## Fields, Alex

## BAN 502

### Module 4 - Assignment 2

#### Task 3 - Q&A

For the tree created in Task 2, how would you classify a 40 year-old parolee from Louisiana who served a 5 year prison sentence? Describe how you “walk through” the classification tree to arrive at your answer. **I Would look at the top with 100% and see what the baseline to move either left (true) or right (false). My tree starts by seeing what state you lived/served in and being from Louisiana I would go to the right. I would look at my age. Being 40 I would go to the left. Knowing I served 5 years I would move to the left and it states that I would have violated parole.**

#### Task 4 - Q&A

Using the *printcp()* function I was able to see the splits move to 7 at a cp value of 0.01. When looking at anything not maxing out the cp value is at 3 splits with a cp value of 0.136 which seems to be the optimal one.

#### Task 5 - Q&A

age, crime, multiple.offenses state, time.served variables were used. *State* has the variable of importance and seems to be used to most.

### Task 6 - Q&A

Using the confusion matrix on the training data I get Sensitivity of 0.9617, Specificity of 0.4909 and Accuracy of 0.907. This is a pretty decent percentage for the model.

### Task 7 - Q&A

Using the confusion matrix on the testing data I get Sensitivity of 1.0000, Specificity of 0.0000 and Accuracy of 0.886. This seems to be less accurate of a model than the training data. The model has a Kappa of 0 which is not good. Since this measures how much better the classier is, compared to guessing with the target distribution, we can see that the model can be better.

### Task 9 - Q&A

For the Blood Dataset, the optimal CP for the training set was 0.016 while the test set was 0.012579. The CP for the testing set seemed to be a more precise model since the testing had 5 splits while the training had 4. The error on the testing set was more accurate compared to the training set by ~0.18.

### Task 10 - Q&A

While pruning the tree to the optimal CP value of 0.126, you can see that the Accuracy is 0.817, Specificity is 0.9240, Sensitivity is 0.4717, and this has a balanced Accuracy of 0.6978. We can also see the Kappa value is close to 0.5.

library(readr)  
parole <- read\_csv("parole.csv")  
parole = as\_tibble(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",  
"other" = "2"))  
  
parole = parole %>% mutate(state = as\_factor(as.character(state))) %>%  
mutate(state = fct\_recode(state,  
"other" = "1",  
"Kentucky" = "2",  
"Louisiana" = "3",  
"Virginia" = "4"))  
  
parole = parole %>% mutate(crime = as\_factor(as.character(crime))) %>%  
mutate(crime = fct\_recode(crime,  
"other" = "1",  
"larceny" = "2",  
"drug-related crime" = "3",  
"driving-related crime" = "4"))  
  
parole = parole %>% mutate(violator = as\_factor(as.character(violator))) %>%  
mutate(violator = fct\_recode(violator,  
"Violated Parole" = "0",  
"No Parole Violation" = "1"))  
  
parole = parole %>% drop\_na() #drops N/A's  
str(parole)

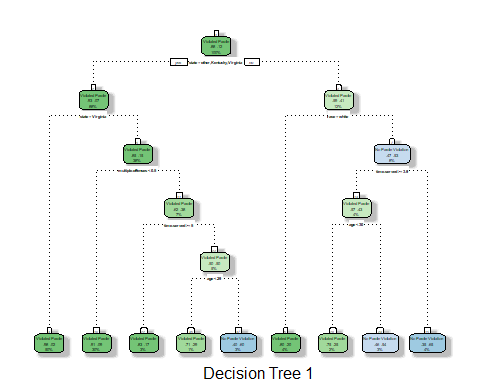
## tibble [675 x 9] (S3: tbl\_df/tbl/data.frame)  
## $ male : Factor w/ 2 levels "male","female": 1 2 1 1 1 1 1 2 2 1 ...  
## $ race : Factor w/ 2 levels "white","other": 1 1 2 1 2 2 1 1 1 2 ...  
## $ age : num [1:675] 33.2 39.7 29.5 22.4 21.6 46.7 31 24.6 32.6 29.1 ...  
## $ state : Factor w/ 4 levels "other","Kentucky",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ time.served : num [1:675] 5.5 5.4 5.6 5.7 5.4 6 6 4.8 4.5 4.7 ...  
## $ max.sentence : num [1:675] 18 12 12 18 12 18 18 12 13 12 ...  
## $ multiple.offenses: num [1:675] 0 0 0 0 0 0 0 0 0 0 ...  
## $ crime : Factor w/ 4 levels "driving-related crime",..: 1 2 2 3 3 1 2 3 2 4 ...  
## $ violator : Factor w/ 2 levels "Violated Parole",..: 1 1 1 1 1 1 1 1 1 1 ...

