# Module 4 - Assignment 1

## Amanda Odum

options(tidyverse.quiet = TRUE)  
library(tidyverse)  
library(tidymodels)

## Registered S3 method overwritten by 'tune':  
## method from   
## required\_pkgs.model\_spec parsnip

## ── Attaching packages ────────────────────────────────────── tidymodels 0.1.4 ──

## ✓ broom 0.7.9 ✓ rsample 0.1.1   
## ✓ dials 0.0.10 ✓ tune 0.1.6   
## ✓ infer 1.0.0 ✓ workflows 0.2.4   
## ✓ modeldata 0.1.1 ✓ workflowsets 0.1.0   
## ✓ parsnip 0.1.7 ✓ yardstick 0.0.9   
## ✓ recipes 0.1.17

## ── 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()  
## • 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(RColorBrewer)   
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.

heart <- read\_csv("heart\_disease-1.csv")

## Rows: 918 Columns: 12

## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (5): Sex, ChestPainType, RestingECG, ExerciseAngina, ST\_Slope  
## dbl (7): Age, RestingBP, Cholesterol, FastingBS, MaxHR, Oldpeak, HeartDisease

##   
## 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.

str(heart)

## spec\_tbl\_df [918 × 12] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ Age : num [1:918] 40 49 37 48 54 39 45 54 37 48 ...  
## $ Sex : chr [1:918] "M" "F" "M" "F" ...  
## $ ChestPainType : chr [1:918] "ATA" "NAP" "ATA" "ASY" ...  
## $ RestingBP : num [1:918] 140 160 130 138 150 120 130 110 140 120 ...  
## $ Cholesterol : num [1:918] 289 180 283 214 195 339 237 208 207 284 ...  
## $ FastingBS : num [1:918] 0 0 0 0 0 0 0 0 0 0 ...  
## $ RestingECG : chr [1:918] "Normal" "Normal" "ST" "Normal" ...  
## $ MaxHR : num [1:918] 172 156 98 108 122 170 170 142 130 120 ...  
## $ ExerciseAngina: chr [1:918] "N" "N" "N" "Y" ...  
## $ Oldpeak : num [1:918] 0 1 0 1.5 0 0 0 0 1.5 0 ...  
## $ ST\_Slope : chr [1:918] "Up" "Flat" "Up" "Flat" ...  
## $ HeartDisease : num [1:918] 0 1 0 1 0 0 0 0 1 0 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. Age = col\_double(),  
## .. Sex = col\_character(),  
## .. ChestPainType = col\_character(),  
## .. RestingBP = col\_double(),  
## .. Cholesterol = col\_double(),  
## .. FastingBS = col\_double(),  
## .. RestingECG = col\_character(),  
## .. MaxHR = col\_double(),  
## .. ExerciseAngina = col\_character(),  
## .. Oldpeak = col\_double(),  
## .. ST\_Slope = col\_character(),  
## .. HeartDisease = col\_double()  
## .. )  
## - attr(\*, "problems")=<externalptr>

summary(heart)

## Age Sex ChestPainType RestingBP   
## Min. :28.00 Length:918 Length:918 Min. : 0.0   
## 1st Qu.:47.00 Class :character Class :character 1st Qu.:120.0   
## Median :54.00 Mode :character Mode :character Median :130.0   
## Mean :53.51 Mean :132.4   
## 3rd Qu.:60.00 3rd Qu.:140.0   
## Max. :77.00 Max. :200.0   
## Cholesterol FastingBS RestingECG MaxHR   
## Min. : 0.0 Min. :0.0000 Length:918 Min. : 60.0   
## 1st Qu.:173.2 1st Qu.:0.0000 Class :character 1st Qu.:120.0   
## Median :223.0 Median :0.0000 Mode :character Median :138.0   
## Mean :198.8 Mean :0.2331 Mean :136.8   
## 3rd Qu.:267.0 3rd Qu.:0.0000 3rd Qu.:156.0   
## Max. :603.0 Max. :1.0000 Max. :202.0   
## ExerciseAngina Oldpeak ST\_Slope HeartDisease   
## Length:918 Min. :-2.6000 Length:918 Min. :0.0000   
## Class :character 1st Qu.: 0.0000 Class :character 1st Qu.:0.0000   
## Mode :character Median : 0.6000 Mode :character Median :1.0000   
## Mean : 0.8874 Mean :0.5534   
## 3rd Qu.: 1.5000 3rd Qu.:1.0000   
## Max. : 6.2000 Max. :1.0000

