Module4\_Assignment1

parole <- read\_csv("C:/Users/cltma/OneDrive/Documents/BAN502/Module4/Module4\_Assignment1/parole.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()  
## )

## male race age state time.served   
## Female:130 White :389 Min. :18.40 Other :143 Min. :0.000   
## Male :545 Non-White:286 1st Qu.:25.35 Kentucky :120 1st Qu.:3.250   
## Median :33.70 Louisiana: 82 Median :4.400   
## Mean :34.51 Virginia :330 Mean :4.198   
## 3rd Qu.:42.55 3rd Qu.:5.200   
## Max. :67.00 Max. :6.000   
## max.sentence multiple.offenses crime violator   
## Min. : 1.00 No :313 All Other :315 No :597   
## 1st Qu.:12.00 Yes:362 Larceny :106 Yes: 78   
## Median :12.00 Drug-Related Crime :153   
## Mean :13.06 Driving-Related Crime:101   
## 3rd Qu.:15.00   
## Max. :18.00

## Rows: 675  
## Columns: 9  
## $ male <fct> Male, Female, Male, Male, Male, Male, Male, Femal...  
## $ race <fct> White, White, Non-White, White, Non-White, Non-Wh...  
## $ age <dbl> 33.2, 39.7, 29.5, 22.4, 21.6, 46.7, 31.0, 24.6, 3...  
## $ state <fct> Other, Other, Other, Other, Other, Other, Other, ...  
## $ time.served <dbl> 5.5, 5.4, 5.6, 5.7, 5.4, 6.0, 6.0, 4.8, 4.5, 4.7,...  
## $ max.sentence <dbl> 18, 12, 12, 18, 12, 18, 18, 12, 13, 12, 12, 12, 1...  
## $ multiple.offenses <fct> No, No, No, No, No, No, No, No, No, No, Yes, No, ...  
## $ crime <fct> Driving-Related Crime, Drug-Related Crime, Drug-R...  
## $ violator <fct> No, No, No, No, No, No, No, No, No, No, No, No, N...

### Task 1

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

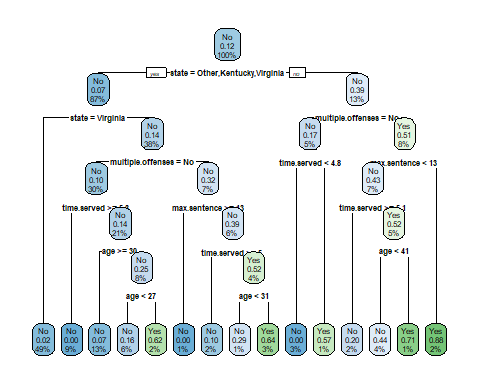
### Task 2

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

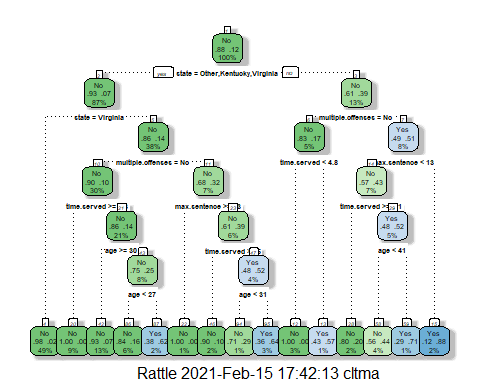
parole\_train\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")

