NBA 4920/6921 Lecture 15

Tree Methods Application

Murat Unal

10/21/2021

```r
rm(list=ls())
options(digits = 3, scipen = 999)
library(tidyverse)
library(ISLR)
library(cowplot)
library(ggcorrplot)
library(stargazer)
library(corrr)
library(lmtest)
library(sandwich)
library(MASS)
library(car)
library(jtools)
library(caret)
library(leaps)
library(future.apply)
library(glmnet)
library(rpart)
library(rpart.plot)
library(ROCR)
```

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

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

Regression Trees

- To train decision trees in R, we can use **caret** , which draws upon `rpart`
- You can get a list of models supported by **caret**, `names(getModelInfo())`
- The tunable parameters for a given model, `modelLookup("rpart")`
- To `train()` our model in **caret**
  Define the `method` as `rpart`
  The main tuning parameter is `cp`, the complexity parameter

```
hit_tree = train(
Salary ~ .,
data = Hitters[train,],
method = "rpart",
trControl = trainControl("cv", number = 5),
# tuneLength = 20
tuneGrid = data.frame(cp = seq(0, 0.1, by = 0.001))
)
names(hit_tree)
```

```
 [1] "method"       "modelInfo"    "modelType"    "results'
 [6] "bestTune"     "call"         "dots"         "metric"
[11] "finalModel"   "preProcess"   "trainingData" "resample
[16] "perfNames"    "maximize"     "yLimits"      "times"
[21] "terms"        "coefnames"    "contrasts"    "xlevels'
```

- To get the CV-chosen final tree

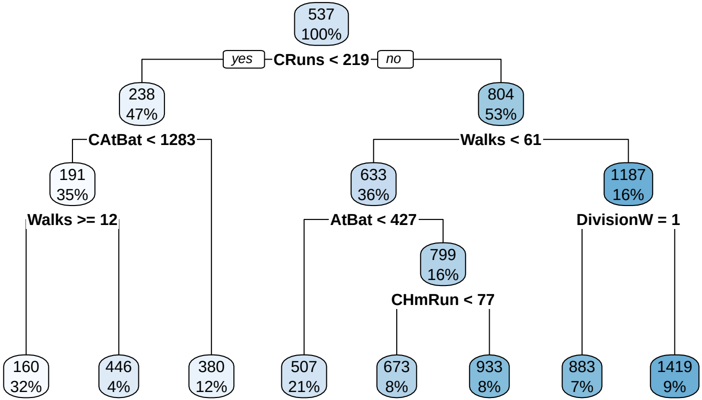
```
hit_tree$finalModel
```

```
n= 184

node), split, n, deviance, yval
      * denotes terminal node

 1) root 184 40400000  537
   2) CRuns< 218 87  5140000  238
     4) CAtBat< 1.28e+03 65  4150000  191
       8) Walks>=11.5 58   311000  160 *
       9) Walks< 11.5 7  3330000  446 *
     5) CAtBat>=1.28e+03 22   401000  380 *
   3) CRuns>=218 97 20500000  804
     6) Walks< 61 67  5700000  633
      12) AtBat< 426 38  1720000  507 *
      13) AtBat>=426 29  2570000  799
        26) CHmRun< 76.5 15   335000  673 *
        27) CHmRun>=76.5 14  1750000  933 *
```

- To plot the CV-chosen tree, we need to

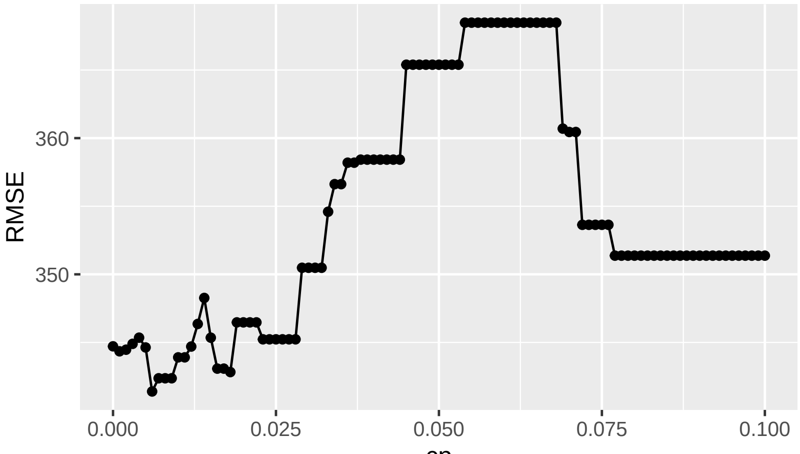
1. extract the fitted model, e.g., `hit_tree$finalModel`
2. apply a plotting function e.g., `rpart.plot()` from **rpart.plot**

```
rpart.plot(hit_tree$finalModel)
```



- Plot the performance metric against the complexity parameter

```
cp.data = data.frame("cp" = hit_tree$results[[1]],
                     "RMSE" =  hit_tree$results[[2]])
ggplot(cp.data, aes(x=cp,y=RMSE))+
  geom_point()+
  geom_line()
```



- Make predictions on the test data

```
pred.tree <- predict(hit_tree, Hitters[-train,])
mean((Hitters[-train,"Salary"] - pred.tree)^2)
```

```
[1] 95393
```

Classification Trees

- We'll use the Carseats data set from ISLR

```
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
```

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

- We change the performance metric to `ROC`, which is simply the `AUC`.

