# Classification Trees

## Katia Mottert

## -- 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()

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

## v broom 0.7.3 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()

## 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

##   
## Attaching package: 'rpart'

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

## 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.

##   
## -- 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()  
## )

## Task 1

set.seed(12345)  
parole\_split = initial\_split(parole, prob = 0.70, strata = violator)  
train = training(parole\_split)  
test = testing(parole\_split)

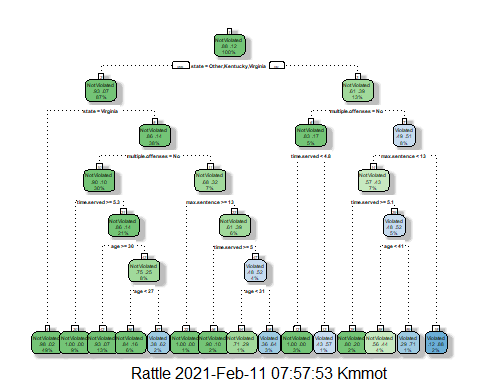
## Task 2

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)

parole\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")

## n= 507   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 507 59 Not Violated (0.88362919 0.11637081)   
## 2) state=Other,Kentucky,Virginia 441 33 Not Violated (0.92517007 0.07482993)   
## 4) state=Virginia 250 6 Not Violated (0.97600000 0.02400000) \*  
## 5) state=Other,Kentucky 191 27 Not Violated (0.85863874 0.14136126)   
## 10) multiple.offenses=No 153 15 Not Violated (0.90196078 0.09803922)   
## 20) time.served>=5.25 45 0 Not Violated (1.00000000 0.00000000) \*  
## 21) time.served< 5.25 108 15 Not Violated (0.86111111 0.13888889)   
## 42) age>=30.35 68 5 Not Violated (0.92647059 0.07352941) \*  
## 43) age< 30.35 40 10 Not Violated (0.75000000 0.25000000)   
## 86) age< 26.7 32 5 Not Violated (0.84375000 0.15625000) \*  
## 87) age>=26.7 8 3 Violated (0.37500000 0.62500000) \*  
## 11) multiple.offenses=Multiple 38 12 Not Violated (0.68421053 0.31578947)   
## 22) max.sentence>=12.5 7 0 Not Violated (1.00000000 0.00000000) \*  
## 23) max.sentence< 12.5 31 12 Not Violated (0.61290323 0.38709677)   
## 46) time.served>=5 10 1 Not Violated (0.90000000 0.10000000) \*  
## 47) time.served< 5 21 10 Violated (0.47619048 0.52380952)   
## 94) age< 31.05 7 2 Not Violated (0.71428571 0.28571429) \*  
## 95) age>=31.05 14 5 Violated (0.35714286 0.64285714) \*  
## 3) state=Louisiana 66 26 Not Violated (0.60606061 0.39393939)   
## 6) multiple.offenses=No 23 4 Not Violated (0.82608696 0.17391304)   
## 12) time.served< 4.8 16 0 Not Violated (1.00000000 0.00000000) \*  
## 13) time.served>=4.8 7 3 Violated (0.42857143 0.57142857) \*  
## 7) multiple.offenses=Multiple 43 21 Violated (0.48837209 0.51162791)   
## 14) max.sentence< 12.5 35 15 Not Violated (0.57142857 0.42857143)   
## 28) time.served>=5.05 10 2 Not Violated (0.80000000 0.20000000) \*  
## 29) time.served< 5.05 25 12 Violated (0.48000000 0.52000000)   
## 58) age< 40.8 18 8 Not Violated (0.55555556 0.44444444) \*  
## 59) age>=40.8 7 2 Violated (0.28571429 0.71428571) \*  
## 15) max.sentence>=12.5 8 1 Violated (0.12500000 0.87500000) \*

tree = parole\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")  
  
fancyRpartPlot(tree, tweak = 1.15)



## Task 3

I would go to the right at the first level as the state does not equal other, Kentucky, or Virginia. Multiple offenses = No would be again to the right since he did commit multiple offenses. Max sentence <13 years go to the left. Then I believe its to the left again (though with adjusting size the labels either are too small or overlaps) as he served 5 years.

