rm4064

Ronan McNally

2024-10-14

Question 1

Ronan McNally

Machine Learning

HW3

P1

(Yes, received an A)

$$\mathbb{P}[\text{default} = \text{Yes} \mid X] = \frac{e^{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n}}{1 + e^{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n}}$$

(a)

$$\underset{\text{(logistic)}}{Y \sim} \beta_0 + \underbrace{\beta_1 X_1}_{\text{hours studied}} + \underbrace{\beta_2 X_2}_{\text{GPA}} \qquad \beta_0 = -6,\ \beta_1 = 0.05,\ \beta_2 = 1$$

$$\mathbb{P}[\text{default} = \text{Yes} \mid X] = \frac{e^{\beta_0 + \beta_1 X_1 + \beta_2 X_2}}{1 + e^{\beta_0 + \beta_1 X_1 + \beta_2 X_2}} = \frac{e^{-6 + (.05)(40) + (1)(3.5)}}{1 + e^{-6 + (.05)(40) + (1)(3.5)}} = 0.3775406\ldots$$

$$\boxed{\mathbb{P}[\text{default} = \text{Yes} \mid X] \simeq 0.378}$$

(b)

$$\mathbb{P}[\text{default} = \text{Yes} \mid X] = \mathbb{P}_j = \frac{e^{-6 + (.05) X_1 + (1)(3.5)}}{1 + e^{-6 + (.05) X_1 + (1)(3.5)}} = \frac{e^{-2.5 + (.05) X_1}}{1 + e^{-2.5 + (.05) X_1}} = 0.5 \Rightarrow$$

$$\ldots \Rightarrow e^{-2.5 + (.05) X_1} = 1 \xrightarrow{\ln()} -2.5 + (.05) X_1 = 0 \Rightarrow X_1 = \frac{2.5}{.05} = 50 \text{ hrs.}$$

$$\boxed{X_1 = 50 \text{ hrs}}$$

Question 2

P2

$$P(Yes \mid \bar{x}=4) = \frac{P(x=4 \mid Yes)\, P(Yes)}{P(\bar{x}=4)} \qquad \sigma^2 = 36$$

$P(Yes) \rightarrow$ given $\rightarrow = 0.8$

$$P(\bar{x}=4 \mid Yes) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(\bar{x}-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}\,6}\, e^{-\frac{(4-10)^2}{2(36)}} = \frac{1}{6\sqrt{2\pi}}\, e^{-0.5} = 0.0403284541\ldots$$

from gaussian distribution of 'Yes' companies, where $\bar{x}=10$

$$P(B) = P(B \mid A)P(A) + P(B \mid A^*)P(A^*)$$

$$P(\bar{x}=4) = \underbrace{P(\bar{x}=4 \mid Yes)}_{\text{just calculated}}\,\underbrace{P(Yes)}_{\text{given}, =0.8} + P(\bar{x}=4 \mid No)\,\underbrace{P(No)}_{\text{"given"} =0.2} = (0.0403..)(.8) + (.0532..)(.2) = 0.042910301$$

$$P(\bar{x}=4 \mid No) = \frac{1}{6\sqrt{2\pi}}\, e^{-\frac{(4-0)^2}{2(36)}} = 0.053241334 3$$

$$P(Yes \mid \bar{x}=4) = \frac{P(x=4 \mid Yes)\, P(Yes)}{P(\bar{x}=4)} = \frac{(.040328\ldots)(0.8)}{(.042911\ldots)} = 0.7518524536\ldots$$

$$\boxed{P(Yes \mid \bar{x}=4) \approx 0.752}$$

Question 3

P3

The log-likelihood from the book: $\boxed{\ell(\beta) = \sum_{i=1}^{N} \{ y_i \beta^T x_i - \log(1 + e^{\beta^T x_i}) \}}$

$$\left(\ell \begin{bmatrix} \beta_1 \\ \beta_0 \end{bmatrix} = \sum_{i=1}^{N} \{ y_i \beta^T x_i - \log(1 + e^{\beta^T x_i}) \} = \sum_{i=1}^{N} [y_i(\beta_0 + \beta_1 x_i) - \log(1 + e^{\beta_0 + \beta_1 x_i})] \right.$$

$$\beta^T x_i = [\beta_0 \ \beta_1] \begin{bmatrix} 1 \\ x_i \end{bmatrix} = \beta_0 + \beta_1 x_i$$

the first order optimal condition is the 1st derivative equalling 0...

