## Daniel Sanchez

## BAN 502

## Dr. Stephen Hill

### Module 4 Assignment 2

### June 8th, 2020

library(tidyverse)  
library(caret)  
library(rpart)  
library(rattle)  
library(RColorBrewer)

parole=read.csv("parole.csv")

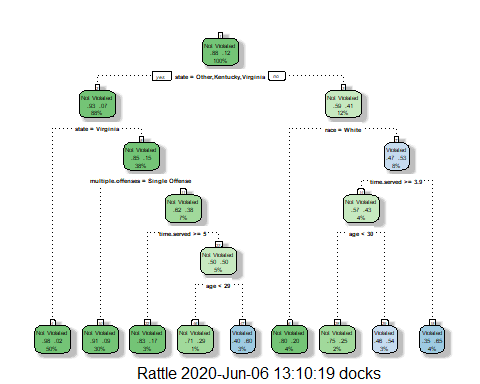
parole = parole %>% mutate(male = as\_factor(as.character(male))) %>%  
mutate(male = fct\_recode(male,  
"Female" = "0",  
"Male" = "1"))  
  
parole = parole %>% mutate(race = as\_factor(as.character(race))) %>%  
mutate(race = fct\_recode(race,  
"White" = "1",  
"Non White" = "2"))  
  
parole = parole %>% mutate(state = as\_factor(as.character(state))) %>%  
mutate(state = fct\_recode(state,  
"Other" = "1",  
"Kentucky" = "2",  
"Louisiana" = "3",  
"Virginia" = "4"))  
  
parole = parole %>% mutate(crime = as\_factor(as.character(crime))) %>%  
mutate(crime = fct\_recode(crime,  
"Other" = "1",  
"Larceny" = "2",  
"Drug-Related" = "3",  
"Driving-Related" = "4"))  
  
parole = parole %>% mutate(multiple.offenses = as\_factor(as.character(multiple.offenses))) %>%  
mutate(multiple.offenses = fct\_recode(multiple.offenses,  
"Single Offense" = "0",  
"Multiple Offenses" = "1"))  
  
parole = parole %>% mutate(violator = as\_factor(as.character(violator))) %>%  
mutate(violator = fct\_recode(violator,  
"Not Violated" = "0",  
"Violated" = "1"))

**Task 1: Split the data into training and testing sets. Your training set should have 70% of the data. Use a random number (set.seed) of 12345.**

set.seed(12345)  
train.rows = createDataPartition(y=parole$violator, p=0.7, list=FALSE)  
train = slice(parole,train.rows)  
test = slice(parole,-train.rows)

**Task 2: Create a classification tree using all of the predictor variables to predict “violator” in the training set. Plot the tree.**

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



**Task 3: For the tree created in Task 2, how would you classify a 40 year-old parolee from Louisiana who served a 5 year prison sentence? Describe how you “walk through” the classification tree to arrive at your answer.**

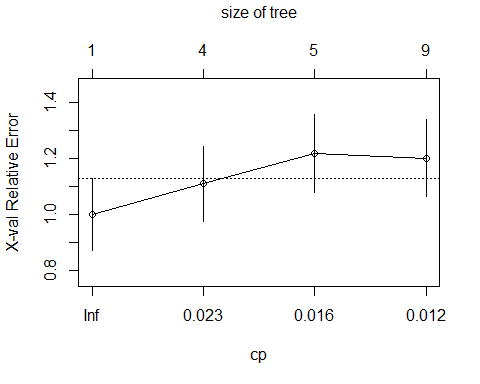
Starting the analysis with state, Louisiana would fall under “No.” Moving right, I would examine race. If the parolee’s race was White, I would project the parolee to not violate his parole. If the parolee’s race was not White, I would analyze the parolee’s time served. With this time being greater than 3.5 years, I would next consider his age. Being over the age of 30, I would project the parolee to be a parole violator.