### Training Data

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

### Tree training plot 1

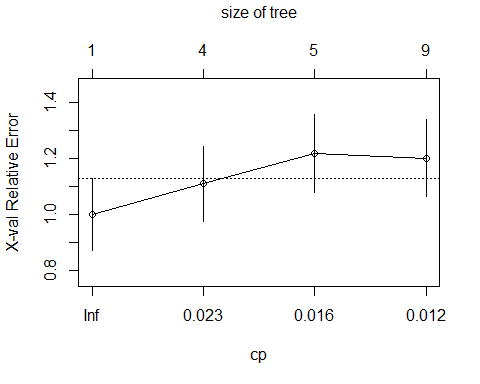
tree1 = rpart(violator ~., train, method = "class")  
fancyRpartPlot(tree1, sub = "Decision Tree 1")



#Showing accuracy of tree and prevents from a too complex tree  
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)



### CP value change

tree2 = rpart(violator ~., train, cp=.013636, method="class")  
#fancyRpartPlot(tree2, sub = "Decision Tree 1")  
printcp(tree2)

##   
## Classification tree:  
## rpart(formula = violator ~ ., data = train, method = "class",   
## cp = 0.013636)  
##   
## 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 0.013636 8 0.83636 1.2727 0.14041

### Tree predicting

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

## 1 2 3 4 5   
## Violated Parole Violated Parole Violated Parole Violated Parole Violated Parole   
## 6   
## Violated Parole   
## Levels: Violated Parole No Parole Violation

### Confusion Matrix

confusionMatrix(treepred,train$violator,positive="Violated Parole") #predictions first then actual

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Violated Parole No Parole Violation  
## Violated Parole 400 28  
## No Parole Violation 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 : 0.9569   
## Specificity : 0.4909   
## Pos Pred Value : 0.9346   
## Neg Pred Value : 0.6000   
## Prevalence : 0.8837   
## Detection Rate : 0.8457   
## Detection Prevalence : 0.9049   
## Balanced Accuracy : 0.7239   
##   
## 'Positive' Class : Violated Parole   
##

### Testing Confusion Matrix

tree2 = rpart(violator ~., test, method = "class")  
treepred = predict(tree2, test, type = "class")  
confusionMatrix(treepred,test$violator,positive="Violated Parole") #predictions first then actual

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Violated Parole No Parole Violation  
## Violated Parole 175 13  
## No Parole Violation 4 10  
##   
## Accuracy : 0.9158   
## 95% CI : (0.8687, 0.9502)  
## No Information Rate : 0.8861   
## P-Value [Acc > NIR] : 0.10836   
##   
## Kappa : 0.4972   
##   
## Mcnemar's Test P-Value : 0.05235   
##   
## Sensitivity : 0.9777   
## Specificity : 0.4348   
## Pos Pred Value : 0.9309   
## Neg Pred Value : 0.7143   
## Prevalence : 0.8861   
## Detection Rate : 0.8663   
## Detection Prevalence : 0.9307   
## Balanced Accuracy : 0.7062   
##   
## 'Positive' Class : Violated Parole   
##

### Blood Data Set

library(readr)  
Blood <- read\_csv("Blood.csv")  
Blood = as\_tibble(Blood)  
  
Blood = Blood %>% mutate(DonatedMarch = as\_factor(as.character(DonatedMarch))) %>%  
mutate(DonatedMarch = fct\_recode(DonatedMarch,  
"Yes" = "1",  
"No" = "0"))

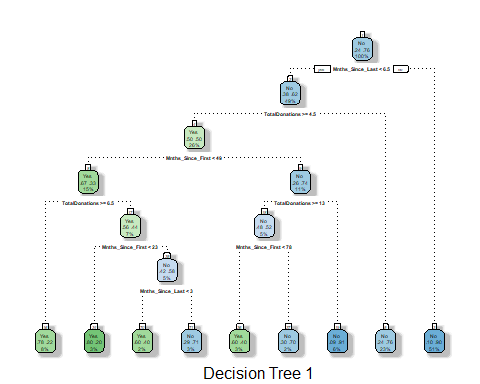
## Training Data

### Training Data Blood Data

set.seed(1234)  
train.rows = createDataPartition(y = Blood$DonatedMarch, p = 0.7, list =FALSE)  
train2 = slice(Blood, train.rows)  
test2 = slice(Blood, -train.rows)

### Tree training plot Blood Data

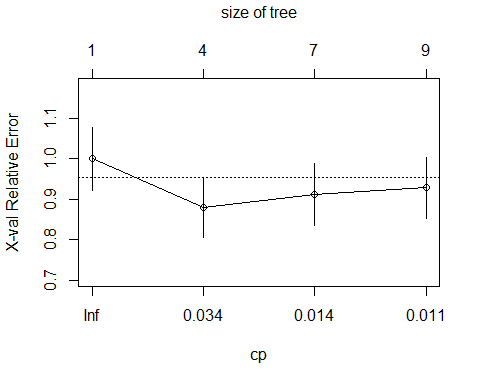
tree\_Blood = rpart(DonatedMarch ~., train2, method = "class")  
fancyRpartPlot(tree\_Blood, sub = "Decision Tree 1")



#Showing accuracy of tree and prevents from a too complex tree  
printcp(tree\_Blood)

##   
## 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(tree\_Blood)



# CP Value Change for Blood Data Training

tree\_Blood = rpart(DonatedMarch ~., train2, cp=.02000, method="class")  
#fancyRpartPlot(tree2, sub = "Decision Tree 1")  
printcp(tree\_Blood)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = train2, method = "class",   
## cp = 0.02)  
##   
## 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 0.020 3 0.784 0.824 0.072775