$$\frac{\partial \ell \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}}{\partial \beta} = \begin{array}{c} g_1 \\ \\ g_2 \end{array} \begin{bmatrix} \partial \ell(\beta) / \partial \beta_0 \\ \partial \ell(\beta) / \partial \beta_1 \end{bmatrix} = \underbrace{\begin{bmatrix} \sum_{i=1}^{N} y_i - \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}} \\ \sum_{i=1}^{N} y_i x_i - \frac{x_i e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}} \end{bmatrix}}_{\text{call this matrix } S \text{ (for score)}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

& 2nd deriv for the function ☆

↓

$$\text{Hessian} = \begin{bmatrix} \frac{\partial g_1(\beta)}{\partial \beta_0} & \frac{\partial g_1(\beta)}{\partial \beta_1} \\ \frac{\partial g_2(\beta)}{\partial \beta_0} & \frac{\partial g_2(\beta)}{\partial \beta_1} \end{bmatrix} = \underbrace{\begin{bmatrix} -\sum_{i=1}^{N} \frac{e^{\beta_0 + \beta_1 x_i}}{(1 + e^{\beta_0 + \beta_1 x_i})^2} & -\sum_{i=1}^{N} x_i \left[\frac{e^{\beta_0 + \beta_1 x_i}}{(1 + e^{\beta_0 + \beta_1 x_i})^2} \right] \\ -\sum_{i=1}^{N} x_i \left[\frac{e^{\beta_0 + \beta_1 x_i}}{(1 + e^{\beta_0 + \beta_1 x_i})^2} \right] & -\sum_{i=1}^{N} x_i^2 \left[\frac{e^{\beta_0 + \beta_1 x_i}}{(1 + e^{\beta_0 + \beta_1 x_i})^2} \right] \end{bmatrix}}_{\text{call this matrix } J}$$

(apply $\frac{d}{dx}\left[\frac{u}{v}\right] = \frac{vu' - uv'}{v^2}$ & simplified...

Newton Method (1-D)

$$x_{t+1} = x_t - \frac{f'(x)}{f''(x)}$$

2D

&

here...

☆ $\beta_{new} = \beta_{old} - [\text{Hessian}]^{-1} [\text{1st deriv}]$

$$\left. \beta_{new} = \beta_{old} - J^{-1} S \right\} \rightarrow \text{iterate } 10\times \text{ in code.}$$

☆ continue in code.

