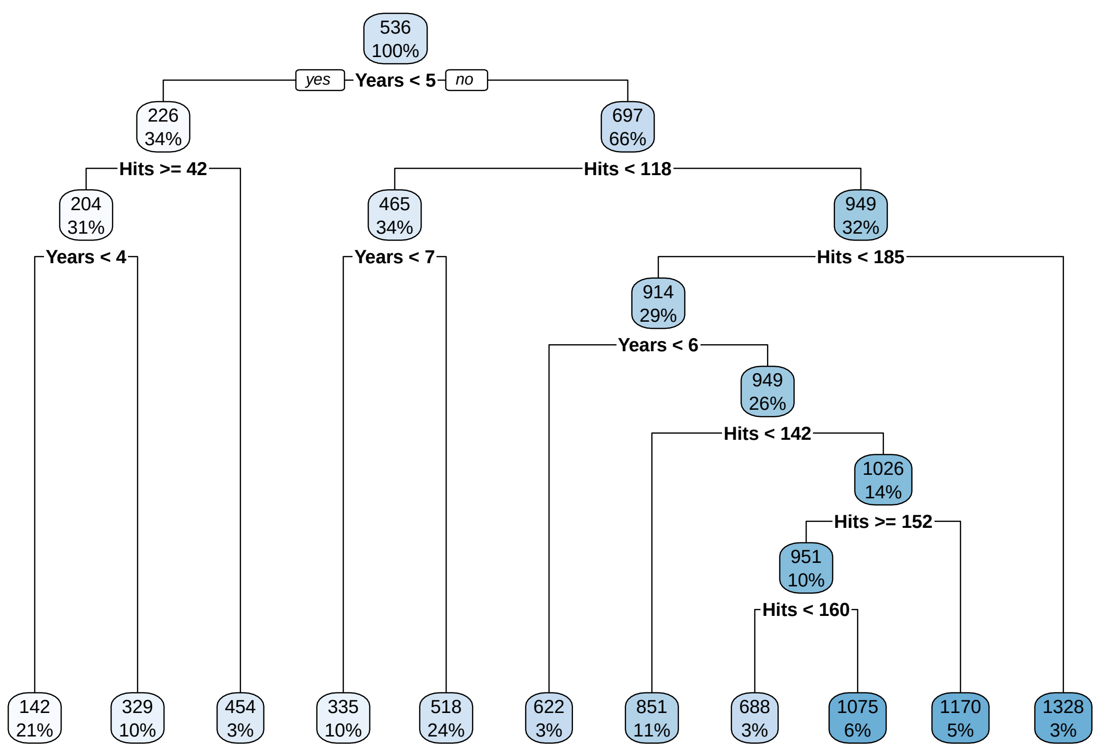
Shuting

3/4/2022

8.1

Draw an example (of your own invention) of a partition of two- dimensional feature space that could result from recursive binary splitting. Your example should contain at least six regions. Draw a decision tree corresponding to this partition. Be sure to label all aspects of your figures, including the regions R1, R2, . . ., the cutpoints t1,t2,…, and so forth.

```
attach(Hitters)
library(rpart)
library(rpart.plot)
fit <- rpart(Salary~Years+Hits, data=Hitters, method='anova')
rpart.plot(fit)
```