### Confusion Matrix Training

tree\_Blood = rpart(DonatedMarch ~., train2, method = "class")  
treepred\_Blood = predict(tree\_Blood, train2, type = "class")  
head(treepred\_Blood)

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

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 58 22  
## No 67 377  
##   
## Accuracy : 0.8302   
## 95% CI : (0.7952, 0.8613)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 8.243e-05   
##   
## Kappa : 0.4665   
##   
## Mcnemar's Test P-Value : 3.101e-06   
##   
## Sensitivity : 0.4640   
## Specificity : 0.9449   
## Pos Pred Value : 0.7250   
## Neg Pred Value : 0.8491   
## Prevalence : 0.2385   
## Detection Rate : 0.1107   
## Detection Prevalence : 0.1527   
## Balanced Accuracy : 0.7044   
##   
## 'Positive' Class : Yes   
##

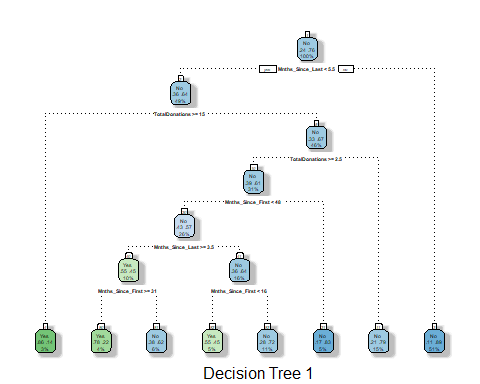
## Testing Data

### Testing Data Blood Data

set.seed(1234)  
train.rows = createDataPartition(y = Blood$DonatedMarch, p = 0.7, list =FALSE)  
train2 = slice(Blood, train.rows)  
test2 = slice(Blood, -train.rows)

### Tree testing plot Blood Data

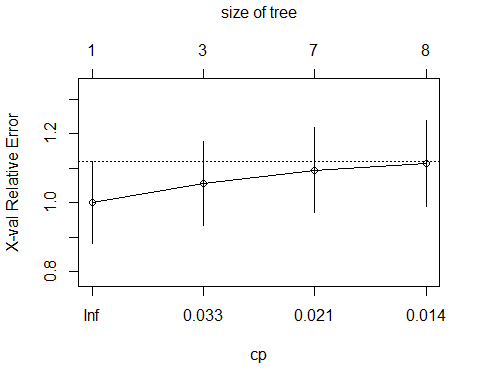
tree\_Blood2 = rpart(DonatedMarch ~., test2, method = "class")  
fancyRpartPlot(tree\_Blood2, sub = "Decision Tree 1")



#Showing accuracy of tree and prevents from a too complex tree  
printcp(tree\_Blood2)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = test2, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] Mnths\_Since\_First Mnths\_Since\_Last TotalDonations   
##   
## Root node error: 53/224 = 0.23661  
##   
## n= 224   
##   
## CP nsplit rel error xerror xstd  
## 1 0.047170 0 1.00000 1.0000 0.12002  
## 2 0.023585 2 0.90566 1.0566 0.12228  
## 3 0.018868 6 0.81132 1.0943 0.12370  
## 4 0.010000 7 0.79245 1.1132 0.12439

plotcp(tree\_Blood2)



# CP Value Change for Blood Data Testing

tree\_Blood = rpart(DonatedMarch ~., test2, cp=0.012579, method="class")  
#fancyRpartPlot(tree2, sub = "Decision Tree 1")  
printcp(tree\_Blood2)

##   
## Classification tree:  
## rpart(formula = DonatedMarch ~ ., data = test2, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] Mnths\_Since\_First Mnths\_Since\_Last TotalDonations   
##   
## Root node error: 53/224 = 0.23661  
##   
## n= 224   
##   
## CP nsplit rel error xerror xstd  
## 1 0.047170 0 1.00000 1.0000 0.12002  
## 2 0.023585 2 0.90566 1.0566 0.12228  
## 3 0.018868 6 0.81132 1.0943 0.12370  
## 4 0.010000 7 0.79245 1.1132 0.12439

### Confusion Matrix Testing

tree\_Blood2 = rpart(DonatedMarch ~., test2, method = "class")  
treepred\_Blood2 = predict(tree\_Blood2, test2, type = "class")  
head(treepred\_Blood2)

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

confusionMatrix(treepred\_Blood2,test2$DonatedMarch,positive="Yes") #predictions first then actual

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 19 8  
## No 34 163  
##   
## Accuracy : 0.8125   
## 95% CI : (0.7551, 0.8614)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.0466967   
##   
## Kappa : 0.3752   
##   
## Mcnemar's Test P-Value : 0.0001145   
##   
## Sensitivity : 0.35849   
## Specificity : 0.95322   
## Pos Pred Value : 0.70370   
## Neg Pred Value : 0.82741   
## Prevalence : 0.23661   
## Detection Rate : 0.08482   
## Detection Prevalence : 0.12054   
## Balanced Accuracy : 0.65585   
##   
## 'Positive' Class : Yes   
##