Question 3 continued

```r
library(matlib)

X <- c(0.0, 0.2, 0.4, 0.6, 0.8, 1.0)
Y <- c(0, 0, 0, 1, 0, 1)

B_old <- c(0, 0)



S.1 <- sum(((exp(B_old[1] + B_old[2]*X))/(1+exp(B_old[1] + B_old[2]*X)))-Y)
S.2 <- sum((X*(exp(B_old[1] + B_old[2]*X)/(1+exp(B_old[1] + B_old[2]*X))))-
(X*Y))
Smat <- matrix( c(S.1, S.2), nrow=2, ncol=1)

J.11 <- sum((exp(B_old[1] + B_old[2]*X))/(1+exp(B_old[1] + B_old[2]*X))^2)
J.12 <- sum(X*exp(B_old[1] + B_old[2]*X)/((1+exp(B_old[1] + B_old[2]*X))^2))
J.21 <- sum(X*exp(B_old[1] + B_old[2]*X)/((1+exp(B_old[1] + B_old[2]*X))^2))
J.22 <- sum((X^2)*exp(B_old[1] + B_old[2]*X)/((1+exp(B_old[1] +
B_old[2]*X))^2))
Jmat <- matrix( c(J.11, J.21, J.12, J.22), nrow=2, ncol=2)

B_all <- matrix(0, nrow=2, ncol=10)

for (i in 1:10) {
  B_new <- B_old - inv(Jmat)%*%Smat
  B_old <- B_new



  S.1 <- sum(((exp(B_old[1] + B_old[2]*X))/(1+exp(B_old[1] + B_old[2]*X)))-Y)
  S.2 <- sum((X*(exp(B_old[1] + B_old[2]*X)/(1+exp(B_old[1] + B_old[2]*X))))-
(X*Y))
  Smat <- matrix( c(S.1, S.2), nrow=2, ncol=1)

  J.11 <- sum((exp(B_old[1] + B_old[2]*X))/(1+exp(B_old[1] + B_old[2]*X))^2)
  J.12 <- sum(X*exp(B_old[1] + B_old[2]*X)/((1+exp(B_old[1] +
B_old[2]*X))^2))
  J.21 <- sum(X*exp(B_old[1] + B_old[2]*X)/((1+exp(B_old[1] +
B_old[2]*X))^2))
  J.22 <- sum((X^2)*exp(B_old[1] + B_old[2]*X)/((1+exp(B_old[1] +
B_old[2]*X))^2))
  Jmat <- matrix( c(J.11, J.21, J.12, J.22), nrow=2, ncol=2)

  B_all[,i]<-B_new

  cat("\nIteration number ",i,", beta0= ",B_new[1],"     beta1=
",B_new[2],"\n")
}
```

```
##
## Iteration number  1 , beta0=  -2.380952      beta1=  3.428571
##
## Iteration number  2 , beta0=  -3.522775      beta1=  4.966947
##
## Iteration number  3 , beta0=  -4.022333      beta1=  5.624766
##
## Iteration number  4 , beta0=  -4.096585      beta1=  5.721513
##
## Iteration number  5 , beta0=  -4.09797      beta1=  5.723308
##
## Iteration number  6 , beta0=  -4.09797      beta1=  5.723309
##
## Iteration number  7 , beta0=  -4.09797      beta1=  5.723309
##
## Iteration number  8 , beta0=  -4.09797      beta1=  5.723309
##
## Iteration number  9 , beta0=  -4.09797      beta1=  5.723309
##
## Iteration number  10 , beta0=  -4.09797      beta1=  5.723309
```

Question 4

P4 $\operatorname{cov}(Y) = \operatorname{cov}(AX) = I$

$\downarrow$

find A to make this true

$$\operatorname{cov}(AX) = E\left[(AX - E(AX))(AX - E(AX))^T\right] = E\left[(AX)(AX)^T\right] = AA^T E(XX^T) = \dots$$

$\uparrow$ variance - covariance formula

$E(AX)$: $= AE(X)$ (A is matrix of constants) $= A(0)$ (given $X \sim N(\underset{\sim}{0}, \Sigma)$)

$E(XX^T)$: this is just Σ

$\operatorname{cov}(X) = \Sigma = E[(x - Ex)(x - Ex)^T]$
$= E[(x)(x)^T]$

$$\dots = AA^T \Sigma = I \longrightarrow AA^T U \Lambda U^T = I \rightarrow AA^T = I(U\Lambda U^T)^{-1} = U^{-1}\Lambda^{-1}U^{T^{-1}} = U^T\Lambda^{-1}U = U^T\Lambda^{-1/2}\Lambda^{-1/2}U \rightarrow AA^T$$

$U^{-1} = U^T$

$\Lambda^{-1} = \Lambda$

$U^{T^{-1}} = U^{T^T}$
$U^{T^T} = U$

$= A?$
$A = U^T\Lambda^{-1/2}$
$A^T = (U^T\Lambda^{-1/2})^T = U\Lambda^{-1/2} = \Lambda^{-1/2}U$
$\downarrow$
$\Lambda^T = \Lambda$

AA^T
$\downarrow$
checks out.

given (1) covariance matrices are always symmetric
(2) symmetric matrices always have REAL eigenvalues
$\therefore$ (3) covariance matrix Σ decomposes as $\Sigma = U \Lambda U^T$

