# Classification Trees

## Amanda Harrison

library(tidyverse)

## ── Attaching packages ──────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.1.0 ✔ purrr 0.2.5  
## ✔ tibble 1.4.2 ✔ dplyr 0.7.7  
## ✔ tidyr 0.8.2 ✔ stringr 1.3.1  
## ✔ readr 1.1.1 ✔ forcats 0.3.0

## ── Conflicts ─────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(rpart)  
library(rattle)

## Rattle: A free graphical interface for data science with R.  
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

library(RColorBrewer)

library(readr)  
parole=read\_csv("parole.csv")

## Parsed with column specification:  
## cols(  
## male = col\_integer(),  
## race = col\_integer(),  
## age = col\_double(),  
## state = col\_integer(),  
## time.served = col\_double(),  
## max.sentence = col\_integer(),  
## multiple.offenses = col\_integer(),  
## crime = col\_integer(),  
## violator = col\_integer()  
## )

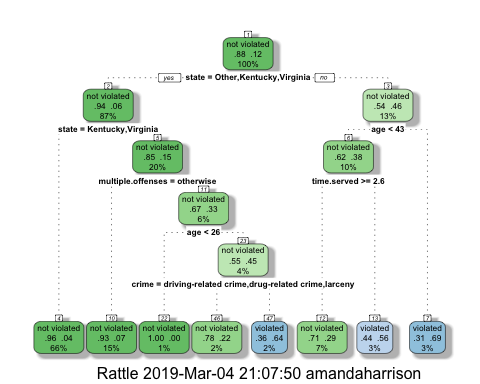
View(parole)  
parole = parole %>% mutate(male = as\_factor(as.character(male)))%>%mutate(male=fct\_recode(male,"Male"="1","Female"="0"))  
parole = parole %>% mutate(race = as\_factor(as.character(race)))%>%mutate(race=fct\_recode(race,"white"="1","otherwise"="2"))  
parole = parole %>% mutate(state = as\_factor(as.character(state)))%>%mutate(state=fct\_recode(state,"Kentucky"="2","Louisiana"="3","Virginia"="4","Other"="1"))  
parole = parole %>% mutate(crime = as\_factor(as.character(crime)))%>%mutate(crime=fct\_recode(crime,"larceny"="2","drug-related crime"="3","driving-related crime"="4","other"="1"))  
parole = parole %>% mutate(multiple.offenses = as\_factor(as.character(multiple.offenses)))%>%mutate(multiple.offenses=fct\_recode(multiple.offenses,"multiple offenses"="1","otherwise"="0"))  
parole = parole %>% mutate(violator = as\_factor(as.character(violator)))%>%mutate(violator=fct\_recode(violator,"violated"="1","not violated"="0"))

### Task 1

set.seed(12345)  
train.rows = createDataPartition(y = parole$violator, p=0.7, list = FALSE)  
paroletrain = parole[train.rows,]   
paroletest = parole[-train.rows,]

### Task 2

tree1=rpart(violator~.,paroletrain,method="class")  
fancyRpartPlot(tree1)



### Task 3

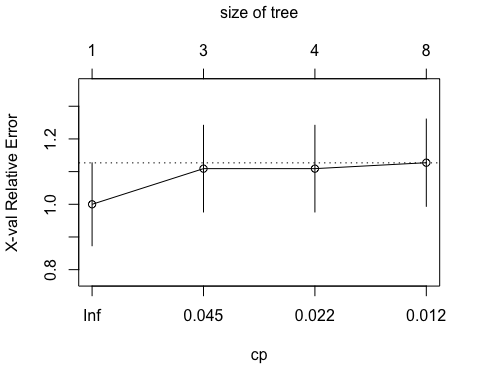
The man in the scenario is from Louisiana, therefore he is not from “other”, “Kentucky”, or “Virginia”. This would lead us to the question of whether his age was less than 43, which it is since he is 40. This would then lead us to the conclusion that he most likely violated his parole.

