Mod4\_Assign3-Quiz3-RandomForest

Grace Williams

2023-06-08

##For this assignment you will need the following libraries: tidyverse, tidymodels, caret, gridExtra, vip, and ranger

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.2 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.4.2 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.1   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(tidymodels)

## ── Attaching packages ────────────────────────────────────── tidymodels 1.1.0 ──  
## ✔ broom 1.0.5 ✔ rsample 1.1.1  
## ✔ dials 1.2.0 ✔ tune 1.1.1  
## ✔ infer 1.0.4 ✔ workflows 1.1.3  
## ✔ modeldata 1.1.0 ✔ workflowsets 1.0.1  
## ✔ parsnip 1.1.0 ✔ yardstick 1.2.0  
## ✔ recipes 1.0.6

## Warning: package 'broom' was built under R version 4.3.1

## ── 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 tidymodels\_prefer() to resolve common conflicts.

library(mice) #package for imputation

##   
## Attaching package: 'mice'  
##   
## The following object is masked from 'package:stats':  
##   
## filter  
##   
## The following objects are masked from 'package:base':  
##   
## cbind, rbind

library(VIM) #visualizing missingness

## Loading required package: colorspace  
## Loading required package: grid  
## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,  
## which was just loaded, will retire in October 2023.  
## Please refer to R-spatial evolution reports for details, especially  
## https://r-spatial.org/r/2023/05/15/evolution4.html.  
## It may be desirable to make the sf package available;  
## package maintainers should consider adding sf to Suggests:.  
## The sp package is now running under evolution status 2  
## (status 2 uses the sf package in place of rgdal)  
## VIM is ready to use.  
##   
## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues  
##   
## Attaching package: 'VIM'  
##   
## The following object is masked from 'package:recipes':  
##   
## prepare  
##   
## The following object is masked from 'package:datasets':  
##   
## sleep

library(ranger) #for random forests  
library(randomForest) #also for random forests

## randomForest 4.7-1.1  
## Type rfNews() to see new features/changes/bug fixes.  
##   
## Attaching package: 'randomForest'  
##   
## The following object is masked from 'package:ranger':  
##   
## importance  
##   
## The following object is masked from 'package:dplyr':  
##   
## combine  
##   
## The following object is masked from 'package:ggplot2':  
##   
## margin

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(skimr)  
library(GGally)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(gridExtra)

##   
## Attaching package: 'gridExtra'  
##   
## The following object is masked from 'package:randomForest':  
##   
## combine  
##   
## The following object is masked from 'package:dplyr':  
##   
## combine

library(vip)

##   
## Attaching package: 'vip'  
##   
## The following object is masked from 'package:utils':  
##   
## vi

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.  
##   
## Attaching package: 'rattle'  
##   
## The following object is masked from 'package:randomForest':  
##   
## importance  
##   
## The following object is masked from 'package:ranger':  
##   
## importance  
##   
## The following object is masked from 'package:VIM':  
##   
## wine

drug <- read\_csv("drug\_data-2.csv")

## Rows: 1885 Columns: 32  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (19): Column14, Column15, Column16, Column17, Column18, Column19, Column...  
## dbl (13): Column1, Column2, Column3, Column4, Column5, Column6, Column7, Col...  
##   
## 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.

## The columns do not have names, so we’ll supply names via the names function

names(drug) = c("ID", "Age", "Gender", "Education", "Country", "Ethnicity", "Nscore", "Escore", "Oscore", "Ascore", "Cscore", "Impulsive", "SS", "Alcohol", "Amphet", "Amyl", "Benzos", "Caff", "Cannabis", "Choc", "Coke", "Crack", "Ecstasy", "Heroin", "Ketamine", "Legalh", "LSD", "Meth", "Mushrooms", "Nicotine", "Semer", "VSA")

## Next-up we change all CL0 and CL1 values to “No” and CL2, CL3, CL4, CL5, and CL6 values to “Yes”. CL0 and CL1 imply the drug was never used or used over a decade ago. CL2 through CL6 imply more recent drug use. The code below finds any CL0, CL1, etc. values in the data frame and replaces them with the appropriate “No” or “Yes”.

drug[drug == "CL0"] = "No"  
drug[drug == "CL1"] = "No"  
drug[drug == "CL2"] = "Yes"  
drug[drug == "CL3"] = "Yes"  
drug[drug == "CL4"] = "Yes"  
drug[drug == "CL5"] = "Yes"  
drug[drug == "CL6"] = "Yes"