## Task 4

.0112 and .0100 have matching xerror values of 1.288 and it looks like its associated with the tree above.

parole\_fit$fit$fit$fit$cptable

## CP nsplit rel error xerror xstd  
## 1 0.03389831 0 1.0000000 1.000000 0.1223796  
## 2 0.02542373 3 0.8983051 1.135593 0.1292432  
## 3 0.01694915 5 0.8474576 1.135593 0.1292432  
## 4 0.01355932 6 0.8305085 1.220339 0.1332155  
## 5 0.01129944 11 0.7627119 1.288136 0.1362352  
## 6 0.01000000 14 0.7288136 1.288136 0.1362352

## Task 5

set.seed(123)  
folds = vfold\_cv(train, v= 5)  
  
parole2\_recipe = recipe (violator ~., train)  
  
tree2\_model = decision\_tree(cost\_complexity = tune()) %>%  
 set\_engine("rpart", model = TRUE) %>%  
 set\_mode("classification")  
  
tree2\_grid = grid\_regular(cost\_complexity(),levels = 25)  
  
parole2\_wflow =   
 workflow() %>%  
 add\_model(tree2\_model) %>%  
 add\_recipe(parole2\_recipe)  
  
tree2\_res =  
 parole2\_wflow %>%  
 tune\_grid(resamples = folds,  
 grid = tree2\_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'

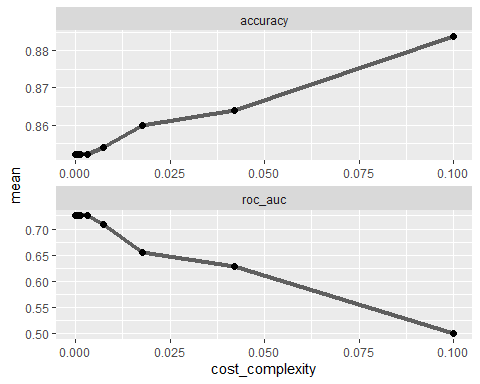
## The following object is masked from 'package:dplyr':  
##   
## data\_frame

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

tree2\_res

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [405/102]> Fold1 <tibble [50 x 5]> <tibble [0 x 1]>  
## 2 <split [405/102]> Fold2 <tibble [50 x 5]> <tibble [0 x 1]>  
## 3 <split [406/101]> Fold3 <tibble [50 x 5]> <tibble [0 x 1]>  
## 4 <split [406/101]> Fold4 <tibble [50 x 5]> <tibble [0 x 1]>  
## 5 <split [406/101]> Fold5 <tibble [50 x 5]> <tibble [0 x 1]>

tree2\_res %>%  
 collect\_metrics() %>%  
 ggplot(aes(cost\_complexity, mean)) +  
 geom\_line(size= 1.5, alpha = .6) +  
 geom\_point(size = 2) +   
 facet\_wrap(~ .metric, scales = "free", nrow = 2)



## Task 6

0.1 cp value yields the optimal accuracy value.

best\_tree = tree2\_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 =   
 parole2\_wflow %>%  
 finalize\_workflow(best\_tree)  
  
final\_fit = fit(final\_wf, train)  
  
tree2 = final\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")  
  
#fancyRpartPlot(tree2, tweak = 1.15)

## Task 8

THere is a 88.36% accuracy on the training set and 88.69% on the testing set. So both are extremely aligned.

#Training set:  
treepred = predict(final\_fit, train, type = "class")  
confusionMatrix(treepred$.pred\_class,train$violator, positive = "Violated")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Not Violated Violated  
## Not Violated 448 59  
## Violated 0 0  
##   
## Accuracy : 0.8836   
## 95% CI : (0.8525, 0.9102)  
## No Information Rate : 0.8836   
## P-Value [Acc > NIR] : 0.5346   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : 4.321e-14   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.8836   
## Prevalence : 0.1164   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : Violated   
##

#Testing Set:  
test1pred = predict(final\_fit, test, type = "class")  
confusionMatrix(test1pred$.pred\_class,test$violator, positive = "Violated")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Not Violated Violated  
## Not Violated 149 19  
## Violated 0 0  
##   
## Accuracy : 0.8869   
## 95% CI : (0.829, 0.9305)  
## No Information Rate : 0.8869   
## P-Value [Acc > NIR] : 0.5607   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : 3.636e-05   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.8869   
## Prevalence : 0.1131   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : Violated   
##

## Task 9

0.0177 is the most optimal cp value on this train set from Blood.