U: unitary matrix

Λ: eigenvector

$\therefore$ $\boxed{A = U^T \Lambda^{-1/2}}$

Question 5

(P5) Show $\alpha_k = \frac{n_k - 1}{n - K}$ minimizes $Var(\hat{\sigma}^2)$ under gaussian assumption...

minimize $Var(\hat{\sigma}^2) = \sum_{k=1}^{K} Var(\alpha_k \hat{\sigma}_k^2) = \sum_{k=1}^{K} \alpha_k^2 \, Var(\hat{\sigma}_k) = \sum_{k=1}^{K} \alpha_k^2 \left(\frac{2\sigma^4}{n_k - 1}\right)$

*

* Under a gaussian assumption, it is known that one can scale $\hat{\sigma}_k^2$ such that it follows a chi-squared distribution

$$\frac{(n_k - 1)\hat{\sigma}_k^2}{\sigma^2} \sim \chi^2$$

The variance of a chi-squared distribution is 2 times its d.o.f.

So, here

$$var\left(\frac{(n_k - 1)\hat{\sigma}_k^2}{\sigma^2}\right) = 2(n_k - 1)$$

$\therefore$ By scaling the above expression, we can find $var(\hat{\sigma}_k^2)$.

$$Var\left(\frac{\sigma^2}{n_k - 1} \frac{(n_k - 1)}{\sigma^2} \hat{\sigma}_k^2\right) = var(\hat{\sigma}_k^2)$$

$$= \left(\frac{\sigma^2}{n_k - 1}\right)^2 Var\left(\frac{(n_k - 1)}{\sigma_2} \hat{\sigma}_k^2\right) = \frac{\sigma^4}{(n_k - 1)^2}(2(n_k - 1))$$

$$\therefore Var(\hat{\sigma}_k^2) = \frac{2\sigma^4}{n_k - 1}$$

$\hookrightarrow$ Normally, to minimize, we could take the derivative & set it to 0, HOWEVER, this would give a $\alpha_k = 0$ to minimize and violates the condition that $\sum_{i=1}^{K} \alpha_i = 1$.

To account for this, we must use perform constrained optimization using Lagrange Multipliers where our constraint is the equality $\sum_{k=1}^{K} \alpha_k = 1$

$$f(\alpha_k) = \sum_{k=1}^{K} \alpha_k^2 \left(\frac{2\sigma^4}{n_k - 1}\right) \qquad g(\alpha_k) = 1 - \sum_{k=1}^{K} \alpha_k = 0$$

$$\mathcal{L}(\alpha_k, \lambda) = f(\alpha_k) - \lambda g(\alpha_k)$$

$$\frac{\partial \mathcal{L}(\alpha_k, \lambda)}{\partial \alpha_k} = 4\sigma^4 \sum_{k=1}^{K} \alpha_k \left(\frac{1}{n - 1}\right) - \lambda \sum_{k=1}^{K} 1 = 0$$

say $\alpha_k = x(n_k - 1)$, where x is some constant independent of k

$$\rightarrow = 4\sigma^4 \sum_{k=1}^{K} x \frac{\cancel{n_k - 1}}{\cancel{n_k - 1}} - \lambda K = 4\sigma^4 K x - \lambda K = 0$$

$$\lambda = 4\sigma^4 x = \frac{4\sigma^4}{n - K}$$

$$\frac{\partial \mathcal{L}(\alpha_k, \lambda)}{\partial \lambda} = g(x) = 1 - \sum_{k=1}^{K} \alpha_k = 1 - \sum_{k=1}^{K} x(n_k - 1) = 1 - xn - xK = 0$$

$$\therefore x = \frac{1}{n - K}$$

plug into λ eq

$$\text{w/ } \boxed{\alpha_k = x(n_k - 1) = \frac{n_k - 1}{n - K}}$$

Question 6

P6

10 estimates of $\mathbb{P}[\text{Class is Red} \mid x]$:

