# Module 4 - Classification Trees

## Cooper, Sarah

library(tidyverse)

library(caret)

library(rpart)  
library(rattle)

library(RColorBrewer)

parole\_1\_ <- read\_csv("C:/Users/Sarah/Downloads/parole (1).csv")

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

parole <- parole\_1\_

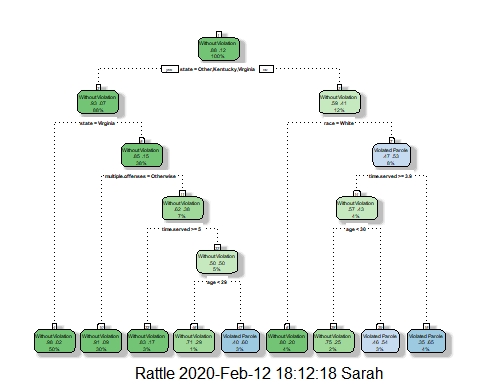
parole = parole %>% mutate(male = as\_factor(as.character(male))) %>%  
mutate(male = fct\_recode(male,  
"Male" = "1",  
"Female" = "0"))  
  
parole = parole %>% mutate(race = as\_factor(as.character(race))) %>%  
mutate(race = fct\_recode(race,  
"White" = "1",  
"Otherwise" = "2"))  
  
parole = 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" = "3",  
"Driving-related" = "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 Parole" = "1",  
"Without Violation" = "0",))

## Task 1

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

## Task 2

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



## Task 3

*If we were to try to classify a 40 year-old parolee from Louisiana who served a 5 year prison sentence, we would begin at the top of the tree and immediately move to the right. The time served was greater than 3.5 years, and the age is greater than 30. Fortunately this tree assumes that the individual at hand will not violate parole.*

## Task 4

printcp(ClassTree)

##   
## 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

*Using the printcp function we can see that with reduction of the complexity parameter, the number of partions increase. To get a healthy balance of potentially over or underfitting, we should use the complexity parameter of 0.010000.*

## Task 5

ClassTree2 = rpart(violator ~., train, cp=0.010000, method="class")  
printcp(ClassTree2)

##   
## Classification tree:  
## rpart(formula = violator ~ ., data = train, method = "class",   
## cp = 0.01)  
##   
## 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 1.2364 0.13874  
## 3 0.013636 4 0.89091 1.2727 0.14041  
## 4 0.010000 8 0.83636 1.3273 0.14286

## Task 6

TreePred = predict(ClassTree, train, type = "class")  
  
confusionMatrix(TreePred, train$violator, positive="Violated Parole")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Without Violation Violated Parole  
## Without Violation 400 28  
## Violated Parole 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 Parole   
##

*Accuracy = 90%, Sensitivity = 49%, Specificity = 96%*

## Task 7

TreePred\_Test = predict(ClassTree, test, type = "class")  
  
confusionMatrix(TreePred\_Test, test$violator, positive = "Violated Parole")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Without Violation Violated Parole  
## Without Violation 171 13  
## Violated Parole 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 Parole   
##

*Accuracy = 90%, Sensitivity = 43%, Specificity = 96%. Since the Accuracy and No Information Rate are so close I would feel confident about publishing this model. It does not appear to be overfitted.*

## Task 8

Blood <- read\_csv("C:/Users/Sarah/Downloads/Blood.csv")

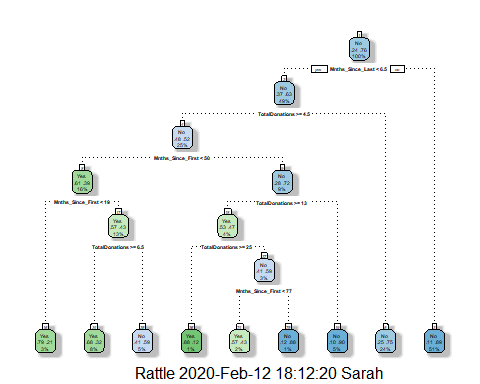
## Parsed with column specification:  
## cols(  
## Mnths\_Since\_Last = col\_double(),  
## TotalDonations = col\_double(),  
## Total\_Donated = col\_double(),  
## Mnths\_Since\_First = col\_double(),  
## DonatedMarch = col\_double()  
## )

