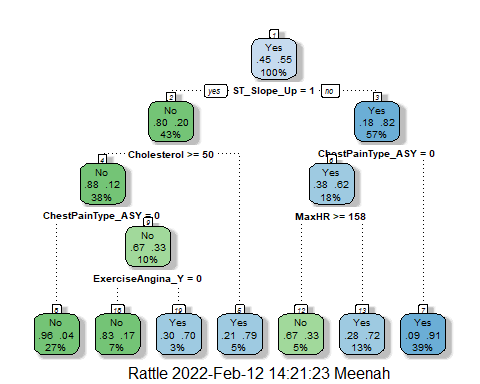
# BAN 502 - Classification Trees Assignment

## Khayrayyah Haamid-Day

heart = heart %>% mutate(Sex = as\_factor(Sex)) %>%  
 mutate(ChestPainType = as\_factor(ChestPainType)) %>%  
 mutate(RestingECG = as\_factor(RestingECG)) %>%  
 mutate(ExerciseAngina = as\_factor(ExerciseAngina)) %>%  
 mutate(HeartDisease = as\_factor(HeartDisease)) %>%  
 mutate(HeartDisease = fct\_recode(HeartDisease, "No" = "0", "Yes" = "1"))

#Task 1  
set.seed(12345)  
  
heart\_split = initial\_split(heart, prop = 0.7, strata = HeartDisease)   
train = training(heart\_split)   
test = testing(heart\_split)

#Task 2  
heart\_recipe = recipe(HeartDisease ~., train) %>%  
 step\_dummy(all\_nominal(),-all\_outcomes())  
  
tree\_model = decision\_tree() %>%   
 set\_engine("rpart", model = TRUE) %>%   
 set\_mode("classification")  
  
heart\_wflow =   
 workflow() %>%   
 add\_model(tree\_model) %>%   
 add\_recipe(heart\_recipe)  
  
heart\_fit = fit(heart\_wflow, train)  
  
tree = heart\_fit %>%   
 extract\_fit\_parsnip() %>%   
 pluck("fit")  
  
  
fancyRpartPlot(tree)



#Task 3  
heart\_fit$fit$fit$fit$cptable

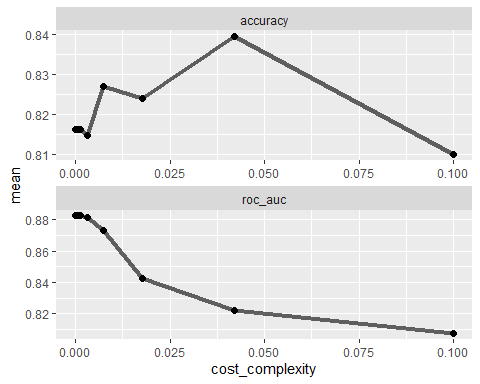
## CP nsplit rel error xerror xstd  
## 1 0.57491289 0 1.0000000 1.0000000 0.04389406  
## 2 0.06620209 1 0.4250871 0.4250871 0.03463635  
## 3 0.01742160 2 0.3588850 0.3588850 0.03240139  
## 4 0.01393728 4 0.3240418 0.3623693 0.03252808  
## 5 0.01000000 6 0.2961672 0.3379791 0.03161802

CP Value of .010 is the most optimal.

#Task 4 - K-fold  
set.seed(123)  
folds = vfold\_cv(train, v = 5)  
  
#Tuning Grid  
heart\_recipe = recipe(HeartDisease ~., train) %>%  
 step\_dummy(all\_nominal(),-all\_outcomes())  
  
tree\_model = decision\_tree(cost\_complexity = tune()) %>%   
 set\_engine("rpart", model = TRUE) %>% #don't forget the model = TRUE flag  
 set\_mode("classification")  
  
tree\_grid = grid\_regular(cost\_complexity(),  
 levels = 25) #try 25 sensible values for cp  
  
heart\_wflow =   
 workflow() %>%   
 add\_model(tree\_model) %>%   
 add\_recipe(heart\_recipe)  
  
tree\_res =   
 heart\_wflow %>%   
 tune\_grid(  
 resamples = folds,  
 grid = tree\_grid  
 )  
  
tree\_res

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [513/129]> Fold1 <tibble [50 x 5]> <tibble [0 x 1]>  
## 2 <split [513/129]> Fold2 <tibble [50 x 5]> <tibble [0 x 1]>  
## 3 <split [514/128]> Fold3 <tibble [50 x 5]> <tibble [0 x 1]>  
## 4 <split [514/128]> Fold4 <tibble [50 x 5]> <tibble [0 x 1]>  
## 5 <split [514/128]> Fold5 <tibble [50 x 5]> <tibble [0 x 1]>