**Task 4: Use the printcp function to evaluate tree performance as a function of the complexity parameter (cp). What cp value should be selected? Note that the printcp table tends to be a more reliable tool than the plot of cp.**

printcp(tree1)

##   
## Classification tree:  
## rpart(formula = violator ~ ., data = train, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] age multiple.offenses race state   
## [5] time.served   
##   
## Root node error: 55/473 = 0.11628  
##   
## n= 473   
##   
## CP nsplit rel error xerror xstd  
## 1 0.030303 0 1.00000 1.0000 0.12676  
## 2 0.018182 3 0.90909 1.1091 0.13253  
## 3 0.013636 4 0.89091 1.2182 0.13788  
## 4 0.010000 8 0.83636 1.2000 0.13702

plotcp(tree1)



As a function of the complexity parameter, tree performance is optimized with 3 splits. The cp value of .018 should be selected.

**Task 5: Prune the tree from Task 2 back to the cp value that you selected in Task 4. Which class (category) in the training set is the majority class (i.e., has the most observations)?**

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

## Call:  
## rpart(formula = violator ~ ., data = train, method = "class")  
## n= 473   
##   
## CP nsplit rel error xerror xstd  
## 1 0.03030303 0 1 1 0.1267582  
##   
## Node number 1: 473 observations  
## predicted class=Not Violated expected loss=0.1162791 P(node) =1  
## class counts: 418 55  
## probabilities: 0.884 0.116

The “Not violated” class is the majority class here.

**Task 6: Use the unpruned tree from Task 2 to develop predictions for the training data. Use caret’s confusionMatrix function to calculate the accuracy, specificity, and sensitivty of this tree on the training data.**

treepred = predict(tree1, train, type = "class")  
confusionMatrix(treepred, train$violator, positive="Violated")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Not Violated Violated  
## Not Violated 400 28  
## Violated 18 27  
##   
## Accuracy : 0.9027   
## 95% CI : (0.8724, 0.9279)  
## No Information Rate : 0.8837   
## P-Value [Acc > NIR] : 0.1095   
##   
## Kappa : 0.4862   
##   
## Mcnemar's Test P-Value : 0.1845   
##   
## Sensitivity : 0.49091   
## Specificity : 0.95694   
## Pos Pred Value : 0.60000   
## Neg Pred Value : 0.93458   
## Prevalence : 0.11628   
## Detection Rate : 0.05708   
## Detection Prevalence : 0.09514   
## Balanced Accuracy : 0.72392   
##   
## 'Positive' Class : Violated   
##

Accuracy is 90.27%, higher than the No Information Rate of 88.37%. Specificity is 95.69%, and Sensitivity is 49.09%.

**Task 7: Use the unpruned tree from Task 2 to develop predictions for the testing data. Use caret’s confusionMatrix function to calculate the accuracy, specificity, and sensitivty of this tree on the testing data. Comment on the quality of the model.**

treepred2 = predict(tree1, test, type = "class")  
confusionMatrix(treepred2, test$violator, positive="Violated")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Not Violated Violated  
## Not Violated 171 13  
## Violated 8 10  
##   
## Accuracy : 0.896   
## 95% CI : (0.8455, 0.9345)  
## No Information Rate : 0.8861   
## P-Value [Acc > NIR] : 0.3797   
##   
## Kappa : 0.4309   
##   
## Mcnemar's Test P-Value : 0.3827   
##   
## Sensitivity : 0.43478   
## Specificity : 0.95531   
## Pos Pred Value : 0.55556   
## Neg Pred Value : 0.92935   
## Prevalence : 0.11386   
## Detection Rate : 0.04950   
## Detection Prevalence : 0.08911   
## Balanced Accuracy : 0.69504   
##   
## 'Positive' Class : Violated   
##

The 89.6% Accuracy of this model is a slight improvement upon the 88.6% from the naive model. The Specificity is 95.5% and the Sensitivity is 43.5%.