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

## spec\_tbl\_df [918 × 12] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ Age : num [1:918] 40 49 37 48 54 39 45 54 37 48 ...  
## $ Sex : Factor w/ 2 levels "M","F": 1 2 1 2 1 1 2 1 1 2 ...  
## $ ChestPainType : Factor w/ 4 levels "ATA","NAP","ASY",..: 1 2 1 3 2 2 1 1 3 1 ...  
## $ RestingBP : num [1:918] 140 160 130 138 150 120 130 110 140 120 ...  
## $ Cholesterol : num [1:918] 289 180 283 214 195 339 237 208 207 284 ...  
## $ FastingBS : num [1:918] 0 0 0 0 0 0 0 0 0 0 ...  
## $ RestingECG : Factor w/ 3 levels "Normal","ST",..: 1 1 2 1 1 1 1 1 1 1 ...  
## $ MaxHR : num [1:918] 172 156 98 108 122 170 170 142 130 120 ...  
## $ ExerciseAngina: Factor w/ 2 levels "N","Y": 1 1 1 2 1 1 1 1 2 1 ...  
## $ Oldpeak : num [1:918] 0 1 0 1.5 0 0 0 0 1.5 0 ...  
## $ ST\_Slope : chr [1:918] "Up" "Flat" "Up" "Flat" ...  
## $ HeartDisease : Factor w/ 2 levels "No","Yes": 1 2 1 2 1 1 1 1 2 1 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. Age = col\_double(),  
## .. Sex = col\_character(),  
## .. ChestPainType = col\_character(),  
## .. RestingBP = col\_double(),  
## .. Cholesterol = col\_double(),  
## .. FastingBS = col\_double(),  
## .. RestingECG = col\_character(),  
## .. MaxHR = col\_double(),  
## .. ExerciseAngina = col\_character(),  
## .. Oldpeak = col\_double(),  
## .. ST\_Slope = col\_character(),  
## .. HeartDisease = col\_double()  
## .. )  
## - attr(\*, "problems")=<externalptr>

**Task 1**

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

**Task2**

heart\_recipe = recipe(HeartDisease ~., train)  
  
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)  
  
heart\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")

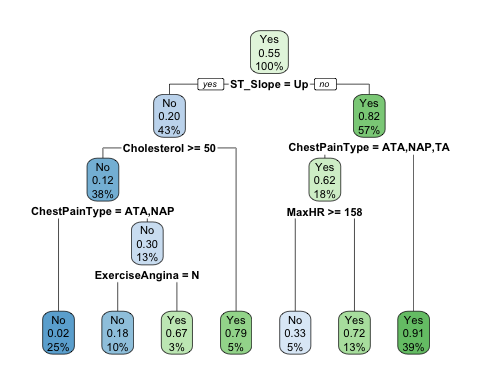
## Warning: `pull\_workflow\_fit()` was deprecated in workflows 0.2.3.  
## Please use `extract\_fit\_parsnip()` instead.

## n= 642   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 642 287 Yes (0.44704050 0.55295950)   
## 2) ST\_Slope=Up 275 55 No (0.80000000 0.20000000)   
## 4) Cholesterol>=50 242 29 No (0.88016529 0.11983471)   
## 8) ChestPainType=ATA,NAP 160 4 No (0.97500000 0.02500000) \*  
## 9) ChestPainType=ASY,TA 82 25 No (0.69512195 0.30487805)   
## 18) ExerciseAngina=N 61 11 No (0.81967213 0.18032787) \*  
## 19) ExerciseAngina=Y 21 7 Yes (0.33333333 0.66666667) \*  
## 5) Cholesterol< 50 33 7 Yes (0.21212121 0.78787879) \*  
## 3) ST\_Slope=Down,Flat 367 67 Yes (0.18256131 0.81743869)   
## 6) ChestPainType=ATA,NAP,TA 116 44 Yes (0.37931034 0.62068966)   
## 12) MaxHR>=157.5 30 10 No (0.66666667 0.33333333) \*  
## 13) MaxHR< 157.5 86 24 Yes (0.27906977 0.72093023) \*  
## 7) ChestPainType=ASY 251 23 Yes (0.09163347 0.90836653) \*

tree = heart\_fit %>% #extract the tree's fit from the fit object  
 pull\_workflow\_fit() %>%   
 pluck("fit")

