## Module 4 Assignment 1: Classification Trees

options(tidyverse.quiet=FALSE)  
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 0.3.0  
## v tibble 2.0.1 v dplyr 0.7.8  
## v tidyr 0.8.2 v stringr 1.3.1  
## v readr 1.3.1 v forcats 0.3.0

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

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

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

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

## -- 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)  
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)  
library(e1071)

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

read in dataset

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()  
## )

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()  
## )

Structure and Summary

str(parole)

## Classes 'spec\_tbl\_df', 'tbl\_df', 'tbl' and 'data.frame': 675 obs. of 9 variables:  
## $ male : num 1 0 1 1 1 1 1 0 0 1 ...  
## $ race : num 1 1 2 1 2 2 1 1 1 2 ...  
## $ age : num 33.2 39.7 29.5 22.4 21.6 46.7 31 24.6 32.6 29.1 ...  
## $ state : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ time.served : num 5.5 5.4 5.6 5.7 5.4 6 6 4.8 4.5 4.7 ...  
## $ max.sentence : num 18 12 12 18 12 18 18 12 13 12 ...  
## $ multiple.offenses: num 0 0 0 0 0 0 0 0 0 0 ...  
## $ crime : num 4 3 3 1 1 4 3 1 3 2 ...  
## $ violator : num 0 0 0 0 0 0 0 0 0 0 ...  
## - attr(\*, "spec")=  
## .. 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()  
## .. )

summary(parole)

## male race age state   
## Min. :0.0000 Min. :1.000 Min. :18.40 Min. :1.000   
## 1st Qu.:1.0000 1st Qu.:1.000 1st Qu.:25.35 1st Qu.:2.000   
## Median :1.0000 Median :1.000 Median :33.70 Median :3.000   
## Mean :0.8074 Mean :1.424 Mean :34.51 Mean :2.887   
## 3rd Qu.:1.0000 3rd Qu.:2.000 3rd Qu.:42.55 3rd Qu.:4.000   
## Max. :1.0000 Max. :2.000 Max. :67.00 Max. :4.000   
## time.served max.sentence multiple.offenses crime   
## Min. :0.000 Min. : 1.00 Min. :0.0000 Min. :1.000   
## 1st Qu.:3.250 1st Qu.:12.00 1st Qu.:0.0000 1st Qu.:1.000   
## Median :4.400 Median :12.00 Median :1.0000 Median :2.000   
## Mean :4.198 Mean :13.06 Mean :0.5363 Mean :2.059   
## 3rd Qu.:5.200 3rd Qu.:15.00 3rd Qu.:1.0000 3rd Qu.:3.000   
## Max. :6.000 Max. :18.00 Max. :1.0000 Max. :4.000   
## violator   
## Min. :0.0000   
## 1st Qu.:0.0000   
## Median :0.0000   
## Mean :0.1156   
## 3rd Qu.:0.0000   
## Max. :1.0000

Recode/Rename the variables within the Parole file

parole = parole %>% mutate(male = as.factor(male)) %>%   
 mutate(male = fct\_recode(male, "female" = "0", "male" = "1" ))   
  
parole = parole %>% mutate(race = as.factor(race)) %>%   
 mutate(race = fct\_recode(race, "white" = "1", "otherwise" = "2" ))  
  
parole = parole %>% mutate(state = as.factor(state)) %>%   
 mutate(state = fct\_recode(state, "any other state" = "1", "Kentucky" = "2", "Louisiana" = "3", "Virginia" = "4" ))  
  
parole = parole %>% mutate(multiple.offenses = as.factor(multiple.offenses)) %>%   
 mutate(multiple.offenses = fct\_recode(multiple.offenses, "otherwise" = "0", "multiple offenses" = "1" ))   
  
parole = parole %>% mutate(crime = as.factor(crime)) %>%   
 mutate(crime = fct\_recode(crime, "larceny" = "2", "drug-related" = "3", "driving-related" = "4", "other crime" = "1" ))   
  
parole = parole %>% mutate(violator = as.factor(violator)) %>%   
 mutate(violator = fct\_recode(violator, "completed parole" = "0", "violated parole" = "1" ))   
  
str(parole)

