NBA 4920/6921 Lecture 17

Tree Methods: Classification Application

Murat Unal

10/28/2021

```r
rm(list=ls())
options(digits = 3, scipen = 999)
library(tidyverse)
library(ISLR)
library(lmtest)
library(sandwich)
library(jtools)
library(caret)
library(leaps)
library(future.apply)
library(glmnet)
library(rpart)
library(rpart.plot)
library(ROCR)
set.seed(2)
```

- We'll use the Carseats dataset from ISLR

```
Carseats <- ISLR::Carseats
Carseats = na.omit(Carseats)
```

```
dim(Carseats)
```

```
[1] 400  11
```

```
names(Carseats)
```

```
 [1] "Sales"       "CompPrice"   "Income"      "Advertising
 [6] "Price"       "ShelveLoc"   "Age"         "Education"
[11] "US"
```

- Let's modify the response from its original numeric variable to a categorical variable with two levels: `high` and `low`

```
Carseats$Sales = as.factor(ifelse(Carseats$Sales <= 8,
                                  "Low", "High"))
```

```
train = sample(1:nrow(Carseats), 0.7*nrow(Carseats))
```

- Grow a classification tree

```
sales_tree = rpart(Sales ~ .,data = Carseats[train,],
                   method="class")

sales_tree
```

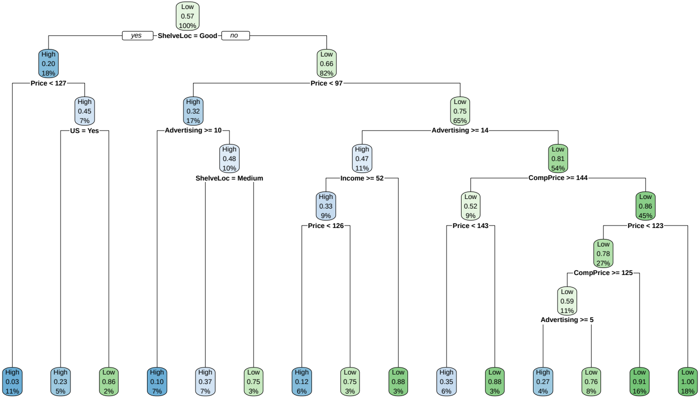
```
n= 280

node), split, n, loss, yval, (yprob)
      * denotes terminal node

  1) root 280 119 Low (0.4250 0.5750)
    2) ShelveLoc=Good 51  10 High (0.8039 0.1961)
      4) Price< 127 31   1 High (0.9677 0.0323) *
      5) Price>=127 20   9 High (0.5500 0.4500)
       10) US=Yes 13   3 High (0.7692 0.2308) *
       11) US=No 7   1 Low (0.1429 0.8571) *
    3) ShelveLoc=Bad,Medium 229  78 Low (0.3406 0.6594)
      6) Price< 96.5 47  15 High (0.6809 0.3191)
       12) Advertising>=9.5 20   2 High (0.9000 0.1000) *
```

- We can visualize our tree model with `rpart.plot()`

```
rpart.plot(sales_tree)
```



- Behind the scenes `rpart()` is automatically applying a range of cost complexity α values to prune the tree.
- To compare the error for each α value, it performs a 10-fold CV (by default)

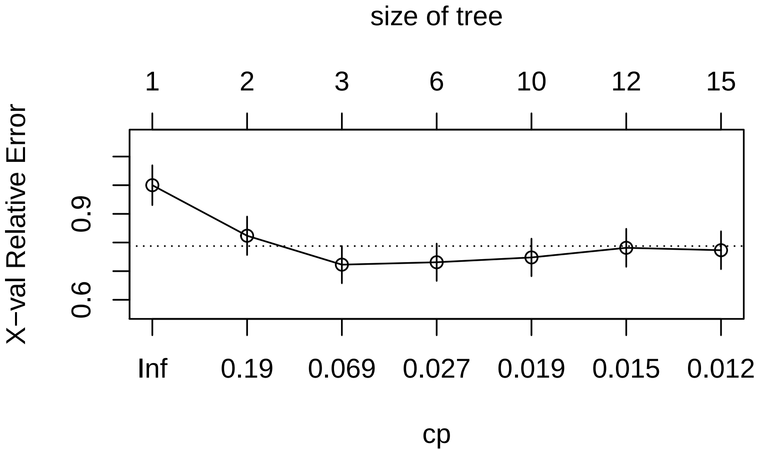
```
sales_tree$cptable
```

```
      CP nsplit rel error xerror   xstd
1 0.2605      0     1.000  1.000 0.0695
2 0.1429      1     0.739  0.824 0.0671
3 0.0336      2     0.597  0.723 0.0649
4 0.0210      5     0.496  0.731 0.0651
5 0.0168      9     0.412  0.748 0.0655
6 0.0140     11     0.378  0.782 0.0662
7 0.0100     14     0.336  0.773 0.0660
```

- Here we don't find much improvement after 3 terminal nodes

- Thus, we could use a tree with just 3 terminal nodes and reasonably expect to experience similar results within a small margin of error.

```
plotcp(sales_tree)
```



- Grow another tree with `maxdepth=2`

```
sales_tree2 = rpart(Sales ~ .,data = Carseats[train,],
                    method="class", maxdepth=2)

sales_tree2
```