## Next up we do a good bit of factor conversion and recoding. Note the use of mutate\_at to target specific ranges of variables. You may see a warning message about the use of the funs() function. It is OK to ignore this

drug\_clean = drug %>% mutate\_at(vars(Age:Ethnicity), funs(as\_factor)) %>%   
 mutate(Age = factor(Age, labels = c("18\_24", "25\_34", "35\_44", "45\_54",  
"55\_64", "65\_"))) %>%  
 mutate(Gender = factor(Gender, labels = c("Male", "Female"))) %>%  
 mutate(Education = factor(Education, labels = c("Under16", "At16", "At17", "At18",  
"SomeCollege","ProfessionalCert",  
"Bachelors", "Masters",  
"Doctorate"))) %>%  
 mutate(Country = factor(Country, labels = c("USA", "NewZealand", "Other", "Australia",  
"Ireland","Canada","UK"))) %>%  
 mutate(Ethnicity = factor(Ethnicity, labels = c("Black", "Asian", "White",  
"White/Black", "Other",  
"White/Asian", "Black/Asian"))) %>%  
 mutate\_at(vars(Alcohol:VSA), funs(as\_factor)) %>%  
 select(-ID)

## Warning: `funs()` was deprecated in dplyr 0.8.0.  
## ℹ Please use a list of either functions or lambdas:  
##   
## # Simple named list: list(mean = mean, median = median)  
##   
## # Auto named with `tibble::lst()`: tibble::lst(mean, median)  
##   
## # Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.

## Warning: `funs()` was deprecated in dplyr 0.8.0.  
## ℹ Please use a list of either functions or lambdas:  
##   
## # Simple named list: list(mean = mean, median = median)  
##   
## # Auto named with `tibble::lst()`: tibble::lst(mean, median)  
##   
## # Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.

str(drug\_clean)

## tibble [1,885 × 31] (S3: tbl\_df/tbl/data.frame)  
## $ Age : Factor w/ 6 levels "18\_24","25\_34",..: 3 2 3 1 3 6 4 3 3 5 ...  
## $ Gender : Factor w/ 2 levels "Male","Female": 2 1 1 2 2 2 1 1 2 1 ...  
## $ Education: Factor w/ 9 levels "Under16","At16",..: 6 9 6 8 9 4 8 2 6 8 ...  
## $ Country : Factor w/ 7 levels "USA","NewZealand",..: 7 7 7 7 7 6 1 7 6 7 ...  
## $ Ethnicity: Factor w/ 7 levels "Black","Asian",..: 6 3 3 3 3 3 3 3 3 3 ...  
## $ Nscore : num [1:1885] 0.313 -0.678 -0.467 -0.149 0.735 ...  
## $ Escore : num [1:1885] -0.575 1.939 0.805 -0.806 -1.633 ...  
## $ Oscore : num [1:1885] -0.5833 1.4353 -0.8473 -0.0193 -0.4517 ...  
## $ Ascore : num [1:1885] -0.917 0.761 -1.621 0.59 -0.302 ...  
## $ Cscore : num [1:1885] -0.00665 -0.14277 -1.0145 0.58489 1.30612 ...  
## $ Impulsive: num [1:1885] -0.217 -0.711 -1.38 -1.38 -0.217 ...  
## $ SS : num [1:1885] -1.181 -0.216 0.401 -1.181 -0.216 ...  
## $ Alcohol : Factor w/ 2 levels "Yes","No": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Amphet : Factor w/ 2 levels "Yes","No": 1 1 2 2 2 2 2 2 2 2 ...  
## $ Amyl : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 1 ...  
## $ Benzos : Factor w/ 2 levels "Yes","No": 1 2 2 1 2 2 2 2 2 2 ...  
## $ Caff : Factor w/ 2 levels "Yes","No": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Cannabis : Factor w/ 2 levels "No","Yes": 1 2 2 2 2 1 1 1 1 1 ...  
## $ Choc : Factor w/ 2 levels "Yes","No": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Coke : Factor w/ 2 levels "No","Yes": 1 2 1 2 1 1 1 1 1 1 ...  
## $ Crack : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Ecstasy : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 1 ...  
## $ Heroin : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Ketamine : Factor w/ 2 levels "No","Yes": 1 2 1 2 1 1 1 1 1 1 ...  
## $ Legalh : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ LSD : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 1 ...  
## $ Meth : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 1 ...  
## $ Mushrooms: Factor w/ 2 levels "No","Yes": 1 1 1 1 2 1 1 1 1 1 ...  
## $ Nicotine : Factor w/ 2 levels "Yes","No": 1 1 2 1 1 1 1 2 1 1 ...  
## $ Semer : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ VSA : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...

## We’ll focus on Nicotine use, so let’s get rid of the remaining drug use variables. We’ll use select for this.

drug\_clean = drug\_clean %>% select(!(Alcohol:Mushrooms)) %>%  
 select(!(Semer:VSA))

## Question 1: Check for missing data in our “drug\_clean” dataframe.

## True/False: There is missingness in the dataset.

### False

which(is.na(drug\_clean))

## integer(0)

## Question 2: Split the dataset into training (70%) and testing (30%) sets. Use a set.seed of 1234. Stratify by the “Nicotine” variable.

