Group Assignment 7

Group 5

2022-11-03

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

Exercise 8.4 Problem #9: This problem involves the OJ data set which is part of the ISLR package.

#a

```
set.seed(10)
library(ISLR)
library(tree)
```

```
## Warning: 程辑包'tree'是用R版本4.2.2 来建造的
```

```
dim(OJ)
```

```
## [1] 1070   18
```

```
train = sample(1:nrow(OJ), 800)
test = OJ[-train,]
train = data.frame(OJ[train,])
test = data.frame(test)
```

#b

```
#attach(train)
treeoj = tree(Purchase ~.,data = train)
summary(treeoj)
```

```
##
## Classification tree:
## tree(formula = Purchase ~ ., data = train)
## Variables actually used in tree construction:
## [1] "LoyalCH"   "DiscMM"    "PriceDiff"
## Number of terminal nodes:  7
## Residual mean deviance:  0.7983 = 633 / 793
## Misclassification error rate: 0.1775 = 142 / 800
```

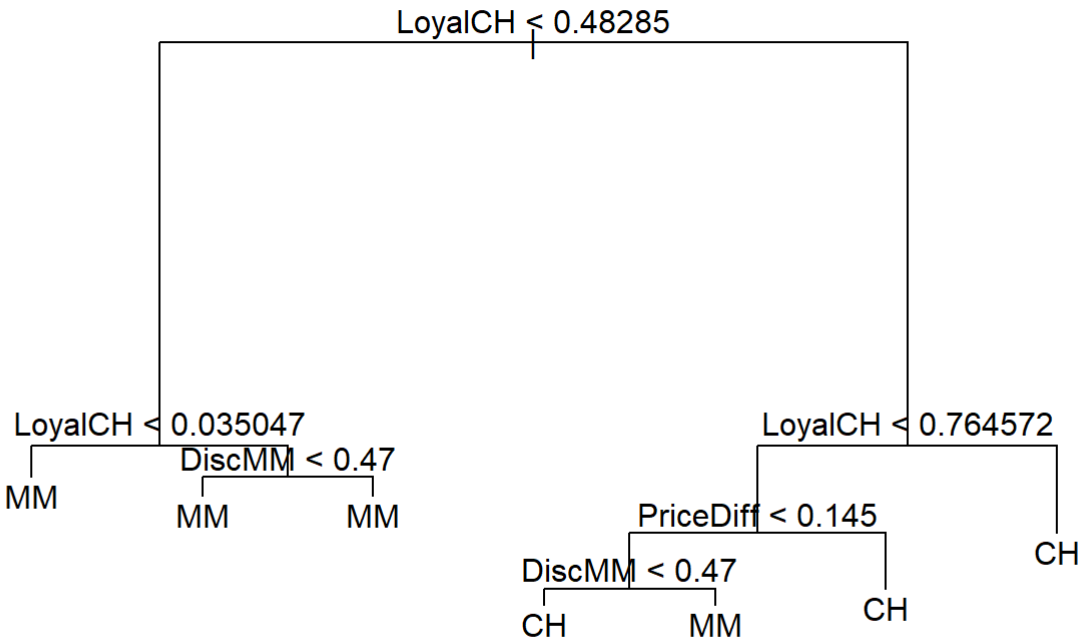
There are 7 nodes in the tree, and the error rate is 0.1775