$0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75$

Method 1 $\rightarrow$ Majority Approach

$\Rightarrow$ if $\mathbb{P}(\text{Red}|x) \geq 0.5) \rightarrow$ Red

$\mathbb{P}(\text{Red}|x) < 0.5 \rightarrow$ Green

$0.1, 0.15, 0.2, 0.2 \,\vdots\, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75$

$$\underbrace{\underbrace{G, G, G, G}_{\text{4 Green}} \,\vdots\, \underbrace{R, R, R, R, R, R}_{\text{6 Red}}}_{\text{Majority Vote decides } \underline{\text{RED}}}$$

Method 2 $\rightarrow$ Average Probability

$$\hookrightarrow \frac{(0.1+0.15+0.2+0.2+0.55+0.6+0.6+0.65+0.7+0.75)}{10} = 0.45 < 0.5 \rightarrow$$ Average Probability decides GREEN

Question 7

```r
# P7 Part A

library(ISLR2)
set.seed(1000)

n <- 800
data(OJ)

train_indices <- sample(1:nrow(OJ), n)
train <- OJ[train_indices, ]

test_indices <- -train_indices
test <- OJ[test_indices, ]



# P7 Part B

library(tree)
OJ_tree <- tree(train$Purchase ~ ., data=train)#remove class?
summary(OJ_tree)

##
## Classification tree:
## tree(formula = train$Purchase ~ ., data = train)
## Variables actually used in tree construction:
## [1] "LoyalCH"     "PriceDiff"   "SalePriceMM"
## Number of terminal nodes:  8
## Residual mean deviance:  0.7486 = 592.9 / 792
## Misclassification error rate: 0.16 = 128 / 800

cat("\nP7B output\n\n")

##
## P7B output

cat("\n \t The number of terminal nodes is 8 \n \t The training error
(misclassification error rate) is 0.16 \n \n")

##
##       The number of terminal nodes is 8
##       The training error (misclassification error rate) is 0.16
##

# P7 Part C

cat("\nP7C output\n\n")
```

```
##
## P7C output
```