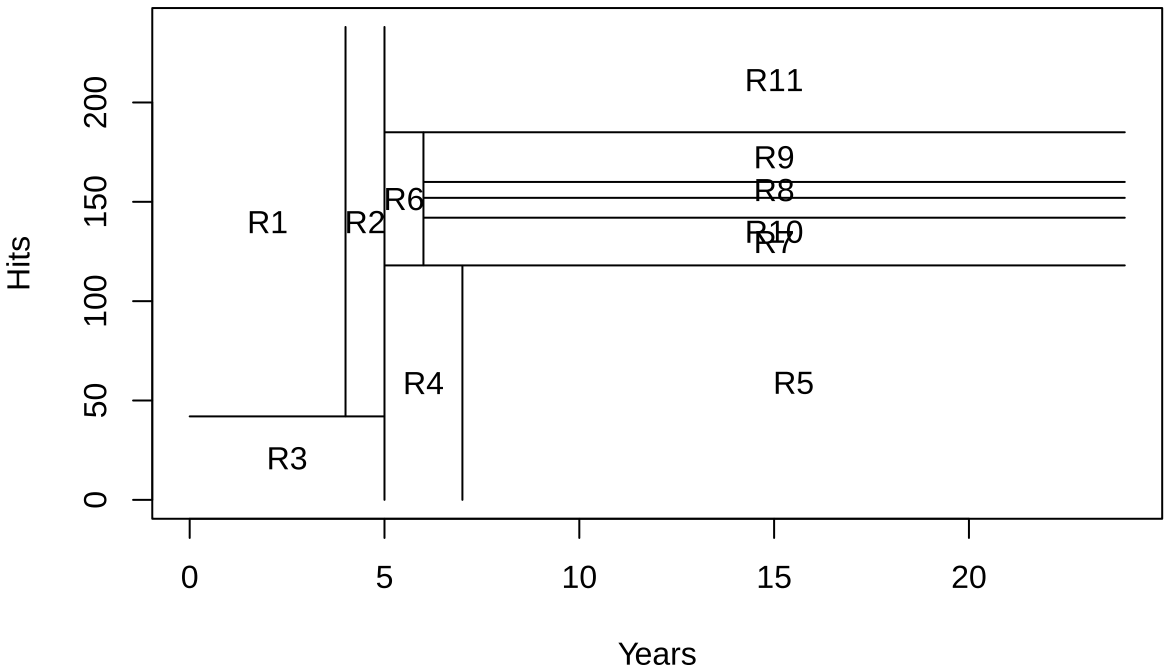


Answer:

```
######
#par(xpd = NA)
plot(NA, NA, type = "n", xlim = c(0, 24), ylim = c(0, 238), xlab = "Years", ylab = "Hits")
# t1: x = 5
lines(x = c(5, 5), y = c(0, 238))
```

```r
# t2: y = 42
lines(x = c(0, 5), y = c(42, 42))
# t3: x = 4
lines(x = c(4, 4), y = c(42, 238))
# t4: y = 118
lines(x = c(5, 24), y = c(118, 118))
# t5: x = 7
lines(x = c(7, 7), y = c(0, 118))
# t6: y = 185
lines(x = c(5, 24), y = c(185, 185))
# t7: x = 6
lines(x = c(6, 6), y = c(118, 185))
# t8: y = 142
lines(x = c(6, 24), y = c(142, 142))
# t9: y = 152
lines(x = c(6, 24), y = c(152, 152))
# t8: y = 160
lines(x = c(6, 24), y = c(160, 160))

text(x = 2, y = (42 + 238)/2, labels = c("R1"))
text(x = 4.5, y = (42 + 238)/2, labels = c("R2"))
text(x = (0 + 5)/2, y = (42 + 0)/2, labels = c("R3"))
text(x = (5 + 7)/2, y = 118/2, labels = c("R4"))
text(x = (24 + 7)/2, y = 118/2, labels = c("R5"))
text(x = (5 + 6)/2, y = (118+185)/2, labels = c("R6"))
text(x = (24 + 6)/2, y = (118+142)/2, labels = c("R7"))
text(x = (24 + 6)/2, y = (152+160)/2, labels = c("R8"))
text(x = (24 + 6)/2, y = (160+185)/2, labels = c("R9"))
text(x = (24 + 6)/2, y = (118+152)/2, labels = c("R10"))
text(x = (24 + 6)/2, y = (185+238)/2, labels = c("R11"))
```

