Assignment2

Jordan Whitaker

6/6/2020

options(tidyverse.quiet = TRUE)  
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.6.3

## Warning: package 'ggplot2' was built under R version 3.6.3

## Warning: package 'tidyr' was built under R version 3.6.3

## Warning: package 'readr' was built under R version 3.6.3

## Warning: package 'dplyr' was built under R version 3.6.3

## Warning: package 'stringr' was built under R version 3.6.3

library(caret)

## Warning: package 'caret' was built under R version 3.6.3

## 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 3.6.3

library(rattle)

## Warning: package 'rattle' was built under R version 3.6.3

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.  
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

library(RColorBrewer)

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

head(parole)

## # A tibble: 6 x 9  
## male race age state time.served max.sentence multiple.offens~ crime  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1 1 33.2 1 5.5 18 0 4  
## 2 0 1 39.7 1 5.4 12 0 3  
## 3 1 2 29.5 1 5.6 12 0 3  
## 4 1 1 22.4 1 5.7 18 0 1  
## 5 1 2 21.6 1 5.4 12 0 1  
## 6 1 2 46.7 1 6 18 0 4  
## # ... with 1 more variable: violator <dbl>

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",  
"other" = "2"))  
  
parole = parole %>% mutate(state = as\_factor(as.character(state))) %>%  
mutate(state = fct\_recode(state,  
"Other" = "1",  
"Kentucky" = "2",  
"Lousiana" = "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(violator = as\_factor(as.character(violator))) %>%  
mutate(violator = fct\_recode(violator,  
"no-violation" = "0",  
"violated-parole" = "1"))  
  
summary(parole)

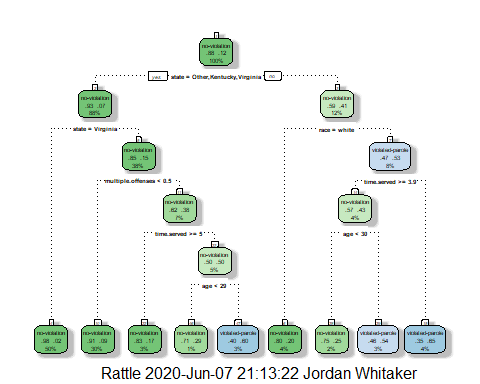
## male race age state time.served   
## male :545 white:389 Min. :18.40 Other :143 Min. :0.000   
## female:130 other:286 1st Qu.:25.35 Kentucky:120 1st Qu.:3.250   
## Median :33.70 Lousiana: 82 Median :4.400   
## Mean :34.51 Virginia:330 Mean :4.198   
## 3rd Qu.:42.55 3rd Qu.:5.200   
## Max. :67.00 Max. :6.000   
## max.sentence multiple.offenses crime violator   
## Min. : 1.00 Min. :0.0000 driving-related:101 no-violation :597   
## 1st Qu.:12.00 1st Qu.:0.0000 drug-related :153 violated-parole: 78   
## Median :12.00 Median :1.0000 Other :315   
## Mean :13.06 Mean :0.5363 larceny :106   
## 3rd Qu.:15.00 3rd Qu.:1.0000   
## Max. :18.00 Max. :1.0000

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

head(parole)

## # A tibble: 6 x 9  
## male race age state time.served max.sentence multiple.offens~ crime  
## <fct> <fct> <dbl> <fct> <dbl> <dbl> <dbl> <fct>  
## 1 male white 33.2 Other 5.5 18 0 driv~  
## 2 fema~ white 39.7 Other 5.4 12 0 drug~  
## 3 male other 29.5 Other 5.6 12 0 drug~  
## 4 male white 22.4 Other 5.7 18 0 Other  
## 5 male other 21.6 Other 5.4 12 0 Other  
## 6 male other 46.7 Other 6 18 0 driv~  
## # ... with 1 more variable: violator <fct>