#c,d

```
treeoj
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 800 1067.000 CH ( 0.61375 0.38625 )
##    2) LoyalCH < 0.48285 290  315.900 MM ( 0.23448 0.76552 )
##      4) LoyalCH < 0.035047 51    9.844 MM ( 0.01961 0.98039 ) *
##      5) LoyalCH > 0.035047 239  283.600 MM ( 0.28033 0.71967 )
##       10) DiscMM < 0.47 220  270.500 MM ( 0.30455 0.69545 ) *
##       11) DiscMM > 0.47 19    0.000 MM ( 0.00000 1.00000 ) *
##    3) LoyalCH > 0.48285 510  466.000 CH ( 0.82941 0.17059 )
##      6) LoyalCH < 0.764572 245  300.200 CH ( 0.69796 0.30204 )
##       12) PriceDiff < 0.145 99  137.000 MM ( 0.47475 0.52525 )
##         24) DiscMM < 0.47 82  112.900 CH ( 0.54878 0.45122 ) *
##         25) DiscMM > 0.47 17   12.320 MM ( 0.11765 0.88235 ) *
##       13) PriceDiff > 0.145 146  123.800 CH ( 0.84932 0.15068 ) *
##      7) LoyalCH > 0.764572 265  103.700 CH ( 0.95094 0.04906 ) *
```

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



From the tree we can know that node 4 is MM, means its loyalCH < 0.035047

#e

```
pred.OJ = predict(treeoj, newdata = test, type = "class")
table(pred.OJ,test$Purchase)
```

```
##
## pred.OJ  CH  MM
##      CH 135  20
##      MM  27  88
```

error rate is 0.2764

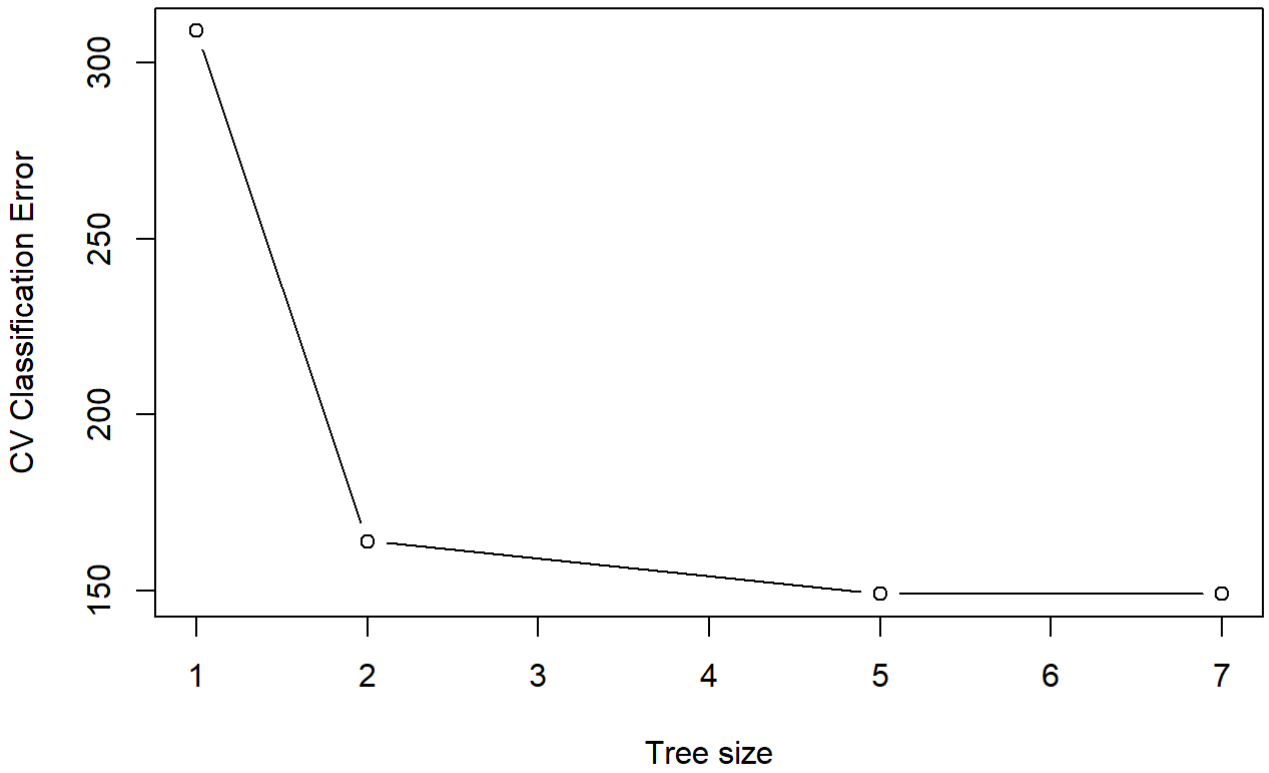
#f

```
set.seed(10)
cv.OJ = cv.tree(treeoj, FUN=prune.misclass)
cv.OJ
```

```
## $size
## [1] 7 5 2 1
##
## $dev
## [1] 149 149 164 309
##
## $k
## [1]       -Inf   0.000000   4.333333 154.000000
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

#g,h