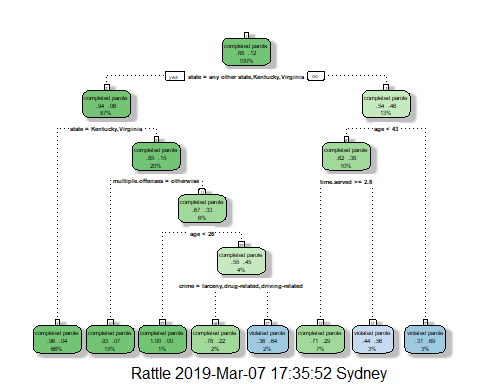
## Classes 'tbl\_df', 'tbl' and 'data.frame': 675 obs. of 9 variables:  
## $ male : Factor w/ 2 levels "female","male": 2 1 2 2 2 2 2 1 1 2 ...  
## $ race : Factor w/ 2 levels "white","otherwise": 1 1 2 1 2 2 1 1 1 2 ...  
## $ age : num 33.2 39.7 29.5 22.4 21.6 46.7 31 24.6 32.6 29.1 ...  
## $ state : Factor w/ 4 levels "any other state",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ time.served : num 5.5 5.4 5.6 5.7 5.4 6 6 4.8 4.5 4.7 ...  
## $ max.sentence : num 18 12 12 18 12 18 18 12 13 12 ...  
## $ multiple.offenses: Factor w/ 2 levels "otherwise","multiple offenses": 1 1 1 1 1 1 1 1 1 1 ...  
## $ crime : Factor w/ 4 levels "other crime",..: 4 3 3 1 1 4 3 1 3 2 ...  
## $ violator : Factor w/ 2 levels "completed parole",..: 1 1 1 1 1 1 1 1 1 1 ...

Task 1: Split the data into training sets. Training set should represent 70% of the data and use a set.seed of 12345

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,]

Task 2: create a classification tree to predict “violator” within the training set

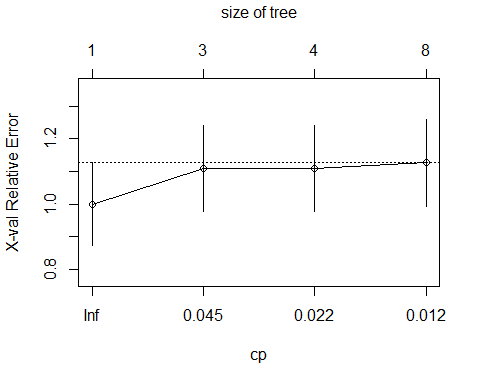
tree1 = rpart(violator ~., train, method = "class")  
fancyRpartPlot(tree1)



Task 3: How would you classify a 40-year old parolee from Louisiana who served a 5 year prison.

Task 4: Use the plotcp and printcp functions to evaluate tree performance as a function of the complexity parameter (cp).

plotcp(tree1)



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

After looking at the results of the classification tree, the optimal CP value is 0.54545.

Task 5: Prune the tree from task 2, using the CP value from task 4.

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

##   
## completed parole violated parole   
## 418 55

The majority class witin the dataset is “Completed parole” with 418 observations.

Task 6: Use caret’s confusion Matrix function to calculate the accuracy, specificity, and sensitivity of this tree on the testing data.

treeprediction = predict(tree1, type = "class")  
confusionMatrix(treeprediction, train$violator, positive = "completed parole")

## 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  
##

Accuracy = 0.907 Sensitivity = 0.49091 Specificity = 0.96172

Task 7:

treepred\_test = predict(tree1, newdata = test, type = "class")  
confusionMatrix(treepred\_test, test$violator, positive = "completed parole")

## 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  
##

Accuracy = 0.8614 Sensitivity = 0.9497 Specificity = 0.1739

Task 8: Read-in blood dataset

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()  
## )

str(blood)

## Classes 'spec\_tbl\_df', 'tbl\_df', 'tbl' and 'data.frame': 748 obs. of 5 variables:  
## $ Mnths\_Since\_Last : num 2 0 1 2 1 4 2 1 2 5 ...  
## $ TotalDonations : num 50 13 16 20 24 4 7 12 9 46 ...  
## $ Total\_Donated : num 12500 3250 4000 5000 6000 1000 1750 3000 2250 11500 ...  
## $ Mnths\_Since\_First: num 98 28 35 45 77 4 14 35 22 98 ...  
## $ DonatedMarch : num 1 1 1 1 0 0 1 0 1 1 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. Mnths\_Since\_Last = col\_double(),  
## .. TotalDonations = col\_double(),  
## .. Total\_Donated = col\_double(),  
## .. Mnths\_Since\_First = col\_double(),  
## .. DonatedMarch = col\_double()  
## .. )

