# Module 4 Assignment 1

## Underwood, Katie

#install.packages("caret")  
#install.packages("rpart")  
#install.packages("rpart.plot")  
#install.packages("rattle")  
#install.packages("RColorBrewer")  
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2 v purrr 0.3.4  
## v tibble 3.0.4 v dplyr 1.0.2  
## v tidyr 1.1.2 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.0

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

library(tidymodels)

## -- Attaching packages -------------------------------------- tidymodels 0.1.2 --

## v broom 0.7.2 v recipes 0.1.15  
## v dials 0.0.9 v rsample 0.0.8   
## v infer 0.5.4 v tune 0.1.2   
## v modeldata 0.1.0 v workflows 0.2.1   
## v parsnip 0.1.5 v yardstick 0.0.7

## -- Conflicts ----------------------------------------- tidymodels\_conflicts() --  
## x scales::discard() masks purrr::discard()  
## x dplyr::filter() masks stats::filter()  
## x recipes::fixed() masks stringr::fixed()  
## x dplyr::lag() masks stats::lag()  
## x yardstick::spec() masks readr::spec()  
## x recipes::step() masks stats::step()

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following objects are masked from 'package:yardstick':  
##   
## precision, recall, sensitivity, specificity

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

library(rpart)

##   
## Attaching package: 'rpart'

## The following object is masked from 'package:dials':  
##   
## prune

library(rpart.plot)  
library(rattle)

## 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 (1).csv")

##   
## -- 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()  
## )

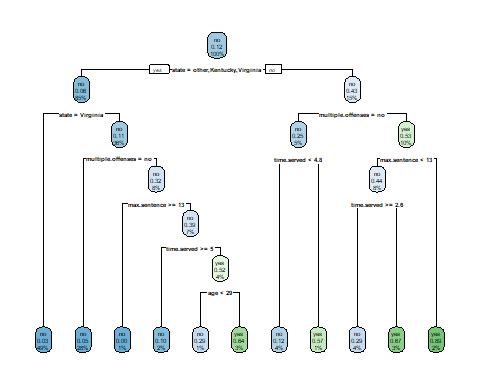
parole = parole %>% mutate(male= as\_factor(male)) %>%  
 mutate(male = fct\_recode(male, "male"= "1", "female" = "0"))  
parole = parole %>% mutate(race= as\_factor(race)) %>%  
 mutate(race = fct\_recode(race, "white" = "1", "other" = "2"))  
parole = parole %>% mutate(state= as\_factor(state)) %>%  
 mutate(state = fct\_recode(state, "other" = "1", "Kentucky" = "2", "Louisiana" = "3", "Virginia" = "4"))  
parole = parole %>% mutate(crime= as\_factor(crime)) %>%  
 mutate(crime = fct\_recode(crime, "other" = "1", "larceny" = "2", "drug" = "3", "drive" = "4"))  
parole = parole %>% mutate(multiple.offenses= as\_factor(multiple.offenses)) %>%  
 mutate(multiple.offenses = fct\_recode(multiple.offenses, "yes" = "1", "no" = "0"))  
parole = parole %>% mutate(violator= as\_factor(violator)) %>%  
 mutate(violator = fct\_recode(violator, "yes" = "1", "no" = "0"))

## Task 1

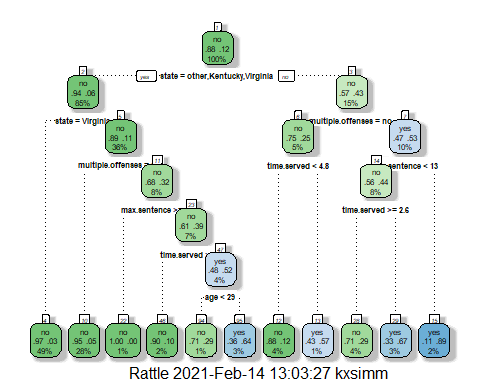
set.seed(12345)  
parole\_split = initial\_split(parole, prop = .7, strata = violator)  
train = training(parole\_split)  
test = testing(parole\_split)

## Task 2

parole\_recipe = recipe(violator ~., train)  
  
tree\_model = decision\_tree() %>%  
 set\_engine("rpart", model = TRUE) %>%  
 set\_mode("classification")  
  
parole\_wflow =   
 workflow() %>%  
 add\_model(tree\_model) %>%  
 add\_recipe(parole\_recipe)  
  
parole\_fit = fit(parole\_wflow, train)  
  
tree = parole\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")  
  
rpart.plot(tree)



fancyRpartPlot(tree, tweak=1.5)



## Task 3

For the given example (40 yo, Louisiana, served 5 years, sentence 10, multiple offenses)

* First decision point is “no” (state=Louisiana)
* Second point is “no” (multiple.offenses=yes)
* Third point is “no” (time served > 4.8 years)

Leads me to a prediction that this person will NOT violate parole. 4% of the population is in this group.