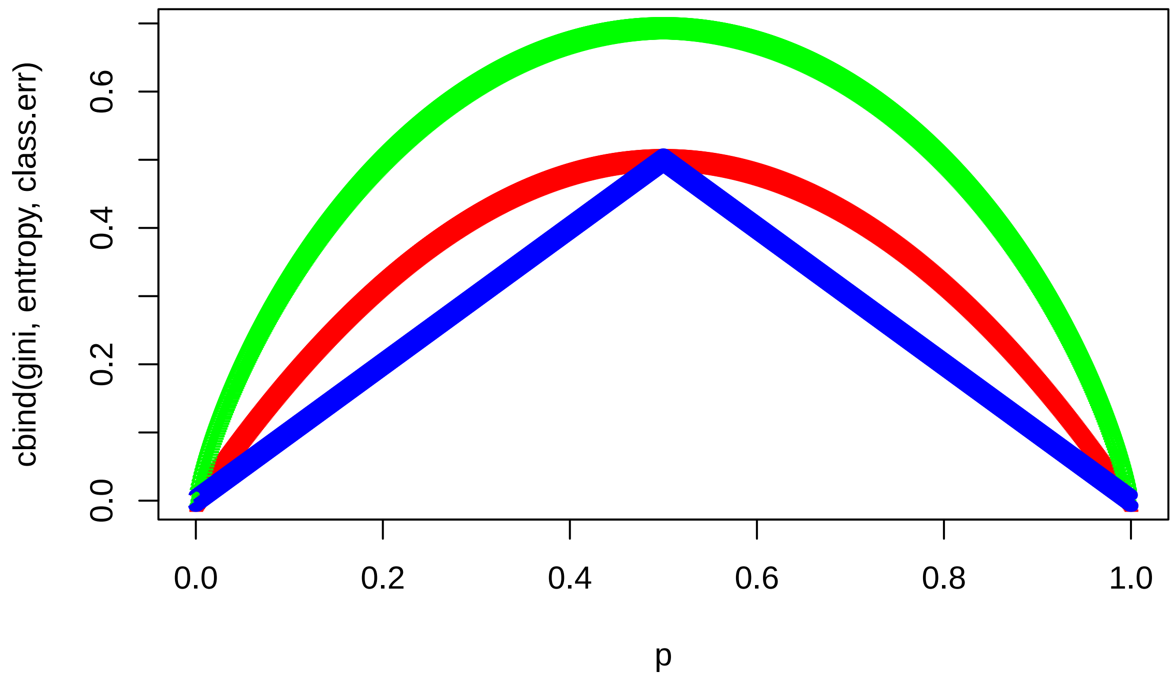


8.2

Answer: For the first stump$f_1(X)$, we denote it as $f_1(X) = \beta_1 * I(X_1 < t_1) + \beta_0$, Then calculate residual $r_1 = Y - \lambda f_1(X)$ and fit the second decision $f_2(X)$. Then model is $f(X) = f_1(X) + f_2(X)$. Repeat the stage for p times and get the final model $f(X) = f_1(X) + f_2(X) + ... + f_p(X)$.

8.3

```
p = seq(0, 1, 0.001)
gini = p * (1 - p) * 2
entropy = -(p * log(p) + (1 - p) * log(1 - p))
class.err = 1 - pmax(p, 1 - p)
matplot(p, cbind(gini, entropy, class.err), col = c("red", "green", "blue"))
```



Answer:

8.5

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

Answer:

```
## [1] 0.45
```

For majority approach, the final class is red. For average approach, the final class is green.