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

rpart.plot(tree) #plot the tree

 **Task 3**

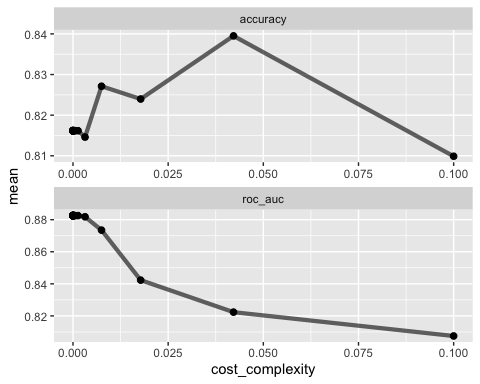
heart\_fit$fit$fit$fit$cptable #examine cp values

## 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

*Two splits (CP=0.01742160) was the optimized cp value tried by R as this resulted in the lowest xerror value of 0.3588850.*

**Task 4**

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) %>%   
 set\_mode("classification")  
  
tree\_grid = grid\_regular(cost\_complexity(),  
 levels = 25)  
heart\_wflow =   
 workflow() %>%   
 add\_model(tree\_model) %>%   
 add\_recipe(heart\_recipe)  
  
tree\_res =   
 heart\_wflow %>%   
 tune\_grid(  
 resamples = folds,  
 grid = tree\_grid  
 )  
  
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

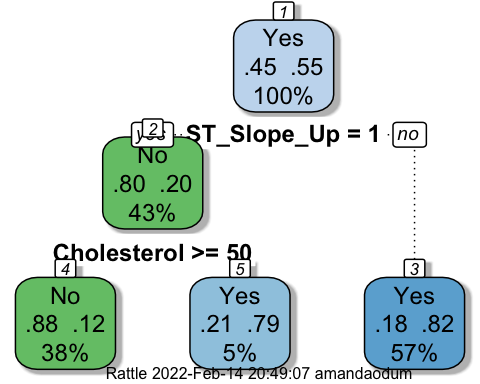
## # A tibble: 1 × 2  
## cost\_complexity .config   
## <dbl> <chr>   
## 1 0.0422 Preprocessor1\_Model24

**Task 6**

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)



**Task 7**

treepred1 = predict(final\_fit, train, type = "class")  
head(treepred1)

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

confusionMatrix(treepred1$.pred\_class,train$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   
##

treepred1\_test = predict(final\_fit, test, type = "class")  
head(treepred1\_test)

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

confusionMatrix(treepred1\_test$.pred\_class,test$HeartDisease,positive="Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 91 16  
## Yes 32 137  
##   
## Accuracy : 0.8261   
## 95% CI : (0.7761, 0.8689)  
## No Information Rate : 0.5543   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.6435   
##   
## Mcnemar's Test P-Value : 0.03038   
##   
## Sensitivity : 0.8954   
## Specificity : 0.7398   
## Pos Pred Value : 0.8107   
## Neg Pred Value : 0.8505   
## Prevalence : 0.5543   
## Detection Rate : 0.4964   
## Detection Prevalence : 0.6123   
## Balanced Accuracy : 0.8176   
##   
## 'Positive' Class : Yes   
##

*The accuracy is 82.61% on the test data set.*

**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" ))

**Task 9**

set.seed(1234)   
blood\_split = initial\_split(blood, prop = 0.7, strata = DonatedMarch) #70% in training  
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)  
  
blood\_fit %>%  
 pull\_workflow\_fit() %>%  
 pluck("fit")

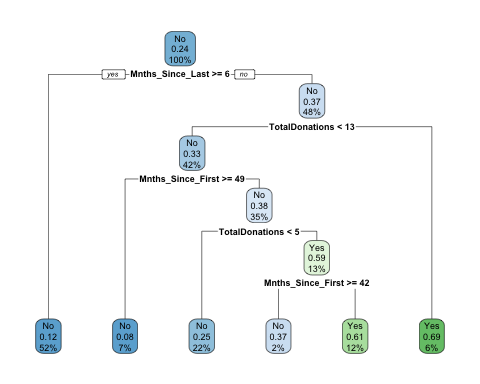
## Warning: `pull\_workflow\_fit()` was deprecated in workflows 0.2.3.  
## Please use `extract\_fit\_parsnip()` instead.