## Task 4

The optimal CP value is .0545 (0 splits) based on the lowest xerror of 1.0. This is not the same as the model from Task 2.

parole\_fit$fit$fit$fit$cptable

## CP nsplit rel error xerror xstd  
## 1 0.05454545 0 1.0000000 1.000000 0.1267582  
## 2 0.01818182 4 0.7818182 1.090909 0.1316005  
## 3 0.01454545 5 0.7636364 1.090909 0.1316005  
## 4 0.01000000 10 0.6909091 1.127273 0.1334515

## Task 5

set.seed(123)  
folds = vfold\_cv(train, v = 5)  
  
parole\_model2 = decision\_tree(cost\_complexity = tune()) %>%  
 set\_engine("rpart", model = TRUE) %>%  
 set\_mode("classification")  
  
parole\_grid = grid\_regular(cost\_complexity(), levels = 25)  
  
parole\_wflow2 =  
 workflow() %>%  
 add\_model(parole\_model2) %>%  
 add\_recipe(parole\_recipe)  
  
tree\_res =   
 parole\_wflow2 %>%  
 tune\_grid(  
 resamples = folds,  
 grid = parole\_grid  
 )

##   
## Attaching package: 'rlang'

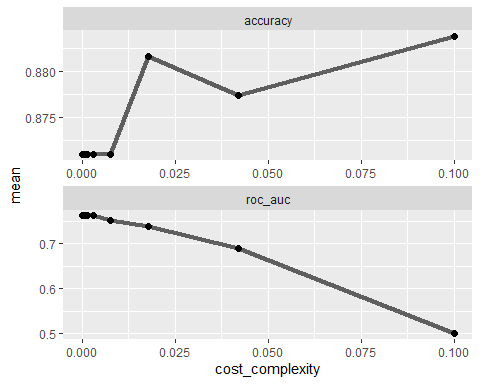
## The following objects are masked from 'package:purrr':  
##   
## %@%, as\_function, flatten, flatten\_chr, flatten\_dbl, flatten\_int,  
## flatten\_lgl, flatten\_raw, invoke, list\_along, modify, prepend,  
## splice

##   
## Attaching package: 'vctrs'

## The following object is masked from 'package:dplyr':  
##   
## data\_frame

## The following object is masked from 'package:tibble':  
##   
## data\_frame

tree\_res %>%  
 collect\_metrics() %>%  
 ggplot(aes(cost\_complexity, mean)) +  
 geom\_line(size = 1.5, alpha = 0.6) +  
 geom\_point(size = 2) +  
 facet\_wrap(~ .metric, scales = "free", nrow = 2)



## Task 6

Optimal cp value is .1

best\_tree = tree\_res %>%  
 select\_best("accuracy")  
  
best\_tree

## # A tibble: 1 x 2  
## cost\_complexity .config   
## <dbl> <chr>   
## 1 0.1 Preprocessor1\_Model25

## Task 7

final\_wf =   
 parole\_wflow2 %>%  
 finalize\_workflow(best\_tree)  
  
final\_fit = fit(final\_wf, train)  
  
tree2 = final\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")  
  
#fancyRpartPlot(tree2, tweak = 1.5)

## Task 8

Accuracy is 88% (on the testing set)

treepred = predict(final\_fit, train, type = "class")  
head(treepred)

## # A tibble: 6 x 1  
## .pred\_class  
## <fct>   
## 1 no   
## 2 no   
## 3 no   
## 4 no   
## 5 no   
## 6 no

confusionMatrix(treepred$.pred\_class,train$violator,positive="yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 418 55  
## yes 0 0  
##   
## Accuracy : 0.8837   
## 95% CI : (0.8513, 0.9112)  
## No Information Rate : 0.8837   
## P-Value [Acc > NIR] : 0.5358   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : 3.305e-13   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.8837   
## Prevalence : 0.1163   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : yes   
##

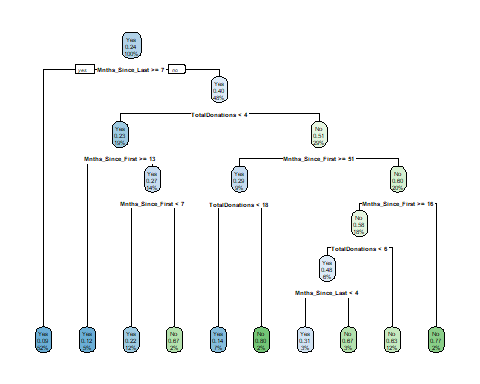
## Task 9

Blood <- read\_csv("Blood.csv")