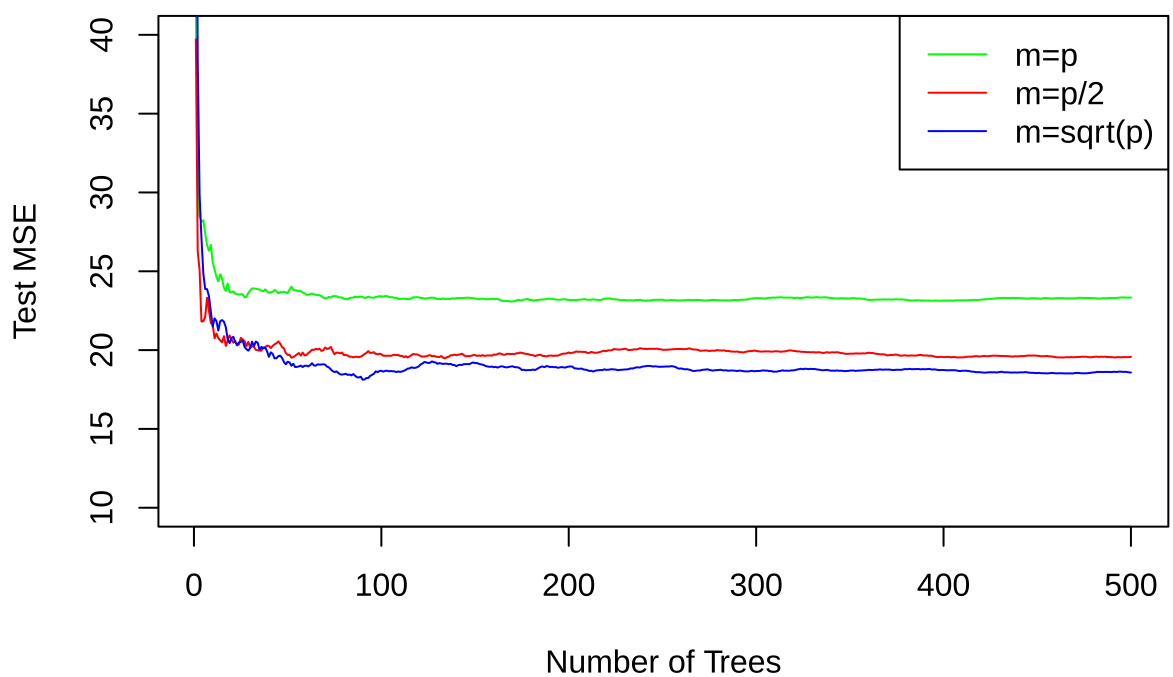
8.7

In the lab, we applied random forests to the Boston data using mtry = 6 and using ntree = 25 and ntree = 500. Create a plot displaying the test error resulting from random forests on this data set for a more comprehensive range of values for mtry and ntree. You can model your plot after Figure 8.10. Describe the results obtained.

```
set.seed(1)
train = sample(dim(Boston)[1], dim(Boston)[1]/2)
X.train = Boston[train, -14]
X.test = Boston[-train, -14]
Y.train = Boston[train, 14]
Y.test = Boston[-train, 14]

p = dim(Boston)[2] - 1
rf.boston.p = randomForest(X.train, Y.train, xtest = X.test, ytest = Y.test,
    mtry = p, ntree = 500)
rf.boston.p.2 = randomForest(X.train, Y.train, xtest = X.test, ytest = Y.test,
    mtry = p/2, ntree = 500)
rf.boston.p.sq = randomForest(X.train, Y.train, xtest = X.test, ytest = Y.test,
    mtry = sqrt(p), ntree = 500)

plot(1:500, rf.boston.p$test$mse, col = "green", type = "l", xlab = "Number of Trees",
    ylab = "Test MSE", ylim = c(10, 40))
lines(1:500, rf.boston.p.2$test$mse, col = "red", type = "l")
lines(1:500, rf.boston.p.sq$test$mse, col = "blue", type = "l")
legend("topright", c("m=p", "m=p/2", "m=sqrt(p)"), col = c("green", "red", "blue"),
    cex = 1, lty = 1)
```



Answer:
When choosing sqrt-number of variables to construct random forest, model's testing error has good performance.

8.8

In the lab, a classification tree was applied to the Carseats data set after converting Sales into a qualitative response variable. Now we will seek to predict Sales using regression trees and related approaches, treating the response as a quantitative variable.

