## Stuart Broach

## Classification Trees

#install.packages("rpart")  
#install.packages("RColorBrewer")  
#install.packages("rattle")  
#install.packages("caret")  
#install.packages('e1071', dependencies=TRUE)  
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.5.2

## -- Attaching packages -------------------------------------------------------------------------------- tidyverse 1.2.1 --

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

## -- Conflicts ----------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(caret)

## Warning: package 'caret' was built under R version 3.5.2

## Loading required package: lattice

##   
## Attaching package: 'caret'

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

library(rpart)

## Warning: package 'rpart' was built under R version 3.5.2

library(RColorBrewer)

## Warning: package 'RColorBrewer' was built under R version 3.5.2

library(rattle)

## Warning: package 'rattle' was built under R version 3.5.2

## 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.

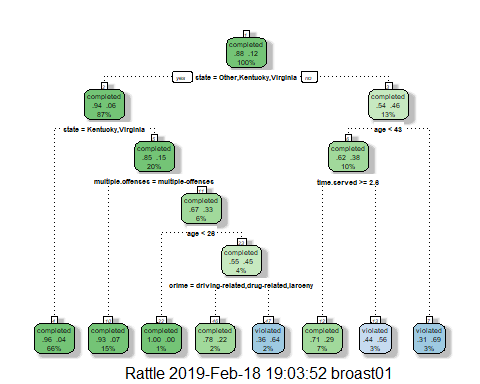
parole = read\_csv("parole.csv")

## 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()  
## )

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" = "0",  
"single-offense" = "1"))  
parole = parole %>% mutate(violator = as\_factor(as.character(violator))) %>%  
mutate(violator = fct\_recode(violator,  
"violated" = "1",  
"completed" = "0"))

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

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

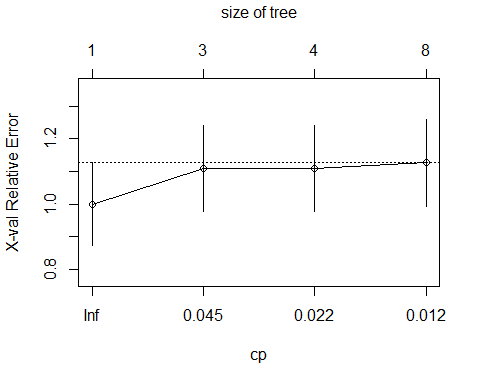


I would classify a 40-year ikd parolee from Louisiana who served a 5 year prison sentence as someone who will likely not violate their parole. I arrived at this answer by starting at the top of the tree and working my way down. I ask the question of the variable for each node. I went to the right for “no” and to the left for “yes”. He wasn’t from Other,Kentucky, or Virginia, he’s less than 43 years og age, he servered more than 2.6 years, so its a 71% chance he will complete his parole.

printcp(tree1)

##   
## Classification tree:  
## rpart(formula = violator ~ ., data = train, 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 cp value that should be selected is .036.

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

“Completed” is the majority class in the training set.

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

## 1 2 3 4 5 6   
## completed completed completed completed completed completed   
## Levels: completed violated

confusionMatrix(treepred,train$violator, positive = "completed")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction completed violated  
## completed 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 : completed   
##

treepred\_test = predict(tree1, newdata = test, type = "class")  
head(treepred\_test)

## 1 2 3 4 5 6   
## completed violated completed completed completed completed   
## Levels: completed violated

confusionMatrix(treepred\_test,test$violator, positive = "completed")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction completed violated  
## completed 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 : completed   
##

The train model had a .02 increaae in accuravy from the niave rate but the test model actually dropped .02%. Both are still highly accurate at 86-90 percent.

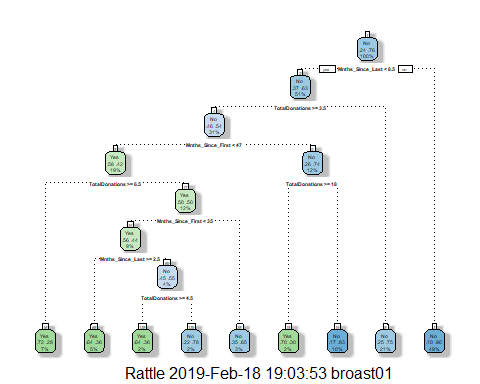
blood = read\_csv("Blood.csv")

## 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()  
## )

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

set.seed(12345)  
train.rows = createDataPartition(y = blood$DonatedMarch, p=0.7, list = FALSE)  
train2 = blood[train.rows,]  
test2 = blood[-train.rows,]

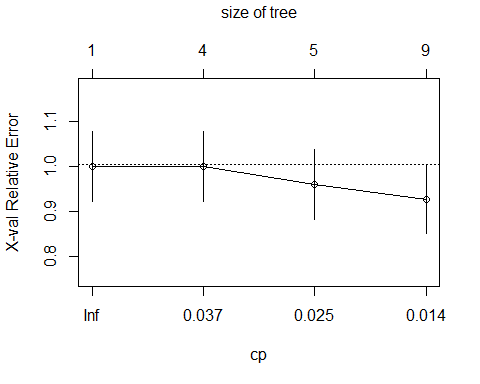
tree3 = rpart(DonatedMarch ~., train2, method="class")  
fancyRpartPlot(tree3)



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 0.042667 0 1.000 1.000 0.078049  
## 2 0.032000 3 0.872 1.000 0.078049  
## 3 0.020000 4 0.840 0.960 0.076949  
## 4 0.010000 8 0.760 0.928 0.076030

plotcp(tree3)



0.01 is the best cp selection for this model.

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

treepred2 = predict(tree3, train2, type = "class")  
head(treepred2)

## 1 2 3 4 5 6   
## Yes Yes Yes Yes Yes Yes   
## Levels: Yes No

confusionMatrix(treepred2,train2$DonatedMarch, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 56 26  
## No 69 373  
##   
## Accuracy : 0.8187   
## 95% CI : (0.783, 0.8508)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 0.0009448   
##   
## Kappa : 0.4341   
## Mcnemar's Test P-Value : 1.639e-05   
##   
## Sensitivity : 0.4480   
## Specificity : 0.9348   
## Pos Pred Value : 0.6829   
## Neg Pred Value : 0.8439   
## Prevalence : 0.2385   
## Detection Rate : 0.1069   
## Detection Prevalence : 0.1565   
## Balanced Accuracy : 0.6914   
##   
## 'Positive' Class : Yes   
##

treepred3 = predict(tree3, test2, type = "class")  
head(treepred3)

## 1 2 3 4 5 6   
## Yes Yes Yes Yes Yes No   
## Levels: Yes No

confusionMatrix(treepred3,test2$DonatedMarch, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 20 16  
## No 33 155  
##   
## Accuracy : 0.7812   
## 95% CI : (0.7213, 0.8336)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.29446   
##   
## Kappa : 0.3191   
## Mcnemar's Test P-Value : 0.02227   
##   
## Sensitivity : 0.37736   
## Specificity : 0.90643   
## Pos Pred Value : 0.55556   
## Neg Pred Value : 0.82447   
## Prevalence : 0.23661   
## Detection Rate : 0.08929   
## Detection Prevalence : 0.16071   
## Balanced Accuracy : 0.64190   
##   
## 'Positive' Class : Yes   
##

Both trai2 and test2 models improvaed on accuracy from the No Information Rate with and increase in .05 and 0.2 respectively. train2 has a really low p-value while test2 had a high p-value of .29. The train model seems to be the best of the two.