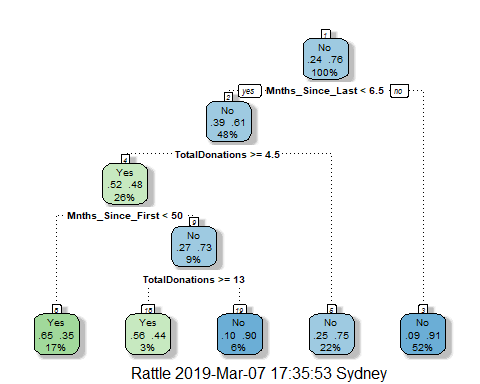
blood = blood %>% mutate(DonatedMarch = as\_factor(as.character(DonatedMarch))) %>%  
mutate(DonatedMarch = fct\_recode(DonatedMarch, "Yes" = "1", "No" = "0"))  
str(blood)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 748 obs. of 5 variables:  
## $ Mnths\_Since\_Last : num 2 0 1 2 1 4 2 1 2 5 ...  
## $ TotalDonations : num 50 13 16 20 24 4 7 12 9 46 ...  
## $ Total\_Donated : num 12500 3250 4000 5000 6000 1000 1750 3000 2250 11500 ...  
## $ Mnths\_Since\_First: num 98 28 35 45 77 4 14 35 22 98 ...  
## $ DonatedMarch : Factor w/ 2 levels "Yes","No": 1 1 1 1 2 2 1 2 1 1 ...

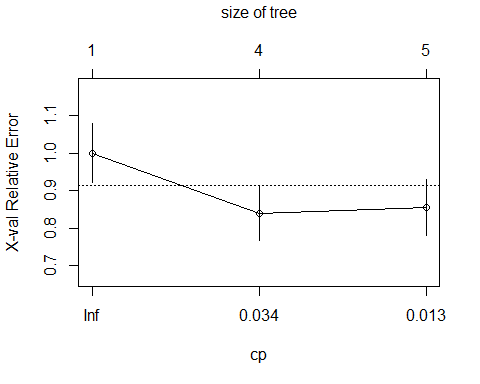
Task 9: Split the dataset into traing (70%) and testing (30%) sets.

set.seed(1234)  
train.rows = createDataPartition(y = blood$DonatedMarch, p=0.7, list =FALSE)  
trainBlood = blood[train.rows,]  
testBlood = blood[-train.rows,]

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



plotcp(tree3)



printcp(tree3)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = trainBlood, 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

The Optimal CP value is 0.016.

Task 10: prune the tree back to the optimal cp value

tree4 = prune(tree1, cp= tree3$cptable[which.min(tree3$cptable[,"xerror"]),"CP"])  
tree4\_train = predict(tree4, type = "class")  
confusionMatrix(tree4\_train, train$violator, positive = "completed parole")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction completed parole violated parole  
## completed parole 406 35  
## violated parole 12 20  
##   
## Accuracy : 0.9006   
## 95% CI : (0.8701, 0.9261)  
## No Information Rate : 0.8837   
## P-Value [Acc > NIR] : 0.140205   
##   
## Kappa : 0.4092   
## Mcnemar's Test P-Value : 0.001332   
##   
## Sensitivity : 0.9713   
## Specificity : 0.3636   
## Pos Pred Value : 0.9206   
## Neg Pred Value : 0.6250   
## Prevalence : 0.8837   
## Detection Rate : 0.8584   
## Detection Prevalence : 0.9323   
## Balanced Accuracy : 0.6675   
##   
## 'Positive' Class : completed parole  
##

tree4\_test = predict(tree4, newdata = test, type = "class")  
confusionMatrix(tree4\_test, test$violator, positive = "completed parole")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction completed parole violated parole  
## completed parole 174 20  
## violated parole 5 3  
##   
## Accuracy : 0.8762   
## 95% CI : (0.8227, 0.9183)  
## No Information Rate : 0.8861   
## P-Value [Acc > NIR] : 0.71682   
##   
## Kappa : 0.1432   
## Mcnemar's Test P-Value : 0.00511   
##   
## Sensitivity : 0.9721   
## Specificity : 0.1304   
## Pos Pred Value : 0.8969   
## Neg Pred Value : 0.3750   
## Prevalence : 0.8861   
## Detection Rate : 0.8614   
## Detection Prevalence : 0.9604   
## Balanced Accuracy : 0.5513   
##   
## 'Positive' Class : completed parole  
##