P9120 HW4 answer

Guojing Wu | UNI: gw2383

11/18/2019

1. Suppose we produce ten bootstrapped smaples 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(Class is Red|X):

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

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?

- For majority vote approach, we have 6 voted for red and 4 voted for green, so the final classification will be red.

- For average probability approach, we have p = 0.45, so the final classification will be green.

2. This problem involves the OJ data set which is part of the ISLR package.

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

(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?

```
Classification tree:
tree(formula = Purchase ~ ., data = Train)
Variables actually used in tree construction:
[1] "LoyalCH"       "PriceDiff"     "ListPriceDiff" "SalePriceMM"
Number of terminal nodes:  8
Residual mean deviance:  0.7235 = 573 / 792
Misclassification error rate: 0.1588 = 127 / 800
```

The tree only uses 4 variables: `LoyalCH`, `PriceDiff`, `ListPriceDiff` and `SalePriceMM`. The training error rate (miss classification rate) is 0.159. The tree has 8 terminal nodes.

(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.

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

```
1) root 800 1070.00 CH ( 0.61000 0.39000 )
  2) LoyalCH < 0.5036 355  420.20 MM ( 0.27887 0.72113 )
    4) LoyalCH < 0.280875 167  130.20 MM ( 0.13174 0.86826 )
      8) LoyalCH < 0.0356415 59   10.14 MM ( 0.01695 0.98305 ) *
      9) LoyalCH > 0.0356415 108  106.40 MM ( 0.19444 0.80556 ) *
    5) LoyalCH > 0.280875 188  254.40 MM ( 0.40957 0.59043 )
     10) PriceDiff < 0.05 81   74.58 MM ( 0.17284 0.82716 ) *
     11) PriceDiff > 0.05 107  144.90 CH ( 0.58879 0.41121 ) *
  3) LoyalCH > 0.5036 445  336.80 CH ( 0.87416 0.12584 )
    6) LoyalCH < 0.764572 180  204.60 CH ( 0.74444 0.25556 )
     12) ListPriceDiff < 0.18 49   66.27 MM ( 0.40816 0.59184 ) *
     13) ListPriceDiff > 0.18 131  101.10 CH ( 0.87023 0.12977 )
       26) SalePriceMM < 2.04 51   59.94 CH ( 0.72549 0.27451 ) *
       27) SalePriceMM > 2.04 80   25.59 CH ( 0.96250 0.03750 ) *
    7) LoyalCH > 0.764572 265   85.16 CH ( 0.96226 0.03774 ) *
```

For example, we picked up the terminal node (11), the splitting variable at this node is `PriceDiff`, the splitting value is 0.05. There are 107 points in the subtree below this node. The deviance for all points contained in region below this node is 144.90. The prediction at this node is `Purchase = CH`. About 58.9% points in this node have `Purchase = CH` and about 41.1% points in this node have `Purchase = MM`.

(d) Create a plot of the tree, and interpret the results.
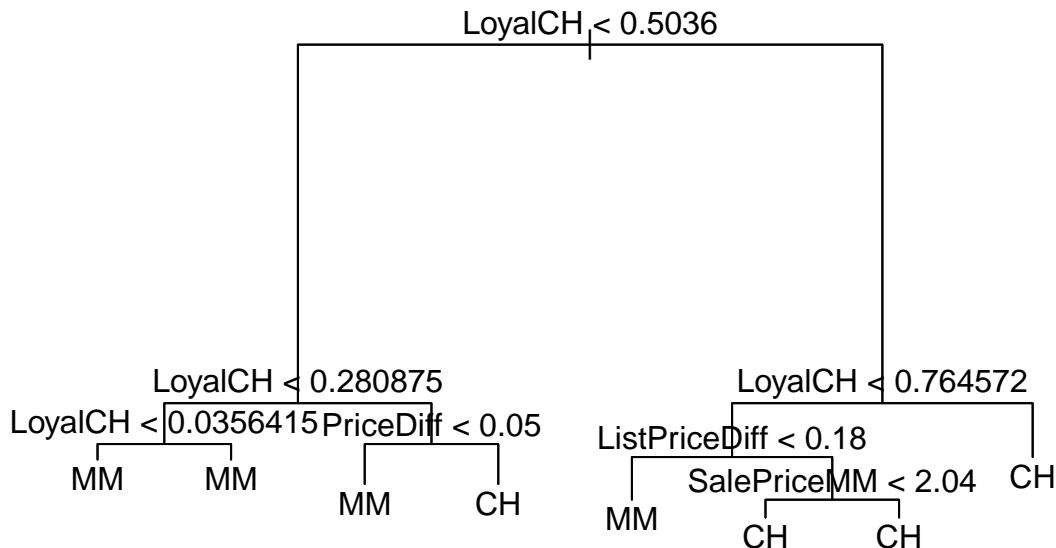
`LoyalCH` may be the most important variable here, when `LoyalCH` < 0.0356415 the tree predicts `Purchase = MM` and when `LoyalCH` > 0.764572 the tree predicts `Purchase = CH`. For intermediate value of `LoyalCH` between 0.0356415 and 0.764572, the prediction also depends on variable `PriceDiff`, `ListPriceDiff` and `SalePriceMM`.

