## Module 4 Assignment 1

### Hackett, Evan

library(e1071)

## Warning: package 'e1071' was built under R version 3.5.2

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(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.

library(RColorBrewer)

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

parole <- read\_csv("C:/Users/Evan/Desktop/BAN 502/Module 3/Assignment 2/parole.csv") #ReadingInParoleDataset

## 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()  
## )

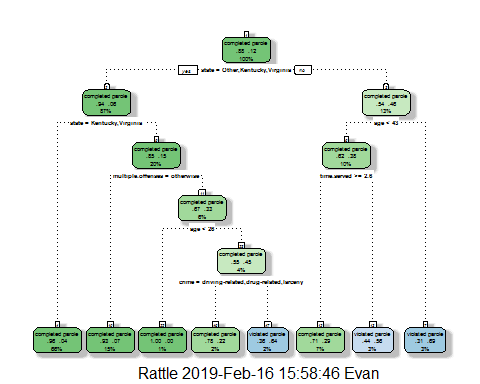
View(parole)  
  
parole = parole %>% mutate(male = as\_factor(as.character(male))) %>%  
mutate(male = fct\_recode(male,  
"male" = "1",  
"female" = "0"))  
  
parole = parole %>% mutate(race = as\_factor(as.character(race))) %>%  
mutate(race = fct\_recode(race,  
"white" = "1",  
"otherwise" = "2"))  
  
parole = parole %>% mutate(state = as\_factor(as.character(state))) %>%  
mutate(state = fct\_recode(state,  
"Other" = "1",  
"Kentucky" = "2",  
"Louisana" = "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,  
"multiple offenses" = "1",  
"otherwise" = "0"))  
  
parole = parole %>% mutate(violator = as\_factor(as.character(violator))) %>%  
mutate(violator = fct\_recode(violator,  
"violated parole" = "1",  
"completed parole" = "0"))

Splitting the dataset into Training / Testing

set.seed(12345)  
train.rows = createDataPartition(y = parole$violator, p=0.7, list = FALSE) #70% in training  
train = parole[train.rows,]   
test = parole[-train.rows,]

Creating a classification tree to predict violator within the training dataset

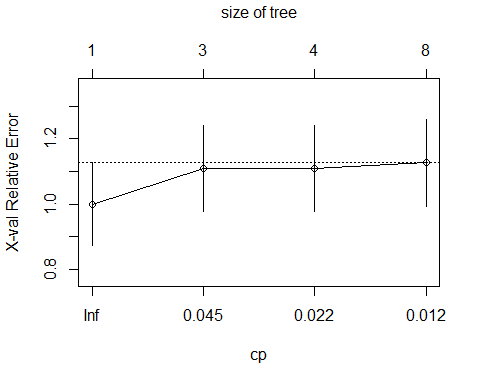
tree1 = rpart(violator ~., train, method="class")  
fancyRpartPlot(tree1)

 In looking at the classification tree, I would classify a 40 year old parolle from lousiana who served a 5 year prison status as someone who completed their parole. At the very top, we are asked if the person lives in Kentucky, Virginia or Other, knowing that they are from lousiana we immediately know that the answer is no, as Lousiana was one of the states listed within the dataset. This takes us to right where we the next classification is based off of age, if the person is older than 43 they would be classified as somoene violated parole, however our subject is 40 so we move to the next branch where if the time served is longer than 2.6 years they completed their parole, and that matches the information we are given, as they served 5 years.

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 complexity parameter that should be utilized is 0.054545.

tree2 = rpart(violator ~., train, cp=0.054545, method="class")  
summary(train)

## male race age state   
## male :384 white :269 Min. :18.40 Other : 96   
## female: 89 otherwise:204 1st Qu.:25.10 Kentucky: 75   
## Median :33.20 Louisana: 63   
## Mean :34.07 Virginia:239   
## 3rd Qu.:42.00   
## Max. :67.00   
## time.served max.sentence multiple.offenses  
## Min. :0.000 Min. : 1.00 otherwise :205   
## 1st Qu.:3.100 1st Qu.:12.00 multiple offenses:268   
## Median :4.300 Median :12.00   
## Mean :4.136 Mean :13.01   
## 3rd Qu.:5.200 3rd Qu.:15.00   
## Max. :6.000 Max. :18.00   
## crime violator   
## driving-related: 75 completed parole:418   
## drug-related :103 violated parole : 55   
## Other :219   
## larceny : 76   
##   
##

The majority class of the train dataset is completed parole. In a naive model we would assume the majority would complete parole.