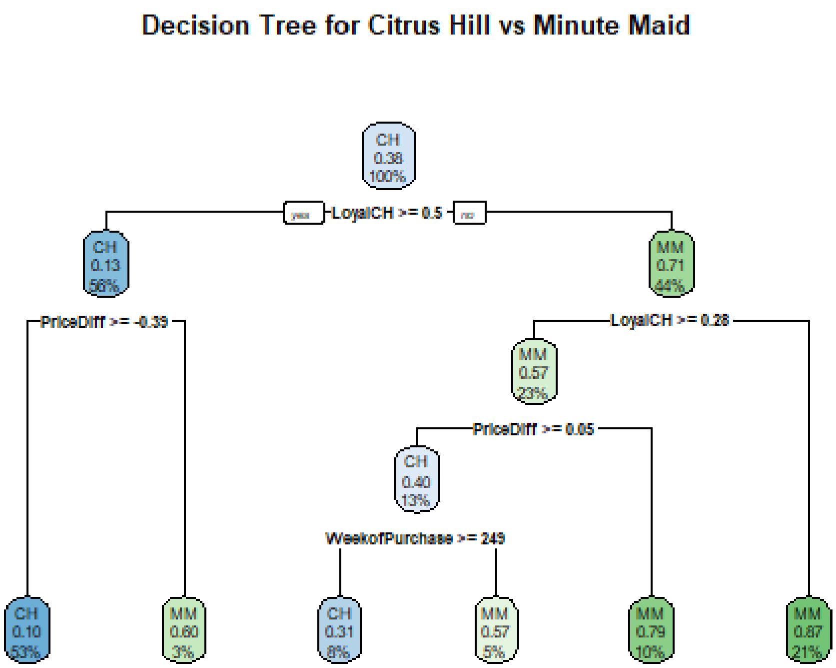
```r
OJ_tree
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 800 1066.00 CH ( 0.61500 0.38500 )
##    2) LoyalCH < 0.5036 353  422.60 MM ( 0.28612 0.71388 )
##      4) LoyalCH < 0.276142 170  131.00 MM ( 0.12941 0.87059 )
##        8) LoyalCH < 0.035047 57   10.07 MM ( 0.01754 0.98246 ) *
##        9) LoyalCH > 0.035047 113  108.50 MM ( 0.18584 0.81416 ) *
##      5) LoyalCH > 0.276142 183  250.30 MM ( 0.43169 0.56831 )
##       10) PriceDiff < 0.05 78   79.16 MM ( 0.20513 0.79487 ) *
##       11) PriceDiff > 0.05 105  141.30 CH ( 0.60000 0.40000 ) *
##    3) LoyalCH > 0.5036 447  337.30 CH ( 0.87472 0.12528 )
##      6) LoyalCH < 0.764572 187  206.40 CH ( 0.75936 0.24064 )
##       12) SalePriceMM < 2.125 120  156.60 CH ( 0.64167 0.35833 )
##         24) PriceDiff < -0.35 16   17.99 MM ( 0.25000 0.75000 ) *
##         25) PriceDiff > -0.35 104  126.70 CH ( 0.70192 0.29808 ) *
##       13) SalePriceMM > 2.125 67   17.99 CH ( 0.97015 0.02985 ) *
##      7) LoyalCH > 0.764572 260   91.11 CH ( 0.95769 0.04231 ) *
```

```r
cat("\n\nTerminating node: '24) PriceDiff < -0.35 16   17.99 MM ( 0.25000
0.75000 ) *' \n \nTerminating node (24) divides along price difference The -
.35 refers to MM being 35 cents less expensive than CH as a dividing line.
Meaning if CH is less than 35 cents more expensive than MM, people choose MM.
If CH is more than 35 cents more expensive than MM, people choose CH. The
numer of people in this decision split is n=16. The 'deviance' is a
representation of the 'purity' of the dividing line (a metric of how many MM
are in my CH bucket, how many CH in my MM bucket). MM represents the choice
direction (e.g. people choose MM when <-.35) and the probability parenthesis
is the CH vs MM probability within this MM bucket.")
```

```
##
##
## Terminating node: '24) PriceDiff < -0.35 16   17.99 MM ( 0.25000 0.75000 )
*'
##
## Terminating node (24) divides along price difference The -.35 refers to MM
being 35 cents less expensive than CH as a dividing line. Meaning if CH is
less than 35 cents more expensive than MM, people choose MM. If CH is more
than 35 cents more expensive than MM, people choose CH. The numer of people
in this decision split is n=16. The 'deviance' is a representation of the
'purity' of the dividing line (a metric of how many MM are in my CH bucket,
how many CH in my MM bucket). MM represents the choice direction (e.g. people
choose MM when <-.35) and the probability parenthesis is the CH vs MM
probability within this MM bucket.
```