## n= 507   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 507 59 No (0.88362919 0.11637081)   
## 2) state=Other,Kentucky,Virginia 441 33 No (0.92517007 0.07482993)   
## 4) state=Virginia 250 6 No (0.97600000 0.02400000) \*  
## 5) state=Other,Kentucky 191 27 No (0.85863874 0.14136126)   
## 10) multiple.offenses=No 153 15 No (0.90196078 0.09803922)   
## 20) time.served>=5.25 45 0 No (1.00000000 0.00000000) \*  
## 21) time.served< 5.25 108 15 No (0.86111111 0.13888889)   
## 42) age>=30.35 68 5 No (0.92647059 0.07352941) \*  
## 43) age< 30.35 40 10 No (0.75000000 0.25000000)   
## 86) age< 26.7 32 5 No (0.84375000 0.15625000) \*  
## 87) age>=26.7 8 3 Yes (0.37500000 0.62500000) \*  
## 11) multiple.offenses=Yes 38 12 No (0.68421053 0.31578947)   
## 22) max.sentence>=12.5 7 0 No (1.00000000 0.00000000) \*  
## 23) max.sentence< 12.5 31 12 No (0.61290323 0.38709677)   
## 46) time.served>=5 10 1 No (0.90000000 0.10000000) \*  
## 47) time.served< 5 21 10 Yes (0.47619048 0.52380952)   
## 94) age< 31.05 7 2 No (0.71428571 0.28571429) \*  
## 95) age>=31.05 14 5 Yes (0.35714286 0.64285714) \*  
## 3) state=Louisiana 66 26 No (0.60606061 0.39393939)   
## 6) multiple.offenses=No 23 4 No (0.82608696 0.17391304)   
## 12) time.served< 4.8 16 0 No (1.00000000 0.00000000) \*  
## 13) time.served>=4.8 7 3 Yes (0.42857143 0.57142857) \*  
## 7) multiple.offenses=Yes 43 21 Yes (0.48837209 0.51162791)   
## 14) max.sentence< 12.5 35 15 No (0.57142857 0.42857143)   
## 28) time.served>=5.05 10 2 No (0.80000000 0.20000000) \*  
## 29) time.served< 5.05 25 12 Yes (0.48000000 0.52000000)   
## 58) age< 40.8 18 8 No (0.55555556 0.44444444) \*  
## 59) age>=40.8 7 2 Yes (0.28571429 0.71428571) \*  
## 15) max.sentence>=12.5 8 1 Yes (0.12500000 0.87500000) \*

tree = parole\_train\_fit %>%   
 pull\_workflow\_fit() %>%   
 pluck("fit")  
  
rpart.plot(tree, tweak=1.3)



fancyRpartPlot(tree, tweak=1.3)



### Task 3

A 40 year-old parolee from Louisiana who served 5 years in prison, with a sentence of 10 years, and committed multiple offenses would have a decent chance off being a parole violator. Based on the tree 44% of the data set who met this criteria were parole violators. R determined that state is the most dominant variable in predicting violators. Since in the example, the statement is untrue we move to the right side of the tree and see that 13% of our training data set are from Louisiana, with 39% being parole violators. The second level looks at multiple offenders from Louisiana, the stated condition of “multiple.offenders=no” is untrue for our example so we move to the right again and see that 8% of the training data set falls into this classification with 51% being parole violators. The third level looks at time sentenced less than 13 years, and since our parolee was only sentenced to 10 years, we follow the tree to the left. The next condition is time.served>=5.1 and since the the condition is untrue for our example we move to the right again. The next condition asks if age is less than 41, which is true for our example so we move to the left. At this point we see that 4% of the data fall into the classification we followed in the tree with 44% of our data set being parole violators.

### Task 4

parole\_train\_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

R found that the optimal CP Value is 0.01. This aligns with the tree from task 2 since R chose 5 variables to include in the classification tree - state, multiple.offenses, max.sentence, time.served and age with 14 total splits.