Predictions on training set

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

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

Caret confusion matrix and accuracy, etc. calcs on training

confusionMatrix(treepred\_train,train$violator,positive="completed parole") #predictions first then actual

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction completed parole violated parole  
## completed parole 402 28  
## violated parole 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 parole  
##

Predictions on testing set

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

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

Caret confusion matrix and accuracy, etc. calcs on testing

confusionMatrix(treepred\_test,test$violator,positive="completed parole") #predictions first then actual

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction completed parole violated parole  
## completed parole 170 19  
## violated parole 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 parole  
##

The model is 86.14% accurate qhich is actually less accurate than what we would assume if we had no data and had a naive model. However, the diffeerence between the two is not statisically significant as the p-value is greater than .05.

Reading in Blood dataset and converting dataset

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()  
## )

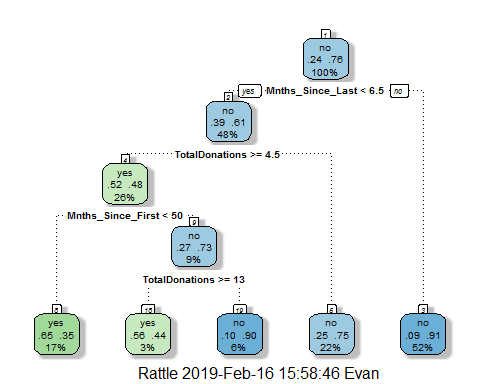
View(Blood)  
  
Blood = Blood %>% mutate(DonatedMarch = as\_factor(as.character(DonatedMarch))) %>%  
mutate(DonatedMarch = fct\_recode(DonatedMarch,  
"yes" = "1",  
"no" = "0"))

splitting Blood dataset into training/testing

set.seed(1234)  
train.rows = createDataPartition(y = Blood$DonatedMarch, p=0.7, list = FALSE) #70% in training  
train2 = Blood[train.rows,]   
test2 = Blood[-train.rows,]

Creating a classification tree to predict March Donation within the training dataset

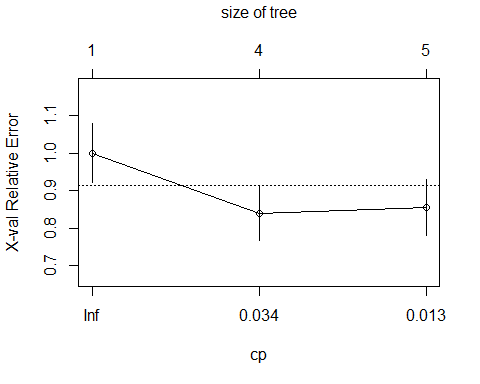
Bloodtree1 = rpart(DonatedMarch ~., train2, method="class")  
fancyRpartPlot(Bloodtree1)



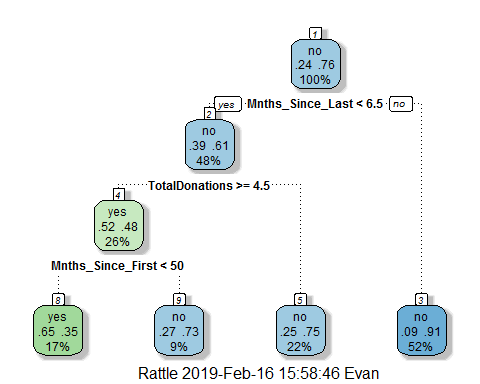
printcp(Bloodtree1)

##   
## 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 0.840 0.073304  
## 3 0.010 4 0.768 0.856 0.073822

plotcp(Bloodtree1)



Bloodtree2 = rpart(DonatedMarch ~., train2, cp=0.016, method="class")  
fancyRpartPlot(Bloodtree2)



Predictions on training set for blood data

treepred\_train2 = predict(Bloodtree2, newdata=train2, type = "class")  
head(treepred\_train2)

## 1 2 3 4 5 6   
## yes yes no no yes yes   
## Levels: yes no

Caret confusion matrix and accuracy, etc. calcs on training

confusionMatrix(treepred\_train2,train2$DonatedMarch,positive="yes") #predictions first then actual

## 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   
##

Predictions on training set for blood data

treepred\_test2 = predict(Bloodtree2, newdata=test2, type = "class")  
head(treepred\_test2)

## 1 2 3 4 5 6   
## no yes yes no no yes   
## Levels: yes no

Caret confusion matrix and accuracy, etc. calcs on training

confusionMatrix(treepred\_test2,test2$DonatedMarch,positive="yes") #predictions first then actual

## 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 accuracy is a bit higher on the training dataset at 81.3%, which is statistically significanly higer than if we had no information at all. We know this because the p-value is less than .05. The accuracy on the testing set did decrease slightly t0 75.45%, which is less than the accuracy if no information was being provided, however the variance between those two is not statistically significant as the p-value is greater than .05.