### Task 4

printcp(tree1)

##   
## Classification tree:  
## rpart(formula = violator ~ ., data = paroletrain, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] age crime multiple.offenses state   
## [5] time.served   
##   
## Root node error: 55/473 = 0.11628  
##   
## n= 473   
##   
## CP nsplit rel error xerror xstd  
## 1 0.054545 0 1.00000 1.0000 0.12676  
## 2 0.036364 2 0.89091 1.1091 0.13253  
## 3 0.013636 3 0.85455 1.1091 0.13253  
## 4 0.010000 7 0.80000 1.1273 0.13345

plotcp(tree1)

 The cpvalue of 0.054545 should be selected.

### Task 5

tree2=prune(tree1,cp=tree1$cptable[which.min(tree1$cptable[,"xerror"]),"CP"])

The category that is a majority class is the state in which they were from.

### Task 6

treepred = predict(tree1, paroletrain, type = "class")  
head(treepred)

## 1 2 3 4 5   
## not violated not violated not violated not violated not violated   
## 6   
## not violated   
## Levels: not violated violated

confusionMatrix(treepred, paroletrain$violator, positive = "not violated")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction not violated violated  
## not violated 402 28  
## violated 16 27  
##   
## Accuracy : 0.907   
## 95% CI : (0.8771, 0.9316)  
## No Information Rate : 0.8837   
## P-Value [Acc > NIR] : 0.06272   
##   
## Kappa : 0.5   
## Mcnemar's Test P-Value : 0.09725   
##   
## Sensitivity : 0.9617   
## Specificity : 0.4909   
## Pos Pred Value : 0.9349   
## Neg Pred Value : 0.6279   
## Prevalence : 0.8837   
## Detection Rate : 0.8499   
## Detection Prevalence : 0.9091   
## Balanced Accuracy : 0.7263   
##   
## 'Positive' Class : not violated   
##

### Task 7

treepred2 = predict(tree1, paroletest, type = "class")  
head(treepred2)

## 1 2 3 4 5   
## not violated violated not violated not violated not violated   
## 6   
## not violated   
## Levels: not violated violated

confusionMatrix(treepred2, paroletest$violator, positive = "not violated")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction not violated violated  
## not violated 170 19  
## violated 9 4  
##   
## Accuracy : 0.8614   
## 95% CI : (0.8059, 0.9059)  
## No Information Rate : 0.8861   
## P-Value [Acc > NIR] : 0.88631   
##   
## Kappa : 0.1525   
## Mcnemar's Test P-Value : 0.08897   
##   
## Sensitivity : 0.9497   
## Specificity : 0.1739   
## Pos Pred Value : 0.8995   
## Neg Pred Value : 0.3077   
## Prevalence : 0.8861   
## Detection Rate : 0.8416   
## Detection Prevalence : 0.9356   
## Balanced Accuracy : 0.5618   
##   
## 'Positive' Class : not violated   
##

The accuracy is a good value at 0.8614. The sensitivity is at 0.174. The specificity is relatively high at 0.950 ### Task 8

library(readr)  
blood <- read\_csv("blood.csv")

## Parsed with column specification:  
## cols(  
## Mnths\_Since\_Last = col\_integer(),  
## TotalDonations = col\_integer(),  
## Total\_Donated = col\_integer(),  
## Mnths\_Since\_First = col\_integer(),  
## DonatedMarch = col\_integer()  
## )

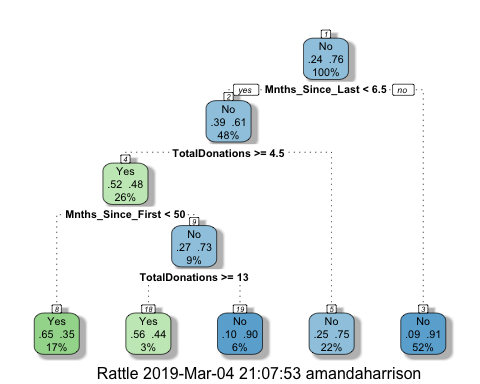
View(blood)

blood = blood %>% mutate(DonatedMarch = as\_factor(as.character(DonatedMarch)))%>%mutate(DonatedMarch=fct\_recode(DonatedMarch,"No"="0","Yes"="1"))