(a)

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

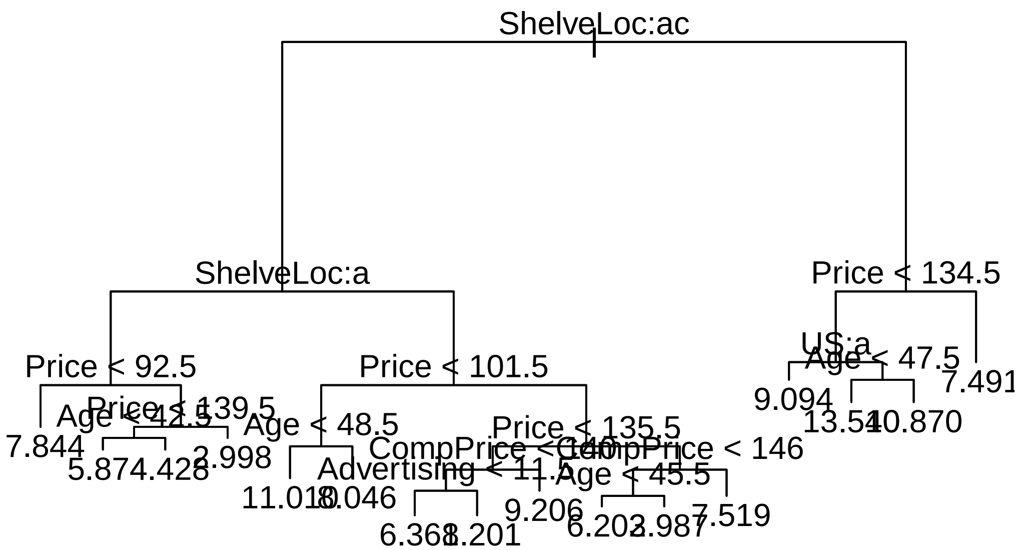
```
attach(Carseats)
set.seed(3)
train = sample(dim(Carseats)[1], dim(Carseats)[1]/2)
Carseats.train = Carseats[train, ]
Carseats.test = Carseats[-train, ]
```

Answer:

(b)

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

```
library(tree)
fit.tree = tree(Sales ~ ., data = Carseats.train)
plot(fit.tree)
text(fit.tree)
```



Answer:

```
fit.pred = predict(fit.tree, Carseats.test)
testMSE = mean((Carseats.test$Sales - fit.pred)^2)
testMSE
```

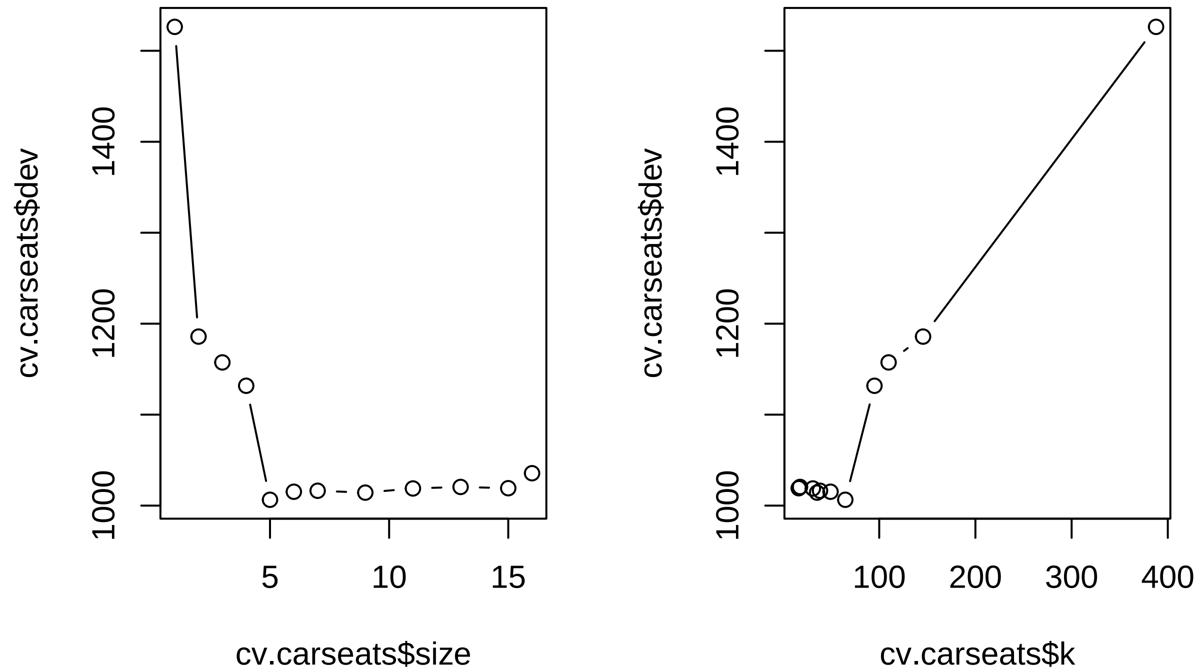
```
## [1] 4.784151
```

The test MSE is 4.78.