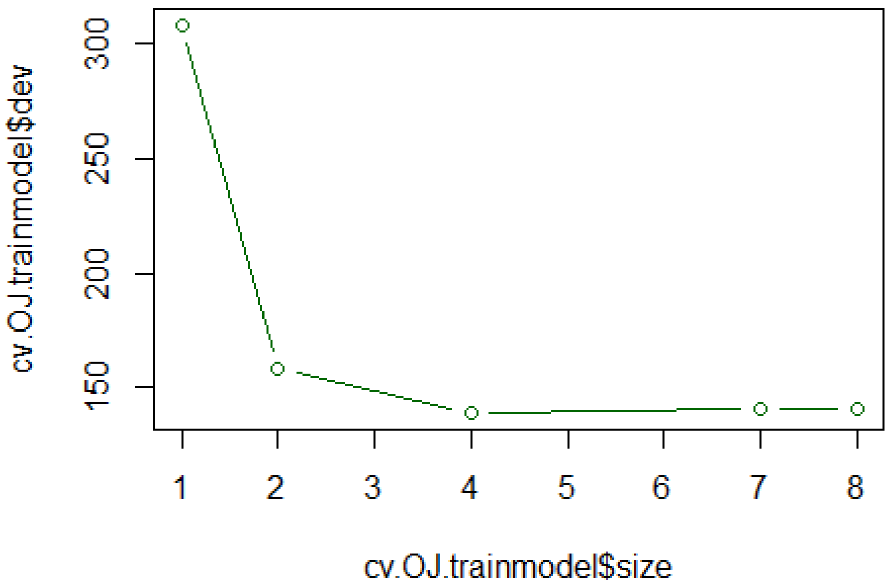
```
# P7 Part D

cat("\nP7 Part D\n")

##
## P7 Part D

library(rpart)
library(rpart.plot)
OJ_tree_pD <- rpart(train$Purchase ~ ., data=train, method = "class")
rpart.plot(OJ_tree_pD, main="Decision Tree for Citrus Hill vs Minute Maid")
```




```
#other plotting option:
#plot(OJ_tree)
#text(OJ_tree, pretty=0)

# P7 Part E

cat("\nP7 Part E\n")

##
## P7 Part E

test.predict.7e <- predict(OJ_tree, test, type="class")
table(test.predict.7e, test$Purchase)
```

```
##
## test.predict.7e  CH  MM
##              CH 150  38
##              MM  11  71

cat("\nPercent Incorrect Predictions for Part E:\n")

##
## Percent Incorrect Predictions for Part E:

cat((1-(150+71)/270)*100, "%")

## 18.14815 %

# P7 Part F

cat("\nP7 Part F\n")

##
## P7 Part F

cv.OJ.trainmodel <- cv.tree(OJ_tree, FUN = prune.misclass)
names(cv.OJ.trainmodel)

## [1] "size"   "dev"    "k"      "method"

cv.OJ.trainmodel

## $size
## [1] 8 7 4 2 1
##
## $dev
## [1] 141 141 139 158 308
##
## $k
## [1]       -Inf   0.000000   2.666667  10.500000 151.000000
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"

cat("\nBest model size according to CV output is a size of 4\n")

##
## Best model size according to CV output is a size of 4

# P7 Part G
cat("\nP7 Part G\n")

##
## P7 Part G
```

```
plot(cv.OJ.trainmodel$size, cv.OJ.trainmodel$dev, type="b", col="darkgreen")
```




```
# P7 Part H

cat("\nP7 Part H\n")

##
## P7 Part H

cat("\nThe results from F and G appear to agree, model size of 4 is
optimal\n")

##
## The results from F and G appear to agree, model size of 4 is optimal

# P7 Part I
cat("\nP7 Part I\n")

##
## P7 Part I

OJ.trainmodel.pruned <- prune.misclass(OJ_tree, best = 4)

# P7 Part J

cat("\nP7 Part J\n")
```

```
##
## P7 Part J

OJ.trainmodel.pruned.summary <- summary(OJ.trainmodel.pruned)
OJ.trainmodel.pruned.summary

##
## Classification tree:
## snip.tree(tree = OJ_tree, nodes = 4:3)
## Variables actually used in tree construction:
## [1] "LoyalCH"   "PriceDiff"
## Number of terminal nodes:  4
## Residual mean deviance:  0.8653 = 688.8 / 796
## Misclassification error rate: 0.17 = 136 / 800

cat("\nThe error in part J for the pruned tree is slightly higher than the
error for the unpruned tree in part B (error rate of 0.17 vs 0.16
respectively)\n")

##
## The error in part J for the pruned tree is slightly higher than the error
for the unpruned tree in part B (error rate of 0.17 vs 0.16 respectively)

# P7 Part K

cat("\nP7 Part K\n")

##
## P7 Part K

test.predict.7k <- predict(OJ.trainmodel.pruned, test, type="class")
table(test.predict.7k, test$Purchase)

##
## test.predict.7k  CH  MM
##              CH 150  44
##              MM  11  65

#summary(test.predict.7k)

cat("\nPercent Incorrect Predictions for Part K:\n")

##
## Percent Incorrect Predictions for Part K:

cat((1-(150+71)/270)*100, "%")

## 18.14815 %

cat("It would appear the model size of 4 (pruned) has a slightly higher test
error rate compared to the unpruned model size.")
```

```
## It would appear the model size of 4 (pruned) has a slightly higher test
error rate compared to the unpruned model size.
```