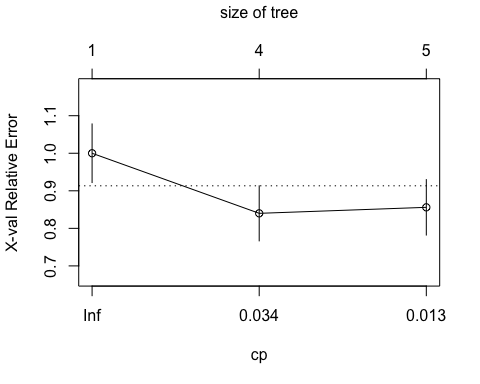
### Task 9

set.seed(1234)  
train.rows = createDataPartition(y = blood$DonatedMarch, p=0.7, list = FALSE)  
bloodtrain = blood[train.rows,]   
bloodtest = blood[-train.rows,]

blood\_tree=rpart(DonatedMarch ~.,bloodtrain,method="class")  
fancyRpartPlot(blood\_tree)



plotcp(blood\_tree)



printcp(blood\_tree)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = bloodtrain, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] Mnths\_Since\_First Mnths\_Since\_Last TotalDonations   
##   
## Root node error: 125/524 = 0.23855  
##   
## n= 524   
##   
## CP nsplit rel error xerror xstd  
## 1 0.072 0 1.000 1.000 0.078049  
## 2 0.016 3 0.784 0.840 0.073304  
## 3 0.010 4 0.768 0.856 0.073822

The cpvalue that seems to be best is 0.016

### Task 10

blood\_tree2=prune(blood\_tree,cp=blood\_tree$cptable[which.min(blood\_tree$cptable[,"xerror"]),"CP"])  
bloodpred =predict(blood\_tree2, type = "class")  
confusionMatrix(bloodpred, bloodtrain$DonatedMarch, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 58 31  
## No 67 368  
##   
## Accuracy : 0.813   
## 95% CI : (0.7769, 0.8455)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 0.002713   
##   
## Kappa : 0.4287   
## Mcnemar's Test P-Value : 0.000407   
##   
## Sensitivity : 0.4640   
## Specificity : 0.9223   
## Pos Pred Value : 0.6517   
## Neg Pred Value : 0.8460   
## Prevalence : 0.2385   
## Detection Rate : 0.1107   
## Detection Prevalence : 0.1698   
## Balanced Accuracy : 0.6932   
##   
## 'Positive' Class : Yes   
##

bloodpred2 =predict(blood\_tree2,bloodtest, type = "class")  
confusionMatrix(bloodpred2, bloodtest$DonatedMarch, positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 14 16  
## No 39 155  
##   
## Accuracy : 0.7545   
## 95% CI : (0.6927, 0.8094)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.657104   
##   
## Kappa : 0.2006   
## Mcnemar's Test P-Value : 0.003012   
##   
## Sensitivity : 0.2642   
## Specificity : 0.9064   
## Pos Pred Value : 0.4667   
## Neg Pred Value : 0.7990   
## Prevalence : 0.2366   
## Detection Rate : 0.0625   
## Detection Prevalence : 0.1339   
## Balanced Accuracy : 0.5853   
##   
## 'Positive' Class : Yes   
##

The model based off of the training set has a higher accuracy, at 0.813, whereas the testing set model has an accuracy of 0.7545. The sensitivity of the training model (0.464) is about twice the amount of the testing method (0.4642). The training set specificity is closer to that of the testing set, with the training model being 0.9223 and the testing being 0.9064.