### Task 5

set.seed(123)  
folds = vfold\_cv(train, v = 5)  
  
parole\_train\_recipe = recipe(violator ~., train) %>%  
 step\_dummy(all\_nominal(),-all\_outcomes())  
  
tree\_model = decision\_tree(cost\_complexity = tune()) %>%   
 set\_engine("rpart", model = TRUE) %>%   
 set\_mode("classification")  
  
tree\_grid = grid\_regular(cost\_complexity(),  
 levels = 25)   
  
parole\_train\_wflow =   
 workflow() %>%   
 add\_model(tree\_model) %>%   
 add\_recipe(parole\_train\_recipe)  
  
tree\_res =   
 parole\_train\_wflow %>%   
 tune\_grid(  
 resamples = folds,  
 grid = tree\_grid)

##   
## Attaching package: 'rlang'

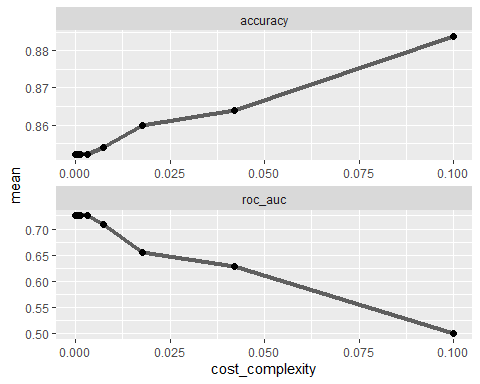
## The following objects are masked from 'package:purrr':  
##   
## %@%, as\_function, flatten, flatten\_chr, flatten\_dbl, flatten\_int,  
## flatten\_lgl, flatten\_raw, invoke, list\_along, modify, prepend,  
## splice

##   
## 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

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

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

A cost\_complexity of 0.1 yields the best accuracy for the tree model.

### Task 7

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

### Task 8

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 448 59  
## Yes 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 : Yes   
##

The accuracy of the “root” in our model is .8836 which is basically no different than the naive model calculation. The p-value between the model created and the naive model is statistically insignificant.

### Task 9

blood <- read\_csv("C:/Users/cltma/OneDrive/Documents/BAN502/Module4/Module4\_Assignment1/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" ))  
  
str(blood)

## tibble [748 x 5] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ Mnths\_Since\_Last : num [1:748] 2 0 1 2 1 4 2 1 2 5 ...  
## $ TotalDonations : num [1:748] 50 13 16 20 24 4 7 12 9 46 ...  
## $ Total\_Donated : num [1:748] 12500 3250 4000 5000 6000 1000 1750 3000 2250 11500 ...  
## $ Mnths\_Since\_First: num [1:748] 98 28 35 45 77 4 14 35 22 98 ...  
## $ DonatedMarch : Factor w/ 2 levels "No","Yes": 2 2 2 2 1 1 2 1 2 2 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. Mnths\_Since\_Last = col\_double(),  
## .. TotalDonations = col\_double(),  
## .. Total\_Donated = col\_double(),  
## .. Mnths\_Since\_First = col\_double(),  
## .. DonatedMarch = col\_double()  
## .. )

set.seed(1234)  
blood\_split = initial\_split(blood, prob = 0.70, strata = DonatedMarch)  
train2 = training(blood\_split)  
test2 = testing(blood\_split)

blood\_train\_recipe = recipe(DonatedMarch ~ ., train2) %>%  
step\_dummy(all\_nominal(),-all\_outcomes())  
  
tree\_model2 = decision\_tree() %>%   
 set\_engine("rpart", model = TRUE) %>%   
 set\_mode("classification")  
  
blood\_train\_wflow =   
 workflow() %>%   
 add\_model(tree\_model2) %>%   
 add\_recipe(blood\_train\_recipe)  
  
blood\_train\_fit = fit(blood\_train\_wflow, train2)