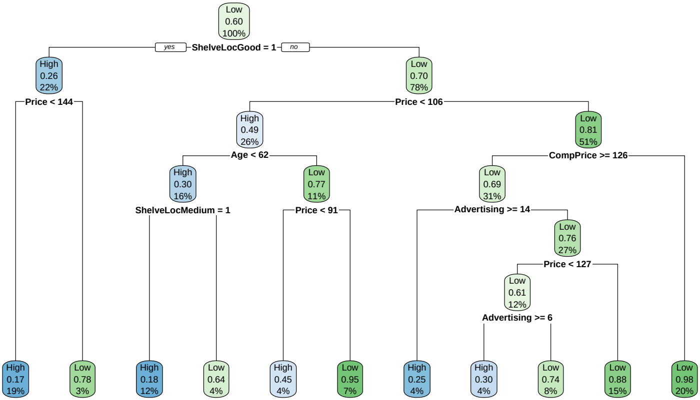
```
sales_tree = train(
Sales ~ .,
data = Carseats[train,],
method = "rpart",
trControl = trainControl("repeatedcv", number = 10,
                         repeats = 3,
                         summaryFunction = twoClassSummary,
                         classProbs = TRUE),
metric = "ROC",
# tuneLength=20
tuneGrid = data.frame(cp = seq(0, 0.2, by = 0.005))
)
```

```
sales_tree
```

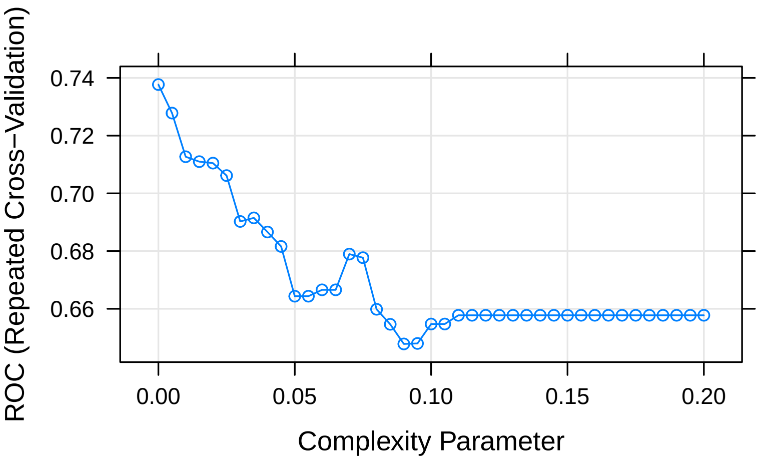
```
CART

280 samples
 10 predictor
  2 classes: 'High', 'Low'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 252, 252, 251, 252, 252, 253, ...
Resampling results across tuning parameters:

  cp     ROC    Sens   Spec
  0.000  0.738  0.602  0.746
  0.005  0.728  0.587  0.750
  0.010  0.713  0.578  0.758
  0.015  0.711  0.572  0.764
  0.020  0.710  0.572  0.765
  0.025  0.706  0.563  0.767
```

- The final model

```
rpart.plot(sales_tree$finalModel)
```



```
plot(sales_tree)
```



- Make predictions

```
sales_pred <- data.frame("p_hat"=predict(sales_tree,
                          Carseats[-train,],type = "prob")$
                        "predicted"=predict(sales_tree,
                          Carseats[-train,], type = "raw"),
                         "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   36  11
      Low    16  57
```

```
cm$overall[1]
```

```
Accuracy
   0.775
```

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

```
Sensitivity Specificity          F1
      0.692       0.838       0.727
```

- ROC curve

```
pred = prediction(sales_pred$p_hat, sales_pred$actual,
                  label.ordering = c("Low","High"))
roc = performance(pred,"tpr","fpr")
plot(roc, colorize = T, lwd = 2)
abline(a = 0, b = 1)
auc = performance(pred, measure = "auc")
subtitle = sprintf("AUC: %f", auc@y.values)
mtext(side=3,line=1,at=0,adj=0,cex=0.7,subtitle)
```