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

## 

## Task 9

set.seed(1234)  
train.rows2 = createDataPartition(y = Blood$DonatedMarch, p=0.7, list = FALSE)  
train2 = Blood[train.rows,]  
test2 = Blood[-train.rows,]  
  
ClassTree3 = rpart(DonatedMarch ~., Blood, method="class")  
fancyRpartPlot(ClassTree3)



printcp(ClassTree3)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = Blood, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] Mnths\_Since\_First Mnths\_Since\_Last TotalDonations   
##   
## Root node error: 178/748 = 0.23797  
##   
## n= 748   
##   
## CP nsplit rel error xerror xstd  
## 1 0.046816 0 1.00000 1.00000 0.065430  
## 2 0.019663 3 0.85955 0.93820 0.063985  
## 3 0.016854 5 0.82022 0.96067 0.064523  
## 4 0.011236 7 0.78652 0.94382 0.064121  
## 5 0.010000 8 0.77528 0.90449 0.063148

*Using the printcp function we can see that with reduction of the complexity parameter, the number of partions increase. The second complexity parameter nose dives from the first which is a sharp contrast to our first pass at printcp in Task 4.*

## Task 10

BloodPrune = rpart(DonatedMarch ~., train2, cp=0.010000, method="class")  
printcp(BloodPrune)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = train2, method = "class",   
## cp = 0.01)  
##   
## Variables actually used in tree construction:  
## [1] Mnths\_Since\_First Mnths\_Since\_Last TotalDonations   
##   
## Root node error: 128/473 = 0.27061  
##   
## n= 473   
##   
## CP nsplit rel error xerror xstd  
## 1 0.046875 0 1.00000 1.00000 0.075487  
## 2 0.010417 4 0.75781 0.86719 0.072007  
## 3 0.010000 7 0.72656 0.86719 0.072007

BloodPredTrain = predict(BloodPrune, train2, type = "class")  
confusionMatrix(BloodPredTrain, train2$DonatedMarch, positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 61 26  
## No 67 319  
##   
## Accuracy : 0.8034   
## 95% CI : (0.7647, 0.8383)  
## No Information Rate : 0.7294   
## P-Value [Acc > NIR] : 0.0001165   
##   
## Kappa : 0.4461   
##   
## Mcnemar's Test P-Value : 3.357e-05   
##   
## Sensitivity : 0.4766   
## Specificity : 0.9246   
## Pos Pred Value : 0.7011   
## Neg Pred Value : 0.8264   
## Prevalence : 0.2706   
## Detection Rate : 0.1290   
## Detection Prevalence : 0.1839   
## Balanced Accuracy : 0.7006   
##   
## 'Positive' Class : Yes   
##

*The accuracy rate of 80% for this training model is concerning. An even lower rate of no information at 73% raises the red flag higher.*

BloodPredTest = predict(BloodPrune, test2, type = "class")  
confusionMatrix(BloodPredTest, test2$DonatedMarch, positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 16 23  
## No 34 202  
##   
## Accuracy : 0.7927   
## 95% CI : (0.74, 0.8391)  
## No Information Rate : 0.8182   
## P-Value [Acc > NIR] : 0.8784   
##   
## Kappa : 0.2382   
##   
## Mcnemar's Test P-Value : 0.1853   
##   
## Sensitivity : 0.32000   
## Specificity : 0.89778   
## Pos Pred Value : 0.41026   
## Neg Pred Value : 0.85593   
## Prevalence : 0.18182   
## Detection Rate : 0.05818   
## Detection Prevalence : 0.14182   
## Balanced Accuracy : 0.60889   
##   
## 'Positive' Class : Yes   
##

*The testing set is very similiar in that the accuracy is 79% and no information rate is 82%. All in all there is likely a better way to test this dataset that would lead to a higher confidence rate in predictions.*