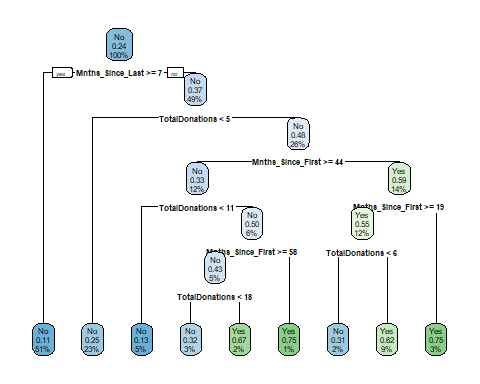
blood\_train\_fit$fit$fit$fit$cptable

## CP nsplit rel error xerror xstd  
## 1 0.03731343 0 1.0000000 1.0000000 0.07538784  
## 2 0.01865672 3 0.8880597 0.9776119 0.07479995  
## 3 0.01492537 5 0.8507463 0.9925373 0.07519371  
## 4 0.01000000 8 0.7985075 1.0000000 0.07538784

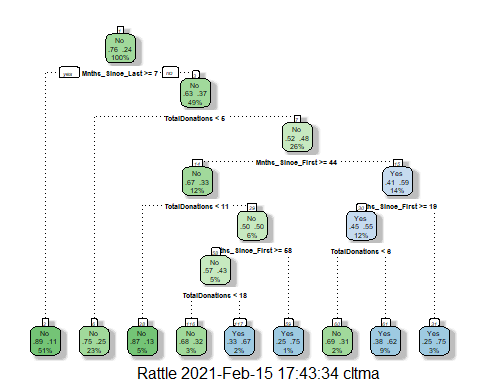
blood\_train\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")

## n= 562   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 562 134 No (0.7615658 0.2384342)   
## 2) Mnths\_Since\_Last>=6.5 285 32 No (0.8877193 0.1122807) \*  
## 3) Mnths\_Since\_Last< 6.5 277 102 No (0.6317690 0.3682310)   
## 6) TotalDonations< 4.5 130 32 No (0.7538462 0.2461538) \*  
## 7) TotalDonations>=4.5 147 70 No (0.5238095 0.4761905)   
## 14) Mnths\_Since\_First>=43.5 66 22 No (0.6666667 0.3333333)   
## 28) TotalDonations< 10.5 30 4 No (0.8666667 0.1333333) \*  
## 29) TotalDonations>=10.5 36 18 No (0.5000000 0.5000000)   
## 58) Mnths\_Since\_First>=57.5 28 12 No (0.5714286 0.4285714)   
## 116) TotalDonations< 18 19 6 No (0.6842105 0.3157895) \*  
## 117) TotalDonations>=18 9 3 Yes (0.3333333 0.6666667) \*  
## 59) Mnths\_Since\_First< 57.5 8 2 Yes (0.2500000 0.7500000) \*  
## 15) Mnths\_Since\_First< 43.5 81 33 Yes (0.4074074 0.5925926)   
## 30) Mnths\_Since\_First>=18.5 65 29 Yes (0.4461538 0.5538462)   
## 60) TotalDonations< 5.5 13 4 No (0.6923077 0.3076923) \*  
## 61) TotalDonations>=5.5 52 20 Yes (0.3846154 0.6153846) \*  
## 31) Mnths\_Since\_First< 18.5 16 4 Yes (0.2500000 0.7500000) \*

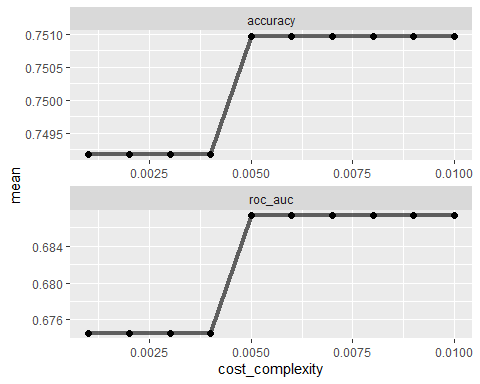
tree2 = blood\_train\_fit %>%   
 pull\_workflow\_fit() %>%   
 pluck("fit")  
  
  
rpart.plot(tree2, tweak=1.3)



fancyRpartPlot(tree2, tweak=1.3)



set.seed(1234)  
folds2 = vfold\_cv(train2, v = 5)  
  