Blood <- read\_csv("~/Courses/BAN502/Module 4/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" = "0", "Yes" = "1" ))

set.seed(1234)  
Blood\_split = initial\_split(Blood, prob = 0.70, strata = DonatedMarch)  
trainBlood = training(Blood\_split)  
testBlood = testing(Blood\_split)  
  
set.seed(1234)  
folds = vfold\_cv(Blood, v= 5)  
  
Blood\_recipe = recipe (DonatedMarch ~., trainBlood)  
  
treeB\_model = decision\_tree(cost\_complexity = tune()) %>%  
 set\_engine("rpart", model = TRUE) %>%  
 set\_mode("classification")  
  
treeB\_grid = grid\_regular(cost\_complexity(),levels = 25)  
  
Blood\_wflow =   
 workflow() %>%  
 add\_model(treeB\_model) %>%  
 add\_recipe(Blood\_recipe)  
  
treeB\_res =  
 Blood\_wflow %>%  
 tune\_grid(resamples = folds,  
 grid = treeB\_grid)  
  
treeB\_res

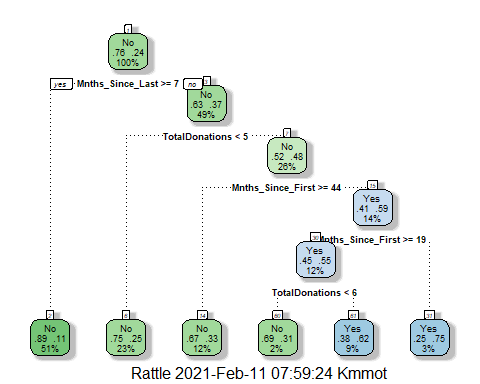
## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [598/150]> Fold1 <tibble [50 x 5]> <tibble [0 x 1]>  
## 2 <split [598/150]> Fold2 <tibble [50 x 5]> <tibble [0 x 1]>  
## 3 <split [598/150]> Fold3 <tibble [50 x 5]> <tibble [0 x 1]>  
## 4 <split [599/149]> Fold4 <tibble [50 x 5]> <tibble [0 x 1]>  
## 5 <split [599/149]> Fold5 <tibble [50 x 5]> <tibble [0 x 1]>

BestB\_tree = treeB\_res %>%  
 select\_best("accuracy")  
BestB\_tree

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

## Task 10

Blood\_wf =   
 Blood\_wflow %>%  
 finalize\_workflow(BestB\_tree)  
  
finalB\_fit = fit(Blood\_wf, trainBlood)  
  
treeB = finalB\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")  
  
fancyRpartPlot(treeB, tweak = 1.15)



## Task 11

79.72% accurate on the training set and 81.18% on the testing set. So relatively the same.

treeBpred = predict(finalB\_fit, trainBlood, type = "class")  
testBpred = predict(finalB\_fit, testBlood, type = "class")  
  
# Training set:  
confusionMatrix(treeBpred$.pred\_class, trainBlood$DonatedMarch, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 404 90  
## Yes 24 44  
##   
## Accuracy : 0.7972   
## 95% CI : (0.7615, 0.8297)  
## No Information Rate : 0.7616   
## P-Value [Acc > NIR] : 0.02523   
##   
## Kappa : 0.3277   
##   
## Mcnemar's Test P-Value : 1.145e-09   
##   
## Sensitivity : 0.32836   
## Specificity : 0.94393   
## Pos Pred Value : 0.64706   
## Neg Pred Value : 0.81781   
## Prevalence : 0.23843   
## Detection Rate : 0.07829   
## Detection Prevalence : 0.12100   
## Balanced Accuracy : 0.63614   
##   
## 'Positive' Class : Yes   
##

# Testing set:  
confusionMatrix(testBpred$.pred\_class, testBlood$DonatedMarch, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 137 30  
## Yes 5 14  
##   
## Accuracy : 0.8118   
## 95% CI : (0.7481, 0.8653)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.06858   
##   
## Kappa : 0.352   
##   
## Mcnemar's Test P-Value : 4.976e-05   
##   
## Sensitivity : 0.31818   
## Specificity : 0.96479   
## Pos Pred Value : 0.73684   
## Neg Pred Value : 0.82036   
## Prevalence : 0.23656   
## Detection Rate : 0.07527   
## Detection Prevalence : 0.10215   
## Balanced Accuracy : 0.64149   
##   
## 'Positive' Class : Yes   
##