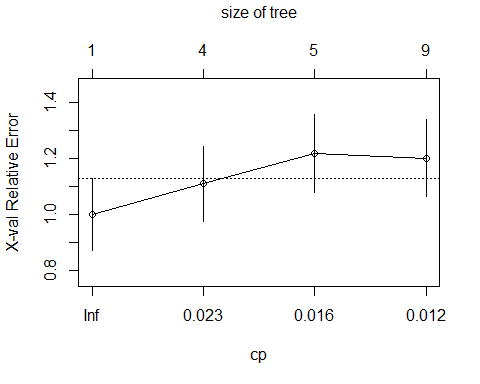
tree1 = rpart(violator ~., train, method="class")  
fancyRpartPlot(tree1)

 Task 4: A 40 year old parolee from Louisiana who served a 5 year prison sentence would be classified as “no-violation”. At the top of the tree, the first “decision point” is based on state. If state =Other, Kentucky, Virginia then you go to the left, otherwise, for a state like Louisiana, you go right “no”. There is then a decision point based on race, since no race is given in the profile, I am going to stop there and classify the person as no-violation.

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)



Task 4: Based on the diagnostic tool above, the ideal CP to reduce error is .030303

tree2 = rpart(violator ~., train, cp=.030303, method="class")  
summary(train)

## male race age state time.served   
## male :375 white:271 Min. :18.40 Other : 95 Min. :0.000   
## female: 98 other:202 1st Qu.:25.10 Kentucky: 83 1st Qu.:3.300   
## Median :33.50 Lousiana: 58 Median :4.400   
## Mean :34.15 Virginia:237 Mean :4.185   
## 3rd Qu.:42.40 3rd Qu.:5.200   
## Max. :65.10 Max. :6.000   
## max.sentence multiple.offenses crime violator   
## Min. : 1.00 Min. :0.0000 driving-related: 65 no-violation :418   
## 1st Qu.:12.00 1st Qu.:0.0000 drug-related :103 violated-parole: 55   
## Median :12.00 Median :1.0000 Other :231   
## Mean :13.03 Mean :0.5518 larceny : 74   
## 3rd Qu.:15.00 3rd Qu.:1.0000   
## Max. :18.00 Max. :1.0000

Task 5: The majority class in the training set is clearly no-violation with a count of 418 versus violated-parole at 55.

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

## 1 2 3 4 5 6   
## no-violation no-violation no-violation no-violation no-violation no-violation   
## Levels: no-violation violated-parole

confusionMatrix(treepred,train$violator, positive="violated-parole")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no-violation violated-parole  
## no-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   
##

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

## 1 2 3 4 5 6   
## no-violation no-violation no-violation no-violation no-violation no-violation   
## Levels: no-violation violated-parole

confusionMatrix(treepred\_test, test$violator, positive="violated-parole")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no-violation violated-parole  
## no-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   
##

Task 7: When the model is applied to the testing data, the accuracy is .896 which is close to the accuracy of the training set (.9027). The accuracy on the testing data is also stronger than the No Information Rate metric which is .8861. With the accuracies being about the same on the training and testing data and with gains over the No Information Rate, I feel confident that this model is not being overfit.

blood = read\_csv("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()  
## )

head(blood)

## # A tibble: 6 x 5  
## Mnths\_Since\_Last TotalDonations Total\_Donated Mnths\_Since\_First DonatedMarch  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 2 50 12500 98 1  
## 2 0 13 3250 28 1  
## 3 1 16 4000 35 1  
## 4 2 20 5000 45 1  
## 5 1 24 6000 77 0  
## 6 4 4 1000 4 0

summary(blood)

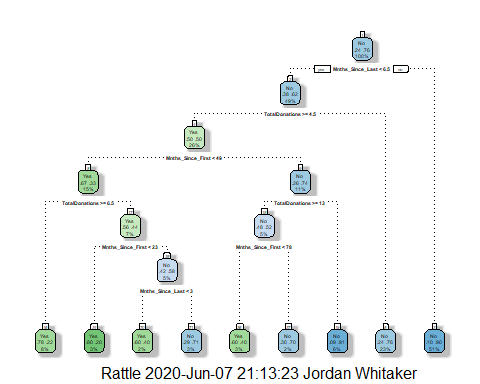
## Mnths\_Since\_Last TotalDonations Total\_Donated Mnths\_Since\_First  
## Min. : 0.000 Min. : 1.000 Min. : 250 Min. : 2.00   
## 1st Qu.: 2.750 1st Qu.: 2.000 1st Qu.: 500 1st Qu.:16.00   
## Median : 7.000 Median : 4.000 Median : 1000 Median :28.00   
## Mean : 9.507 Mean : 5.515 Mean : 1379 Mean :34.28   
## 3rd Qu.:14.000 3rd Qu.: 7.000 3rd Qu.: 1750 3rd Qu.:50.00   
## Max. :74.000 Max. :50.000 Max. :12500 Max. :98.00   
## DonatedMarch   
## Min. :0.000   
## 1st Qu.:0.000   
## Median :0.000   
## Mean :0.238   
## 3rd Qu.:0.000   
## Max. :1.000

