library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

## ✔ ggplot2 3.3.6 ✔ purrr 0.3.4  
## ✔ tibble 3.1.7 ✔ dplyr 1.0.9  
## ✔ tidyr 1.2.0 ✔ stringr 1.4.0  
## ✔ readr 2.1.2 ✔ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(tidymodels)

## ── Attaching packages ────────────────────────────────────── tidymodels 0.2.0 ──

## ✔ broom 0.8.0 ✔ rsample 0.1.1  
## ✔ dials 0.1.1 ✔ tune 0.2.0  
## ✔ infer 1.0.0 ✔ workflows 0.2.6  
## ✔ modeldata 0.1.1 ✔ workflowsets 0.2.1  
## ✔ parsnip 0.2.1 ✔ yardstick 0.0.9  
## ✔ recipes 0.2.0

## ── Conflicts ───────────────────────────────────────── tidymodels\_conflicts() ──  
## ✖ scales::discard() masks purrr::discard()  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ recipes::fixed() masks stringr::fixed()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ yardstick::spec() masks readr::spec()  
## ✖ recipes::step() masks stats::step()  
## • Use suppressPackageStartupMessages() to eliminate package startup messages

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 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

library(RColorBrewer)

heart = read.csv("heart\_disease-1.csv")

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

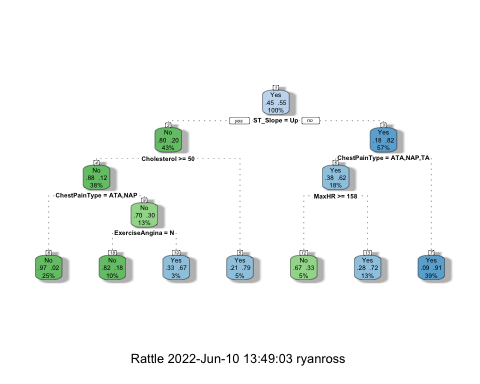
set.seed(12345)   
heart\_split = initial\_split(heart, prop = 0.7, strata = HeartDisease) #70% in training  
train = training(heart\_split)   
test = testing(heart\_split)

heart\_recipe = recipe(HeartDisease ~., train)  
  
tree\_model = decision\_tree() %>%   
 set\_engine("rpart", model = TRUE) %>% #don't forget the model = TRUE flag  
 set\_mode("classification")  
  
heart\_wflow =   
 workflow() %>%   
 add\_model(tree\_model) %>%   
 add\_recipe(heart\_recipe)  
  
heart\_fit = fit(heart\_wflow, train)

tree = heart\_fit %>%   
 pull\_workflow\_fit() %>%   
 pluck("fit")

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

#plot the tree  
fancyRpartPlot(tree)



heart\_fit$fit$fit$fit$cptable

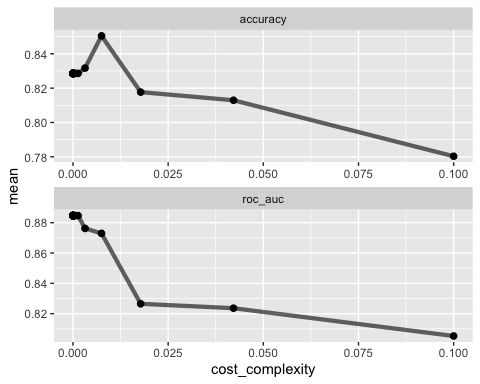
## CP nsplit rel error xerror xstd  
## 1 0.57491289 0 1.0000000 1.0000000 0.04389406  
## 2 0.06620209 1 0.4250871 0.4250871 0.03463635  
## 3 0.01742160 2 0.3588850 0.3588850 0.03240139  
## 4 0.01219512 4 0.3240418 0.3902439 0.03350407  
## 5 0.01000000 6 0.2996516 0.3937282 0.03362154

set.seed(123)  
folds = vfold\_cv(train, v = 5)

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 × 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [513/129]> Fold1 <tibble [50 × 5]> <tibble [0 × 3]>  
## 2 <split [513/129]> Fold2 <tibble [50 × 5]> <tibble [0 × 3]>  
## 3 <split [514/128]> Fold3 <tibble [50 × 5]> <tibble [0 × 3]>  
## 4 <split [514/128]> Fold4 <tibble [50 × 5]> <tibble [0 × 3]>  
## 5 <split [514/128]> Fold5 <tibble [50 × 5]> <tibble [0 × 3]>

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)



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

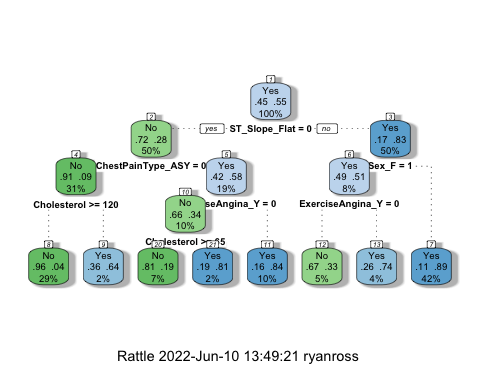
## # A tibble: 1 × 2  
## cost\_complexity .config   
## <dbl> <chr>   
## 1 0.00750 Preprocessor1\_Model22

final\_wf =   
 heart\_wflow %>%   
 finalize\_workflow(best\_tree)

final\_fit = fit(final\_wf, train)  
  
tree = 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(tree, tweak = 1.5)



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

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

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 226 25  
## Yes 61 330  
##   
## Accuracy : 0.866   
## 95% CI : (0.8372, 0.8914)  
## No Information Rate : 0.553   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7258   
##   
## Mcnemar's Test P-Value : 0.0001606   
##   
## Sensitivity : 0.9296   
## Specificity : 0.7875   
## Pos Pred Value : 0.8440   
## Neg Pred Value : 0.9004   
## Prevalence : 0.5530   
## Detection Rate : 0.5140   
## Detection Prevalence : 0.6090   
## Balanced Accuracy : 0.8585   
##   
## 'Positive' Class : Yes   
##

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

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

treepred\_test = predict(heart\_fit, test, type = "class")  
head(treepred\_test)

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

confusionMatrix(treepred\_test$.pred\_class,test$HeartDisease,positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 93 15  
## Yes 30 138  
##   
## Accuracy : 0.837   
## 95% CI : (0.788, 0.8785)  
## No Information Rate : 0.5543   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.666   
##   
## Mcnemar's Test P-Value : 0.03689   
##   
## Sensitivity : 0.9020   
## Specificity : 0.7561   
## Pos Pred Value : 0.8214   
## Neg Pred Value : 0.8611   
## Prevalence : 0.5543   
## Detection Rate : 0.5000   
## Detection Prevalence : 0.6087   
## Balanced Accuracy : 0.8290   
##   
## 'Positive' Class : Yes   
##