#Model Performance  
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 5  
best\_tree = tree\_res %>%  
 select\_best("accuracy")  
  
best\_tree

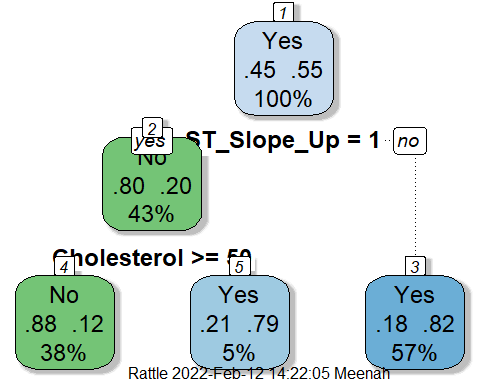
## # A tibble: 1 x 2  
## cost\_complexity .config   
## <dbl> <chr>   
## 1 0.0422 Preprocessor1\_Model24

CP value of .042 yields the optimal accuracy value.

#Task 6  
final\_wf =   
 heart\_wflow %>%   
 finalize\_workflow(best\_tree)  
  
final\_fit = fit(final\_wf, train)  
  
tree2 = final\_fit %>%   
 pull\_workflow\_fit() %>%   
 pluck("fit")

## Warning: `pull\_workflow\_fit()` was deprecated in workflows 0.2.3.  
## Please use `extract\_fit\_parsnip()` instead.

fancyRpartPlot(tree2, tweak = 1.5)



#Task 7  
tree\_pred = predict(final\_fit, train, type = "class")  
head(tree\_pred)

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

confusionMatrix(tree\_pred$.pred\_class,train$HeartDisease,positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 213 29  
## Yes 74 326  
##   
## Accuracy : 0.8396   
## 95% CI : (0.8088, 0.8671)  
## No Information Rate : 0.553   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6705   
##   
## Mcnemar's Test P-Value : 1.455e-05   
##   
## Sensitivity : 0.9183   
## Specificity : 0.7422   
## Pos Pred Value : 0.8150   
## Neg Pred Value : 0.8802   
## Prevalence : 0.5530   
## Detection Rate : 0.5078   
## Detection Prevalence : 0.6231   
## Balanced Accuracy : 0.8302   
##   
## 'Positive' Class : Yes   
##

The accuracy of this tree is 83% or 84%, if rounded up.

#Task 8  
blood <- read\_csv("Blood.csv")

## Rows: 748 Columns: 5

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## dbl (5): Mnths\_Since\_Last, TotalDonations, Total\_Donated, Mnths\_Since\_First,...

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

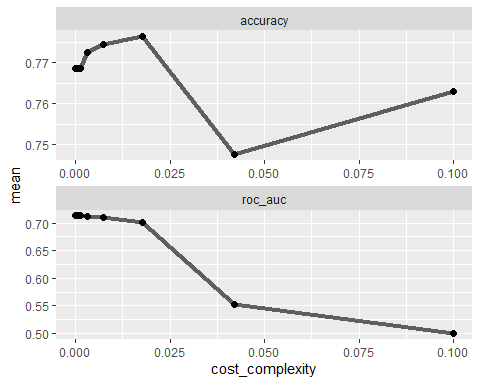
blood = blood %>% mutate(DonatedMarch = as\_factor(DonatedMarch)) %>%  
 mutate(DonatedMarch = fct\_recode(DonatedMarch, "No" = "0", "Yes" = "1"))  
  
summary(blood)

## Mnths\_Since\_Last TotalDonations Total\_Donated Mnths\_Since\_First  
## Min. : 0.000 Min. : 1.000 Min. : 250 Min. : 2.00   
## 1st Qu.: 2.750 1st Qu.: 2.000 1st Qu.: 500 1st Qu.:16.00   
## Median : 7.000 Median : 4.000 Median : 1000 Median :28.00   
## Mean : 9.507 Mean : 5.515 Mean : 1379 Mean :34.28   
## 3rd Qu.:14.000 3rd Qu.: 7.000 3rd Qu.: 1750 3rd Qu.:50.00   
## Max. :74.000 Max. :50.000 Max. :12500 Max. :98.00   
## DonatedMarch  
## No :570   
## Yes:178   
##   
##   
##   
##

#Task 9  
set.seed(1234)  
  
blood\_split = initial\_split(blood, prop = 0.7, strata = DonatedMarch)   
train2 = training(blood\_split)   
test2 = testing(blood\_split)  
  
blood\_recipe = recipe(DonatedMarch ~., train2) %>%  
 step\_dummy(all\_nominal(),-all\_outcomes())  
  
tree\_model2 = decision\_tree() %>%   
 set\_engine("rpart", model = TRUE) %>%   
 set\_mode("classification")  
  
blood\_wflow =   
 workflow() %>%   
 add\_model(tree\_model2) %>%   
 add\_recipe(blood\_recipe)  
  
blood\_fit = fit(blood\_wflow, train2)  
  
tree3 = blood\_fit %>%   
 extract\_fit\_parsnip() %>%   
 pluck("fit")  
  