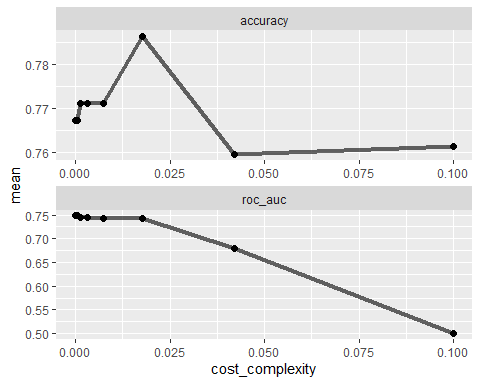
##   
## -- Column specification --------------------------------------------------------  
## cols(  
## Mnths\_Since\_Last = col\_double(),  
## TotalDonations = col\_double(),  
## Total\_Donated = col\_double(),  
## Mnths\_Since\_First = col\_double(),  
## DonatedMarch = col\_double()  
## )

Blood = Blood %>% mutate(DonatedMarch= as\_factor(DonatedMarch)) %>%  
 mutate(DonatedMarch = fct\_recode(DonatedMarch, "No"= "1", "Yes" = "0"))  
set.seed(1234)  
blood\_split = initial\_split(Blood, prop = .7, strata = DonatedMarch)  
train2 = training(blood\_split)  
test2 = testing(blood\_split)

blood\_recipe = recipe(DonatedMarch ~., train2)  
  
tree\_model = decision\_tree() %>%  
 set\_engine("rpart", model = TRUE) %>%  
 set\_mode("classification")  
  
blood\_wflow =   
 workflow() %>%  
 add\_model(tree\_model) %>%  
 add\_recipe(blood\_recipe)  
  
blood\_fit = fit(blood\_wflow, train2)  
  
treeblood = blood\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")  
  
rpart.plot(treeblood)



set.seed(1234)  
folds = vfold\_cv(train2, v = 5)  
  
blood\_model2 = decision\_tree(cost\_complexity = tune()) %>%  
 set\_engine("rpart", model = TRUE) %>%  
 set\_mode("classification")  
  
blood\_grid = grid\_regular(cost\_complexity(), levels = 25)  
  
blood\_wflow2 =  
 workflow() %>%  
 add\_model(blood\_model2) %>%  
 add\_recipe(blood\_recipe)  
  
tree\_res2 =   
 blood\_wflow2 %>%  
 tune\_grid(  
 resamples = folds,  
 grid = blood\_grid  
 )  
  
tree\_res2 %>%  
 collect\_metrics() %>%  
 ggplot(aes(cost\_complexity, mean)) +  
 geom\_line(size = 1.5, alpha = 0.6) +  
 geom\_point(size = 2) +  
 facet\_wrap(~ .metric, scales = "free", nrow = 2)

 The best CP value for the highest accuracy looks to be somewhere between .017-.018

best\_tree2 = tree\_res2 %>%  
 select\_best("accuracy")  
  
best\_tree2

## # A tibble: 1 x 2  
## cost\_complexity .config   
## <dbl> <chr>   
## 1 0.0178 Preprocessor1\_Model23

## Task 10 & Task 11

Accuracy on the training set is .81 and on the testing set is .73.The performance is a decent bit lower on the testing set which makes me concerned about the quality of this model.

final\_wf2 =   
 blood\_wflow2 %>%  
 finalize\_workflow(best\_tree2)  
  
final\_fit2 = fit(final\_wf2, train2)  
  
treeblood2 = final\_fit2 %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")  
  
treepred2 = predict(final\_fit2, train2, type = "class")  
head(treepred2)

## # A tibble: 6 x 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 No   
## 3 No   
## 4 No   
## 5 No   
## 6 No

confusionMatrix(treepred2$.pred\_class,train2$DonatedMarch,positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 355 53  
## No 44 72  
##   
## Accuracy : 0.8149   
## 95% CI : (0.7789, 0.8472)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 0.001931   
##   
## Kappa : 0.4775   
##   
## Mcnemar's Test P-Value : 0.416633   
##   
## Sensitivity : 0.8897   
## Specificity : 0.5760   
## Pos Pred Value : 0.8701   
## Neg Pred Value : 0.6207   
## Prevalence : 0.7615   
## Detection Rate : 0.6775   
## Detection Prevalence : 0.7786   
## Balanced Accuracy : 0.7329   
##   
## 'Positive' Class : Yes   
##

treepred3 = predict(final\_fit2, test2, type = "class")  
head(treepred3)

## # A tibble: 6 x 1  
## .pred\_class  
## <fct>   
## 1 No   
## 2 No   
## 3 No   
## 4 No   
## 5 No   
## 6 Yes

confusionMatrix(treepred3$.pred\_class,test2$DonatedMarch,positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 144 34  
## No 27 19  
##   
## Accuracy : 0.7277   
## 95% CI : (0.6644, 0.7848)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.9077   
##   
## Kappa : 0.2102   
##   
## Mcnemar's Test P-Value : 0.4424   
##   
## Sensitivity : 0.8421   
## Specificity : 0.3585   
## Pos Pred Value : 0.8090   
## Neg Pred Value : 0.4130   
## Prevalence : 0.7634   
## Detection Rate : 0.6429   
## Detection Prevalence : 0.7946   
## Balanced Accuracy : 0.6003   
##   
## 'Positive' Class : Yes   
##