(c)

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

```r
cv.carseats = cv.tree(fit.tree, FUN = prune.tree)
par(mfrow = c(1, 2))
plot(cv.carseats$size, cv.carseats$dev, type = "b")
plot(cv.carseats$k, cv.carseats$dev, type = "b")
```

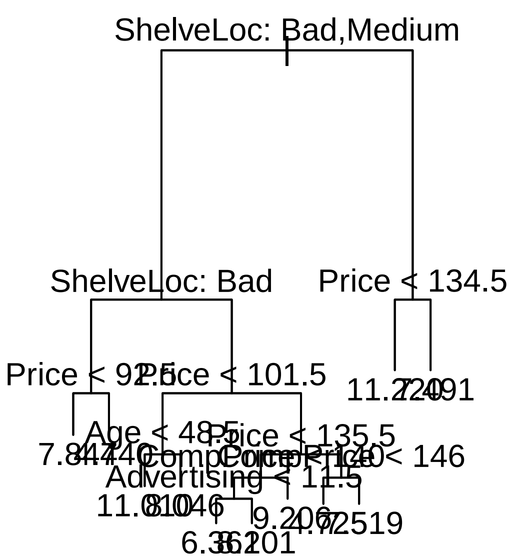


Answer:

```r
pruned.carseats = prune.tree(fit.tree, best = 11)
plot(pruned.carseats)
text(pruned.carseats, pretty = 0)

pred.pruned = predict(pruned.carseats, Carseats.test)
MSE.pruned = mean((Carseats.test$Sales - pred.pruned)^2)
MSE.pruned
```

```
## [1] 4.636068
```



After pruning, the test MSE is 4.63.

(d)

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

```
bag.carseats = randomForest(Sales ~ ., data = Carseats.train, mtry = 10, ntree = 500,
    importance = T)
bag.pred = predict(bag.carseats, Carseats.test)
(MSE.bag = mean((Carseats.test$Sales - bag.pred)^2))
```

Answer:

```
## [1] 2.758793
```

```
importance(bag.carseats)
```

```
##               %IncMSE IncNodePurity
## CompPrice   20.724998    130.421567
## Income       2.616103     66.153373
## Advertising 14.214948    121.847519
## Population  -1.433690     58.523885
## Price       50.544432    408.079671
## ShelveLoc   57.131042    495.464946
## Age         14.442394    118.497755
## Education   -1.849036     38.905898
## Urban       -3.593040      7.957335
## US           5.850580     11.115617
```

Test MSE is 2.82. The most important variables are Price, ShelveLoc.

(e)

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.

```
rf.carseats = randomForest(Sales ~ ., data = Carseats.train, mtry = 5, ntree = 500,
    importance = T)
rf.pred = predict(rf.carseats, Carseats.test)
(MSE.rf = mean((Carseats.test$Sales - rf.pred)^2))
```

Answer:

```
## [1] 2.944176
```

```
importance(rf.carseats)
```

```
##                 %IncMSE IncNodePurity
## CompPrice   13.00451010    127.846808
## Income       1.65976881     88.871227
## Advertising 14.34590005    155.997084
## Population   0.11699701     79.608572
## Price       40.75107160    365.768662
## ShelveLoc   45.93594836    423.846648
## Age         12.31028105    128.337651
## Education   -0.03831749     47.621583
```

```
## Urban       -0.14685440      9.706616
## US           6.44072844     25.417226
```

Test MSE is 2.93. The most important variables are Price, ShelveLoc.