## n= 523   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 523 124 No (0.76290631 0.23709369)   
## 2) Mnths\_Since\_Last>=5.5 273 32 No (0.88278388 0.11721612) \*  
## 3) Mnths\_Since\_Last< 5.5 250 92 No (0.63200000 0.36800000)   
## 6) TotalDonations< 12.5 221 72 No (0.67420814 0.32579186)   
## 12) Mnths\_Since\_First>=48.5 37 3 No (0.91891892 0.08108108) \*  
## 13) Mnths\_Since\_First< 48.5 184 69 No (0.62500000 0.37500000)   
## 26) TotalDonations< 4.5 114 28 No (0.75438596 0.24561404) \*  
## 27) TotalDonations>=4.5 70 29 Yes (0.41428571 0.58571429)   
## 54) Mnths\_Since\_First>=42 8 3 No (0.62500000 0.37500000) \*  
## 55) Mnths\_Since\_First< 42 62 24 Yes (0.38709677 0.61290323) \*  
## 7) TotalDonations>=12.5 29 9 Yes (0.31034483 0.68965517) \*

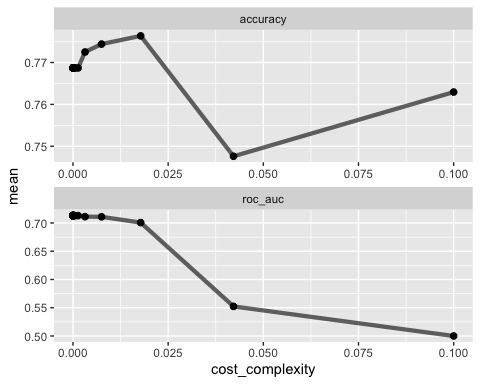
tree2 = blood\_fit %>% #extract the tree's fit from the fit object  
 pull\_workflow\_fit() %>%   
 pluck("fit")

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

rpart.plot(tree2) #plot the tree



set.seed(1234)  
folds = vfold\_cv(train2, v = 5)  
  
blood\_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 = grid\_regular(cost\_complexity(),  
 levels = 25)  
blood\_wflow =   
 workflow() %>%   
 add\_model(tree\_model2) %>%   
 add\_recipe(blood\_recipe)  
  
tree\_res2 =   
 blood\_wflow %>%   
 tune\_grid(  
 resamples = folds,  
 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 × 2  
## cost\_complexity .config   
## <dbl> <chr>   
## 1 0.0178 Preprocessor1\_Model23

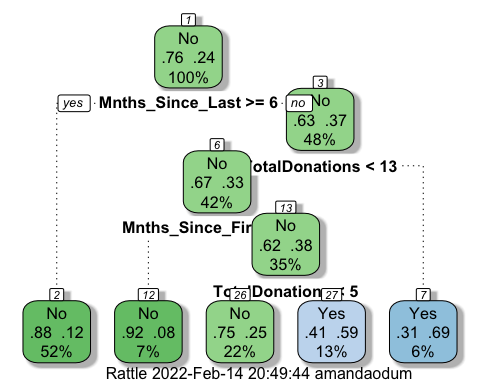
*0.01778279 appears to maximize accuracy.*

**Task 10**

final\_wf =   
 blood\_wflow %>%   
 finalize\_workflow(best\_tree2)  
  
final\_fit = fit(final\_wf, train2)  
  
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 11**

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

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

confusionMatrix(treepred2$.pred\_class,train2$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   
##

treepred2\_test = predict(final\_fit, test2, type = "class")  
head(treepred2\_test)

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

confusionMatrix(treepred2\_test$.pred\_class,test2$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   
##

*Accuracy on the training set is 80.69% and on the testing set is 78.22%. The tree performed well on both sets, though with higher accuracy on the training set.*