blood\_train\_recipe = recipe(DonatedMarch ~., train2) %>%  
 step\_dummy(all\_nominal(),-all\_outcomes())  
  
tree\_model2 = decision\_tree(cost\_complexity = tune()) %>%   
 set\_engine("rpart", model = TRUE) %>%   
 set\_mode("classification")  
  
tree\_grid2 = expand.grid(cost\_complexity = seq(0.001,0.01,by=0.001))  
  
blood\_train\_wflow =   
 workflow() %>%   
 add\_model(tree\_model2) %>%   
 add\_recipe(blood\_train\_recipe)  
  
tree\_res2 =   
 blood\_train\_wflow %>%   
 tune\_grid(  
 resamples = folds2,  
 grid = tree\_grid2)  
  
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)



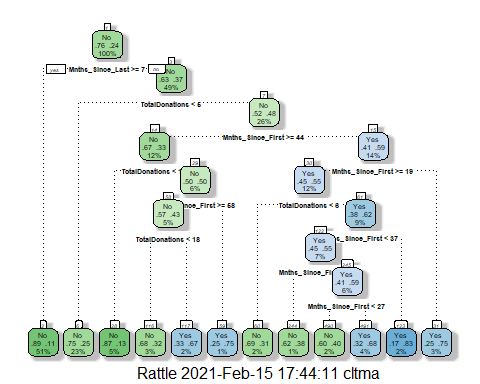
best\_tree2 = tree\_res2 %>%  
 select\_best("accuracy")  
  
best\_tree2

## # A tibble: 1 x 2  
## cost\_complexity .config   
## <dbl> <chr>   
## 1 0.005 Preprocessor1\_Model05

The cp value that appears to optimize accuracy is.005. Please note that when running the treegrid code where R would sample 25 values I was receiving the same error from Task 7. I used the expand tree grid sequence code instead to determine the best fit.

### Task 10

final\_wf2 =   
 blood\_train\_wflow %>%   
 finalize\_workflow(best\_tree2)  
  
final\_fit2 = fit(final\_wf2, train2)  
  
tree2 = final\_fit2 %>%   
 pull\_workflow\_fit() %>%   
 pluck("fit")  
  
fancyRpartPlot(tree2, tweak = 1.5)



### Task 11

treepred2 = predict(final\_fit2, train2, type = "class")  
head(treepred2)

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

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 410 85  
## Yes 18 49  
##   
## Accuracy : 0.8167   
## 95% CI : (0.7822, 0.8479)  
## No Information Rate : 0.7616   
## P-Value [Acc > NIR] : 0.0009708   
##   
## Kappa : 0.3907   
##   
## Mcnemar's Test P-Value : 7.864e-11   
##   
## Sensitivity : 0.36567   
## Specificity : 0.95794   
## Pos Pred Value : 0.73134   
## Neg Pred Value : 0.82828   
## Prevalence : 0.23843   
## Detection Rate : 0.08719   
## Detection Prevalence : 0.11922   
## Balanced Accuracy : 0.66181   
##   
## 'Positive' Class : Yes   
##

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

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

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 136 31  
## Yes 6 13  
##   
## Accuracy : 0.8011   
## 95% CI : (0.7364, 0.8559)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.1301   
##   
## Kappa : 0.315   
##   
## Mcnemar's Test P-Value : 7.961e-05   
##   
## Sensitivity : 0.29545   
## Specificity : 0.95775   
## Pos Pred Value : 0.68421   
## Neg Pred Value : 0.81437   
## Prevalence : 0.23656   
## Detection Rate : 0.06989   
## Detection Prevalence : 0.10215   
## Balanced Accuracy : 0.62660   
##   
## 'Positive' Class : Yes   
##

The accuracy of the predictions for trainingdata set is .8167 while the accuracy of the testing set is.8011. For the training predictions the p-value is statistically significant for our model since it is less than .05. However the p-value for the testing data frame is not statistically significant when comparing the naive model to the created model.