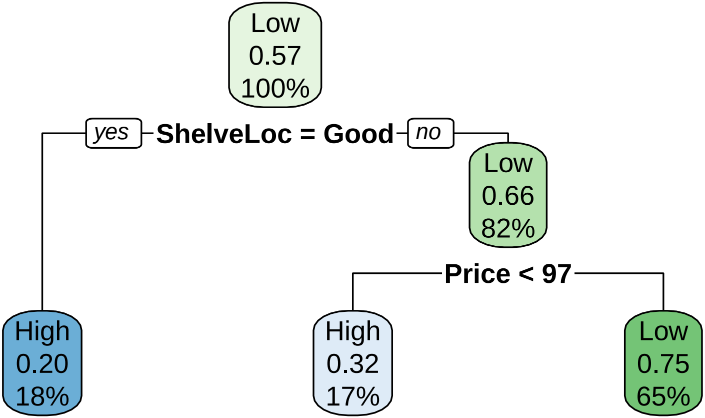
```
n= 280

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 280 119 Low (0.425 0.575)
  2) ShelveLoc=Good 51  10 High (0.804 0.196) *
  3) ShelveLoc=Bad,Medium 229  78 Low (0.341 0.659)
    6) Price< 96.5 47  15 High (0.681 0.319) *
    7) Price>=96.5 182  46 Low (0.253 0.747) *
```

- We can visualize our tree model with `rpart.plot()`

```
rpart.plot(sales_tree2)
```



- Make predictions on the test data using both trees

```
sales_pred <- data.frame("p_hat"=predict(sales_tree,
                Carseats[-train,],type = "prob")[,"High"],
                        "predicted"=predict(sales_tree,
                Carseats[-train,], type = "class"),
                        "actual"=Carseats[-train,"Sales"])

sales_pred2 <- data.frame("p_hat"=predict(sales_tree2,
                Carseats[-train,],type = "prob")[,"High"],
                      "predicted"=predict(sales_tree2,
                Carseats[-train,], type = "class"),
                      "actual"=Carseats[-train,"Sales"])
```

- Call the `confusion matrix`

```
cm <- confusionMatrix(data=sales_pred$predicted,
                      reference=sales_pred$actual,
                      positive="High")
cm$table
```

```
          Reference
Prediction High Low
      High   34  14
      Low    11  61
```

- Call the `confusion matrix`

```
cm2 <- confusionMatrix(data=sales_pred2$predicted,
                       reference=sales_pred2$actual,
                       positive="High")
cm2$table
```

```
          Reference
Prediction High Low
      High   31  18
      Low    14  57
```

- Performance metrics

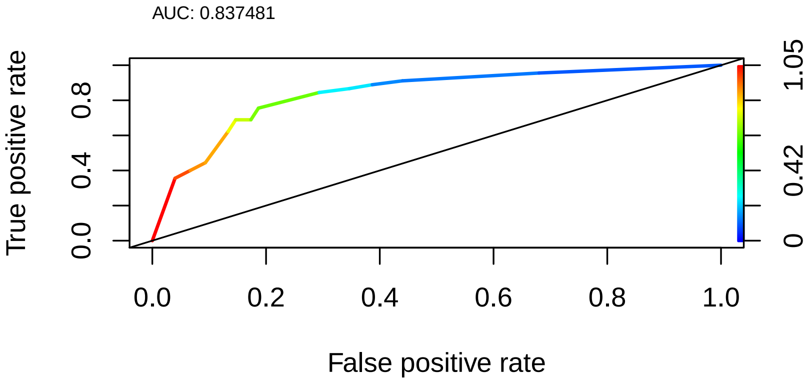
```
c(cm$overall[1],cm$byClass[c(1,2,7)])
```

```
   Accuracy Sensitivity Specificity          F1
      0.792       0.756       0.813       0.731
```

```
c(cm2$overall[1],cm2$byClass[c(1,2,7)])
```

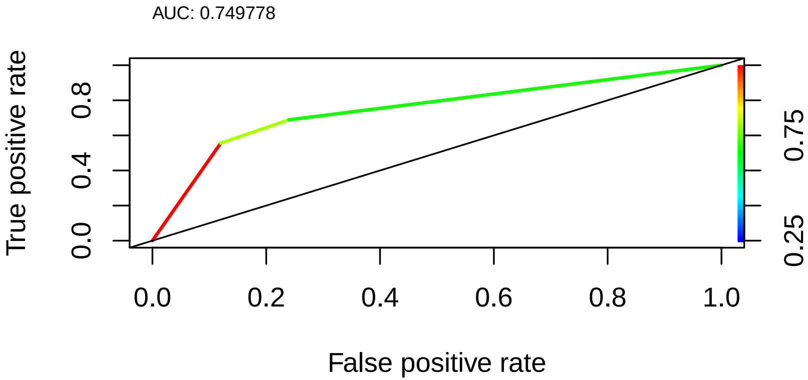
```
   Accuracy Sensitivity Specificity          F1
      0.733       0.689       0.760       0.660
```

- ROC curve



AUC: 0.837481
True positive rate
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate
0
0.42
1.05

- ROC curve


AUC: 0.749778
True positive rate
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate
0.25
0.75