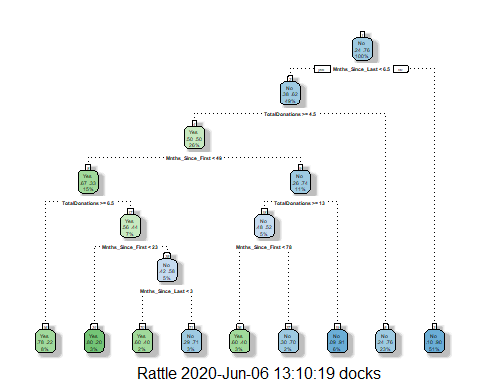
**Task 8: Read in the “Blood.csv” dataset. Convert the DonatedMarch variable to a factor and recode the variable so 0 = “No” and 1 = “Yes”.**

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

**Task 9: Split the dataset into training (70%) and testing (30%) sets. Then develop a classification tree on the training set to predict “DonatedMarch”. Evaluate the complexity parameter (cp) selection for this model.**

set.seed(1234)  
train.rows = createDataPartition(y=Blood$DonatedMarch, p=0.7, list=FALSE)  
train2 = slice(Blood,train.rows)  
test2 = slice(Blood,-train.rows)

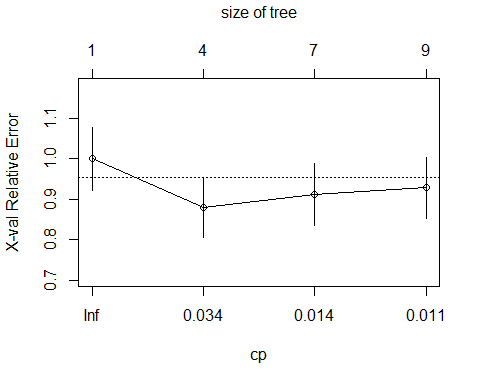
tree3 = rpart(DonatedMarch ~., train2, method="class")  
fancyRpartPlot(tree3)



printcp(tree3)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = train2, 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 0.880 0.074580  
## 3 0.012 6 0.736 0.912 0.075556  
## 4 0.010 8 0.712 0.928 0.076030

plotcp(tree3)



As a function of the complexity parameter, tree performance is optimized with three splits. In this scenario, the cp value of .016 should be selected.

**Task 10: Prune the tree back to the optimal cp value, make predictions, and use the confusionMatrix function on the both training and testing sets. Comment on the quality of the predictions.**

tree4 = prune(tree3,cp= tree3$cptable[which.min(tree3$cptable[,"xerror"]),"CP"])

treepred3 = predict(tree4, train2, type = "class")  
confusionMatrix(treepred3, train2$DonatedMarch, positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 53 26  
## No 72 373  
##   
## Accuracy : 0.813   
## 95% CI : (0.7769, 0.8455)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 0.002713   
##   
## Kappa : 0.4107   
##   
## Mcnemar's Test P-Value : 5.476e-06   
##   
## Sensitivity : 0.4240   
## Specificity : 0.9348   
## Pos Pred Value : 0.6709   
## Neg Pred Value : 0.8382   
## Prevalence : 0.2385   
## Detection Rate : 0.1011   
## Detection Prevalence : 0.1508   
## Balanced Accuracy : 0.6794   
##   
## 'Positive' Class : Yes   
##

The 81.3% Accuracy of this model is a substantial improvement upon the 76.15% No Information Rate. The Specificity is 93.48% and the Sensitivity is 42.4%.

treepred4 = predict(tree4, test2, type = "class")  
confusionMatrix(treepred4, test2$DonatedMarch, positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 18 20  
## No 35 151  
##   
## Accuracy : 0.7545   
## 95% CI : (0.6927, 0.8094)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.65710   
##   
## Kappa : 0.2468   
##   
## Mcnemar's Test P-Value : 0.05906   
##   
## Sensitivity : 0.33962   
## Specificity : 0.88304   
## Pos Pred Value : 0.47368   
## Neg Pred Value : 0.81183   
## Prevalence : 0.23661   
## Detection Rate : 0.08036   
## Detection Prevalence : 0.16964   
## Balanced Accuracy : 0.61133   
##   
## 'Positive' Class : Yes   
##

Predictions on the testing set actually provide a lower accuracy rate (75.45%) than the naive model (76.34%). Specificity in this case is 88.3% and Sensitivity is 33.96%.