## How many rows are in the training set?

### 1318

set.seed(1234)   
drug\_clean\_split = initial\_split(drug\_clean, prop = 0.7, strata = Nicotine) #70% in training  
train = training(drug\_clean\_split)  
test = testing(drug\_clean\_split)  
  
nrow(train)

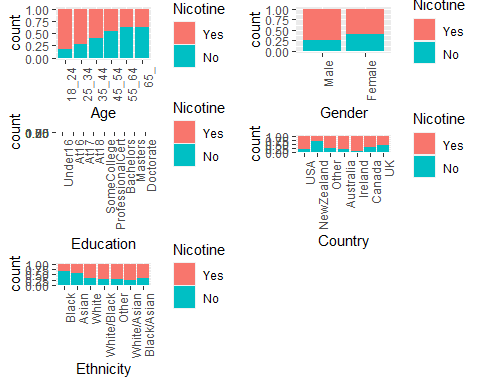
## [1] 1318

## Question 3: Create appropriate visualizations (12 in all) to examine the relationships between each variable and “Nicotine”. Use grid.arrange (from the gridExtra package) to organize these visuals (perhaps in groups of four visualizations?).

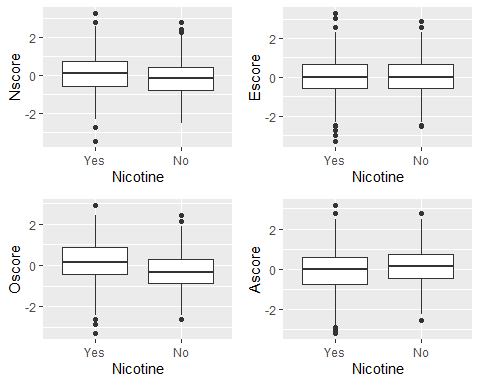
## True/False: Individuals in the 18-24 age group are proportionally more likely to be Nicotine users than not.

### TRUE

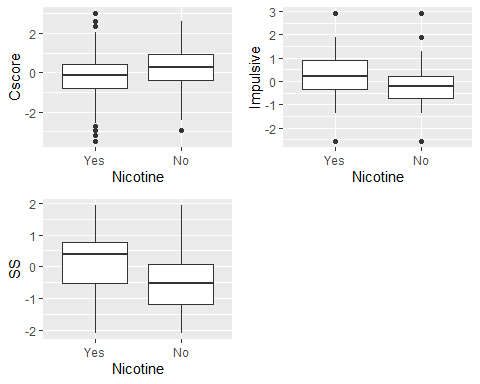
p1 = ggplot(train, aes(x = Age, fill = Nicotine)) +  
 geom\_bar(position = "fill") +  
 theme(axis.text.x=element\_text(angle=90, hjust=1))  
  
p2 = ggplot(train, aes(x = Gender, fill = Nicotine)) +  
 geom\_bar(position = "fill") +  
 theme(axis.text.x=element\_text(angle=90, hjust=1))  
  
p3 = ggplot(train, aes(x = Education, fill = Nicotine)) +  
 geom\_bar(position = "fill") +  
 theme(axis.text.x=element\_text(angle=90, hjust=1))  
  
p4 = ggplot(train, aes(x = Country, fill = Nicotine)) +  
 geom\_bar(position = "fill") +  
 theme(axis.text.x=element\_text(angle=90, hjust=1))  
  
p5 = ggplot(train, aes(x = Ethnicity, fill = Nicotine)) +  
 geom\_bar(position = "fill") +  
 theme(axis.text.x=element\_text(angle=90, hjust=1))  
  
grid.arrange(p1,p2,p3,p4,p5)



p6 = ggplot(train, aes(x = Nicotine, y = Nscore)) +  
 geom\_boxplot()  
  
p7 = ggplot(train, aes(x = Nicotine, y = Escore)) +  
 geom\_boxplot()  
  
p8 = ggplot(train, aes(x = Nicotine, y = Oscore)) +  
 geom\_boxplot()  
  
p9 = ggplot(train, aes(x = Nicotine, y = Ascore)) +  
 geom\_boxplot()  
  
grid.arrange(p6,p7,p8,p9, ncol = 2)



p10 = ggplot(train, aes(x = Nicotine, y = Cscore)) +  
 geom\_boxplot()  
  
p11 = ggplot(train, aes(x = Nicotine, y = Impulsive)) +  
 geom\_boxplot()  
  
p12 = ggplot(train, aes(x = Nicotine, y = SS)) +  
 geom\_boxplot()  
  
grid.arrange(p10,p11,p12, ncol = 2)



## Question 4: True/False: Individuals with higher “Impulsive” scores more likely to be Nicotine users than not.

### TRUE

## Question 5: Create a random forest model (using the ranger package) on the training set to predict Nicotine using all of the variables in the dataset. You 5-fold, k-fold cross-validation (random number seed of 123 for the folds). Allow R to select mtry values between 2 and 8 and min\_n values between 5 and 20. Use 10 levels in your “grid\_regular” function. Set a random number seed of 123 for the tune\_grid function. Use 100 trees.