```
plot(cv.OJ$size, cv.OJ$dev, xlab = "Tree size", ylab = "CV Classification Error", type = "b")
```



size 6 has the lowest error rate

#i

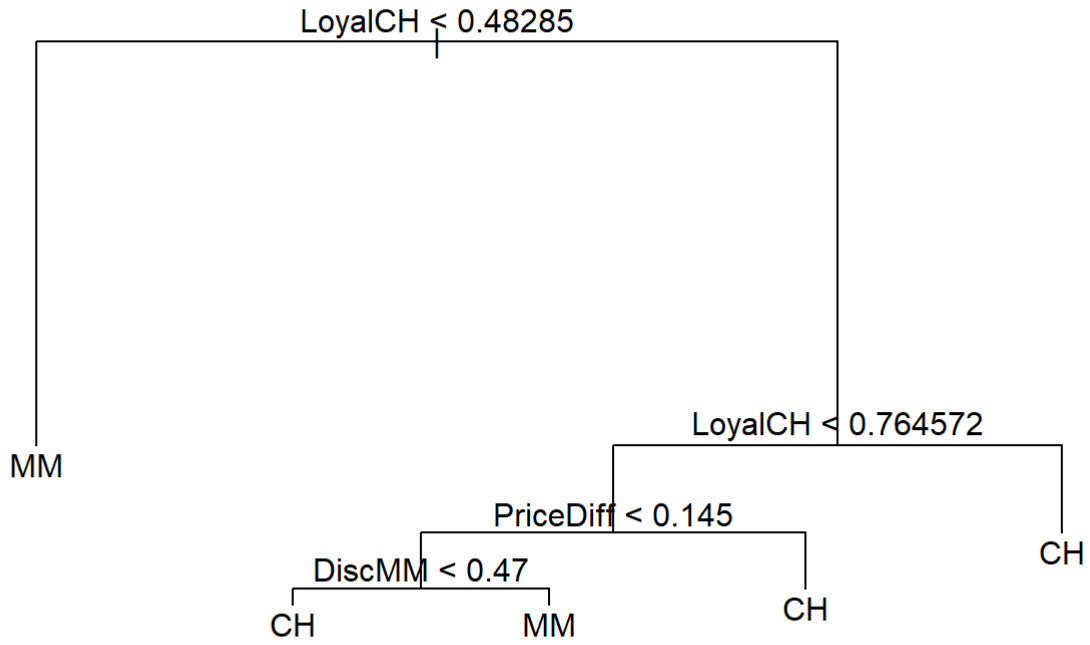
```
prune.OJ = prune.misclass(treeoj,best=5)
summary(prune.OJ)
```

```
##
## Classification tree:
## snip.tree(tree = treeoj, nodes = 2L)
## Variables actually used in tree construction:
## [1] "LoyalCH"   "PriceDiff" "DiscMM"
## Number of terminal nodes:  5
## Residual mean deviance:  0.841 = 668.6 / 795
## Misclassification error rate: 0.1775 = 142 / 800
```

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

```
##
## pred.prune  CH  MM
##         CH 135  20
##         MM  27  88
```

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



We can see that the pruned and unpruned trees almost have the same training error rate and test error rate。