(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?
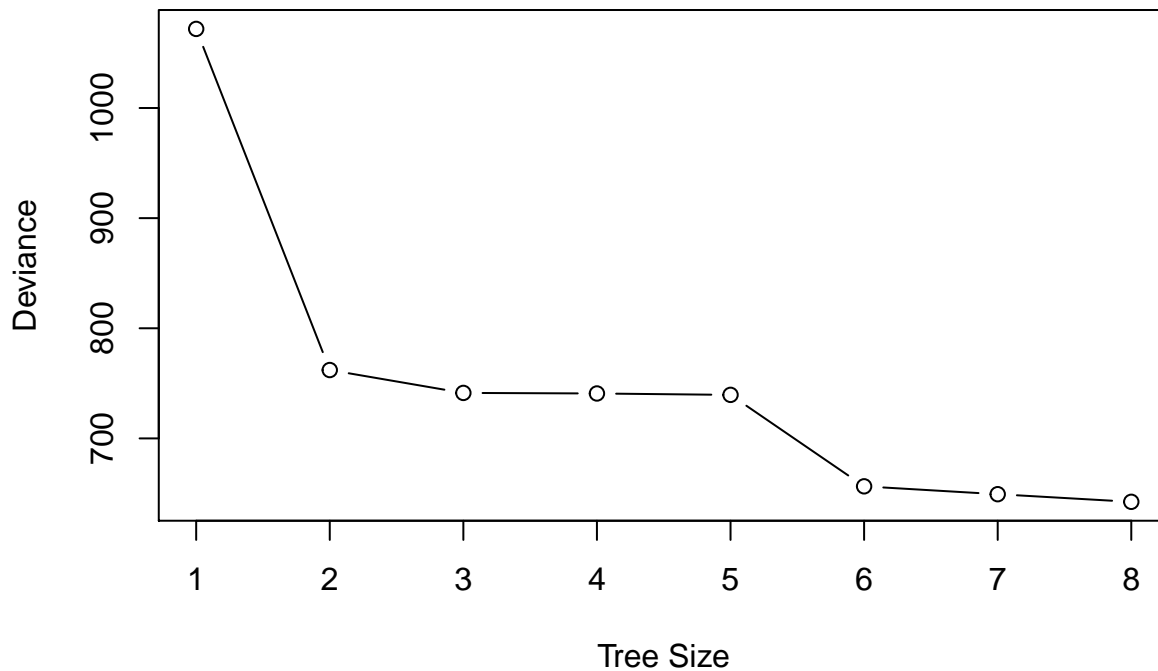
```
    pred.tree
     CH  MM
 CH 142  23
```

```
MM  36  69
```

The test error rate is 0.219.

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

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

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

We can't decide based on the plot, because the CV classification error rate decreases through tree size.

(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.

```
Classification tree:
snip.tree(tree = fit.tree, nodes = c(4L, 13L, 5L))
Variables actually used in tree construction:
[1] "LoyalCH"       "ListPriceDiff"
Number of terminal nodes:  5
Residual mean deviance:  0.8014 = 637.1 / 795
Misclassification error rate: 0.1825 = 146 / 800
```

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

Misclassification error of pruned tree is higher than that of original tree.

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

So the test error rates for pruned tree is 0.233, which is higher than the unpruned tree

3. On the book website, www.StatLearning.com, there is a gene expression data set (`Ch10Ex11.csv`) that consists of 40 tissue samples with measurements on 1,000 genes. The first 20 samples are from healthy patients, while the second 20 are from a diseased group.

(a) Load in the data using `read.csv()`. You will need to select `header=F`.

(b) Apply hierarchical clustering to the samples using correlation-based distance, and plot the dendrogram. Do the genes separate the samples into the two groups? Do your results depend on the type of linkage used?

(c) Your collaborator wants to know which genes differ the most across the two groups. Suggest a way to answer this question, and apply it here.

Appendix

```r
knitr::opts_chunk$set(echo = FALSE, message = FALSE, warning = FALSE, comment = "")
library(tidyverse)
library(ISLR) # for OJ dataset
library(tree)
set.seed(100)

# exercise 1
x = c(0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75)
sum.red = sum(x >= 0.5)
sum.green = sum(x < 0.5)
x.mean = mean(x)
# loading OJ data
data(OJ)
OJ = OJ %>% as_tibble()

# train test split
splitIndex = sample(nrow(OJ), 800, replace = F)
Train = OJ[splitIndex,]
Test = OJ[-splitIndex,]

# modeling
fit.tree <- tree(Purchase ~ ., data = Train)
fit.sum <- summary(fit.tree)
fit.sum

# print the tree
fit.tree

# plot the tree
plot(fit.tree)
text(fit.tree, pretty = 0)
```

```r
# confusion matrix
pred.tree = predict(fit.tree, Test, type = "class")
table(Test$Purchase, pred.tree)

# optimal tree size
fit.tree.cv <- cv.tree(fit.tree, FUN = prune.tree)

# plot tree size and CV error rate
plot(fit.tree.cv$size, fit.tree.cv$dev, type = "b", xlab = "Tree Size", ylab = "Deviance")

# pruned tree
fit.tree.pruned <- prune.tree(fit.tree, best = 5)
summary(fit.tree.pruned)

# compare between pruned tree and unpruned tree
pred.tree.pruned <- predict(fit.tree.pruned, Test, type = "class")
```