(f)

Now analyze the data using BART, and report your results.

```
xtrain <- Carseats.train[,-1]
ytrain <- Carseats.train[,1]
xtest <- Carseats.test[,-1]
ytest <- Carseats.test[,1]
set.seed(3)
bart.carseats <- gbart(xtrain, ytrain, x.test = xtest)
```

Answer:

```
## *****Calling gbart: type=1
## *****Data:
## data:n,p,np: 200, 14, 200
## y1,yn: 0.331550, 0.851550
## x1,x[n*p]: 129.000000, 1.000000
## xp1,xp[np*p]: 138.000000, 1.000000
## *****Number of Trees: 200
## *****Number of Cut Points: 63 ... 1
## *****burn,nd,thin: 100,1000,1
## *****Prior:beta,alpha,tau,nu,lambda,offset: 2,0.95,0.281075,3,0.197675,7.33845
## *****sigma: 1.007374
## *****w (weights): 1.000000 ... 1.000000
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,14,0
## *****printevery: 100
##
## MCMC
## done 0 (out of 1100)
## done 100 (out of 1100)
## done 200 (out of 1100)
## done 300 (out of 1100)
## done 400 (out of 1100)
## done 500 (out of 1100)
## done 600 (out of 1100)
## done 700 (out of 1100)
## done 800 (out of 1100)
## done 900 (out of 1100)
## done 1000 (out of 1100)
## time: 4s
## trcnt,tecnt: 1000,1000
```

```
yhat.bart <- bart.carseats$yhat.test.mean
(MSE.bart <- mean((ytest - yhat.bart)^2))
```

```
## [1] 1.649891
```

```
ord <- order(bart.carseats$varcount.mean, decreasing = T)
bart.carseats$varcount.mean[ord]
```

```
##       Price   CompPrice  ShelveLoc1         US2      Urban2  ShelveLoc3
##      23.428      18.811      17.477      17.381      17.153      16.617
##      Income  ShelveLoc2         Age         US1      Urban1  Population
##      16.466      16.441      16.227      15.989      15.947      15.255
##   Education Advertising
##      14.968      14.753
```

Tset MSE is 1.59. The most important variables are Price, CompPrice.

8.11

(a)

Create a training set consisting of the first 1,000 observations, and a test set consisting of the remaining observations.

```
train = 1:1000
Caravan$Purchase = ifelse(Caravan$Purchase == "Yes", 1, 0)
Caravan.train = Caravan[train, ]
Caravan.test = Caravan[-train, ]
```

Answer:

(b)

Fit a boosting model to the training set with Purchase as the response and the other variables as predictors. Use 1,000 trees, and a shrinkage value of 0.01. Which predictors appear to be the most important?

Answer: PPERSAUT and MKOOPKLA are most important variables.

(c)

Use the boosting model to predict the response on the test data. Predict that a person will make a purchase if the estimated probability of purchase is greater than 20 %. Form a confusion matrix. What fraction of the people predicted to make a purchase do in fact make one? How does this compare with the results obtained from applying KNN or logistic regression to this data set?

```
boost.prob = predict(boost.caravan, Caravan.test, n.trees = 1000, type = "response")
boost.pred = ifelse(boost.prob > 0.2, 1, 0)
table(Caravan.test$Purchase, boost.pred)
```

Answer:

```
##    boost.pred
##        0    1
##   0 4410  123
##   1  256   33
```

```
33/(123+33)
```

```
## [1] 0.2115385
```

```
####
lm.caravan = glm(Purchase ~ ., data = Caravan.train, family = binomial)
lm.prob = predict(lm.caravan, Caravan.test, type = "response")
```

```r
lm.pred = ifelse(lm.prob > 0.2, 1, 0)
table(Caravan.test$Purchase, lm.pred)
```

```
##    lm.pred
##        0    1
##   0 4183  350
##   1  231   58
```

```r
58/(350+58)
```

```
## [1] 0.1421569
```

21.15% of people predicted to make purchase do in fact make one. 14.21% of people using logistic predicted to make purchase do in fact make one.