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

set.seed(1234)  
train.rows2 = createDataPartition(y= blood$DonatedMarch, p=0.7, list= FALSE)  
  
train2 = blood[train.rows2,]  
test2 = blood[-train.rows2,]  
  
head(train2)

## # A tibble: 6 x 5  
## Mnths\_Since\_Last TotalDonations Total\_Donated Mnths\_Since\_First DonatedMarch  
## <dbl> <dbl> <dbl> <dbl> <fct>   
## 1 0 13 3250 28 Yes   
## 2 1 16 4000 35 Yes   
## 3 2 20 5000 45 Yes   
## 4 1 24 6000 77 No   
## 5 4 4 1000 4 No   
## 6 2 9 2250 22 Yes

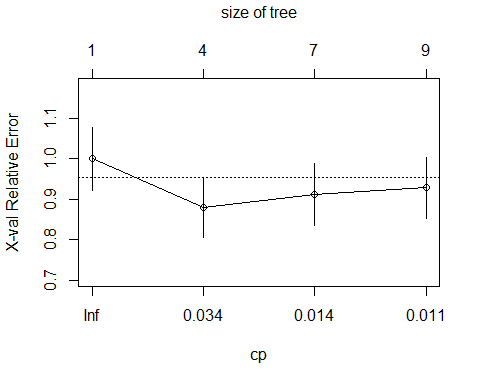
blood\_tree = rpart(DonatedMarch ~., train2, method="class")  
  
fancyRpartPlot(blood\_tree)



printcp(blood\_tree)

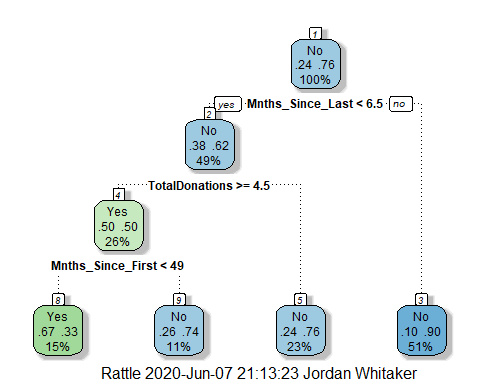
##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = train2, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] Mnths\_Since\_First Mnths\_Since\_Last TotalDonations   
##   
## Root node error: 125/524 = 0.23855  
##   
## n= 524   
##   
## CP 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(blood\_tree)



Task 9: The optimal complexity parameter seems to be 0.016 accoarding to the printcp, it clearly shows that is the parameter that will result in the lowest error. But instead of manually applying that value, I will use the snippet below to prune back to the optimal cp.

blood\_tree2 = prune(blood\_tree, cp=blood\_tree$cptable[which.min(blood\_tree$cptable[,"xerror"]), "CP"])  
  
fancyRpartPlot(blood\_tree2)



#blood\_tree2 = rpart(DonatedMarch ~., train2, cp=0.016, method="class")  
  
#Training Prediction and Confusion Matrix  
blood\_treepred = predict(blood\_tree2, train2, type="class")  
head(blood\_treepred)

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

confusionMatrix(blood\_treepred, 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   
##

blood\_treepred\_test = predict(blood\_tree2, test2, type="class")  
head(blood\_treepred\_test)

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

confusionMatrix(blood\_treepred\_test, 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   
##

When the model was applied to the testing data it had an accuracy of .7545 which unfortunately resulted in a performance drop of roughly 6%. The model also performed about a percent worse than the No Information Rate. The predictions overall were not great.