#K-fold  
set.seed(1234)  
folds = vfold\_cv(train2, v = 5)  
  
#Tuning Grid  
blood\_recipe2 = recipe(DonatedMarch ~., train2) %>%  
 step\_dummy(all\_nominal(),-all\_outcomes())  
  
tree\_model3 = decision\_tree(cost\_complexity = tune()) %>%   
 set\_engine("rpart", model = TRUE) %>%   
 set\_mode("classification")  
  
tree\_grid2 = grid\_regular(cost\_complexity(),  
 levels = 25)   
  
blood\_wflow2 =   
 workflow() %>%   
 add\_model(tree\_model3) %>%   
 add\_recipe(blood\_recipe2)  
  
tree\_res2 =   
 blood\_wflow2 %>%   
 tune\_grid(  
 resamples = folds,  
 grid = tree\_grid2)  
  
tree\_res2

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [418/105]> Fold1 <tibble [50 x 5]> <tibble [0 x 1]>  
## 2 <split [418/105]> Fold2 <tibble [50 x 5]> <tibble [0 x 1]>  
## 3 <split [418/105]> Fold3 <tibble [50 x 5]> <tibble [0 x 1]>  
## 4 <split [419/104]> Fold4 <tibble [50 x 5]> <tibble [0 x 1]>  
## 5 <split [419/104]> Fold5 <tibble [50 x 5]> <tibble [0 x 1]>

#Model Performance  
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)



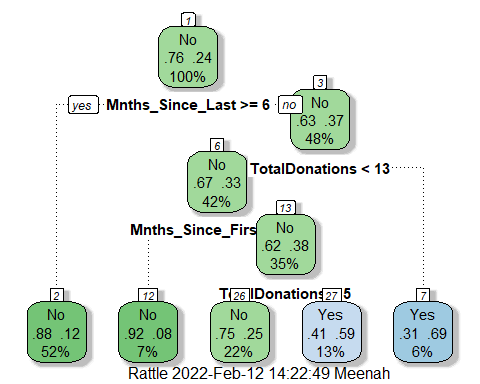
#Task 10  
best\_tree2 = tree\_res2 %>%  
 select\_best("accuracy")  
best\_tree2

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

final\_wf2 =   
 blood\_wflow2 %>%   
 finalize\_workflow(best\_tree2)  
  
final\_fit2 = fit(final\_wf2, train2)  
  
tree4 = final\_fit2 %>%   
 pull\_workflow\_fit() %>%   
 pluck("fit")

## Warning: `pull\_workflow\_fit()` was deprecated in workflows 0.2.3.  
## Please use `extract\_fit\_parsnip()` instead.

fancyRpartPlot(tree4, tweak = 1.5)



#Task 11  
tree\_pred2 = predict(final\_fit2, train2, type = "class")  
head(tree\_pred2)

## # A tibble: 6 x 1  
## .pred\_class  
## <fct>   
## 1 Yes   
## 2 Yes   
## 3 Yes   
## 4 Yes   
## 5 Yes   
## 6 Yes

confusionMatrix(tree\_pred2$.pred\_class,train2$DonatedMarch,positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 361 63  
## Yes 38 61  
##   
## Accuracy : 0.8069   
## 95% CI : (0.7704, 0.8399)  
## No Information Rate : 0.7629   
## P-Value [Acc > NIR] : 0.009214   
##   
## Kappa : 0.4263   
##   
## Mcnemar's Test P-Value : 0.016936   
##   
## Sensitivity : 0.4919   
## Specificity : 0.9048   
## Pos Pred Value : 0.6162   
## Neg Pred Value : 0.8514   
## Prevalence : 0.2371   
## Detection Rate : 0.1166   
## Detection Prevalence : 0.1893   
## Balanced Accuracy : 0.6983   
##   
## 'Positive' Class : Yes   
##

tree\_pred3 = predict(final\_fit2, test2, type = "class")  
head(tree\_pred3)

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

confusionMatrix(tree\_pred3$.pred\_class,test2$DonatedMarch,positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 150 28  
## Yes 21 26  
##   
## Accuracy : 0.7822   
## 95% CI : (0.7225, 0.8343)  
## No Information Rate : 0.76   
## P-Value [Acc > NIR] : 0.2434   
##   
## Kappa : 0.3753   
##   
## Mcnemar's Test P-Value : 0.3914   
##   
## Sensitivity : 0.4815   
## Specificity : 0.8772   
## Pos Pred Value : 0.5532   
## Neg Pred Value : 0.8427   
## Prevalence : 0.2400   
## Detection Rate : 0.1156   
## Detection Prevalence : 0.2089   
## Balanced Accuracy : 0.6793   
##   
## 'Positive' Class : Yes   
##

The training set has a accuracy of 81% and the testing set is 78%; tree’s performance on these sets is good.