## NOTE: This model make take a few minutes to run. Be patient. Visualize the relationships between parameters and performance metrics.

## The highest accuracy in this visualization is just greater than which value:

## A. 0.725

## B. 0.730

## C. 0.720

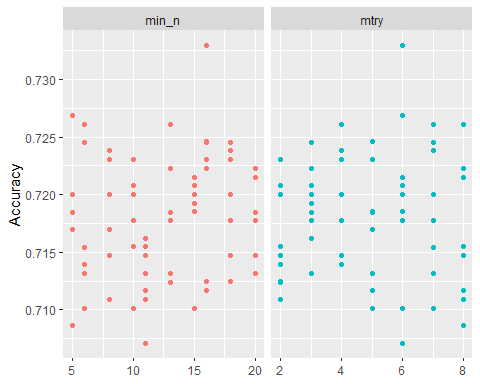
## D. 0.715

### B.

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

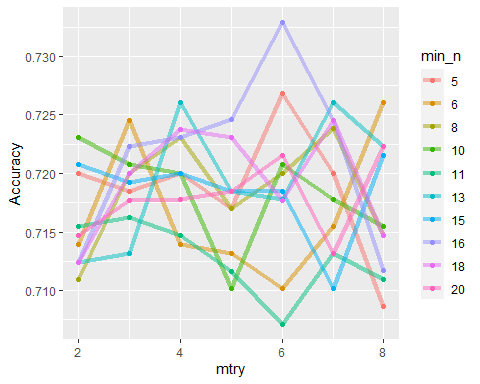
train\_recipe = recipe(Nicotine ~., train) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes())  
  
train\_rf\_model = rand\_forest(mtry = tune(), min\_n = tune(), trees = 100) %>%   
 set\_engine("ranger", importance = "permutation") %>% #added importance metric  
 set\_mode("classification")  
  
train\_wflow =   
 workflow() %>%   
 add\_model(train\_rf\_model) %>%   
 add\_recipe(train\_recipe)  
  
train\_rf\_grid = grid\_regular(  
 mtry(range = c(2,8)),  
 min\_n(range = c(5,20)),   
 levels = 10  
)  
  
set.seed(123)  
rf\_res\_tuned = tune\_grid(  
 train\_wflow,  
 resamples = rf\_folds,  
 grid = train\_rf\_grid #use the tuning grid  
)

rf\_res\_tuned %>%  
 collect\_metrics() %>%  
 filter(.metric == "accuracy") %>%  
 select(mean, min\_n, mtry) %>%  
 pivot\_longer(min\_n:mtry,  
 values\_to = "value",  
 names\_to = "parameter"  
 ) %>%  
 ggplot(aes(value, mean, color = parameter)) +  
 geom\_point(show.legend = FALSE) +  
 facet\_wrap(~parameter, scales = "free\_x") +  
 labs(x = NULL, y = "Accuracy")

 # OR

rf\_res\_tuned %>%  
 collect\_metrics() %>%  
 filter(.metric == "accuracy") %>%  
 mutate(min\_n = factor(min\_n)) %>%  
 ggplot(aes(mtry, mean, color = min\_n)) +  
 geom\_line(alpha = 0.5, size = 1.5) +  
 geom\_point() +  
 labs(y = "Accuracy")

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## ℹ Please use `linewidth` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



summary(rf\_res\_tuned)

## splits.Length splits.Class splits.Mode id   
## 4 vfold\_split list Length:5   
## 4 vfold\_split list Class :character   
## 4 vfold\_split list Mode :character   
## 4 vfold\_split list   
## 4 vfold\_split list   
## .metrics.Length .metrics.Class .metrics.Mode  
## 6 tbl\_df list   
## 6 tbl\_df list   
## 6 tbl\_df list   
## 6 tbl\_df list   
## 6 tbl\_df list   
## .notes.Length .notes.Class .notes.Mode  
## 3 tbl\_df list   
## 3 tbl\_df list   
## 3 tbl\_df list   
## 3 tbl\_df list   
## 3 tbl\_df list

## Question 6: Use the best mtry and min\_n values from Question 5 to finalize the workflow and fit the model to training set. Examine variable importance.

## Which variable is most important?

## A. Oscore

## B. Cscore

## C. Impulsive

## D. SS

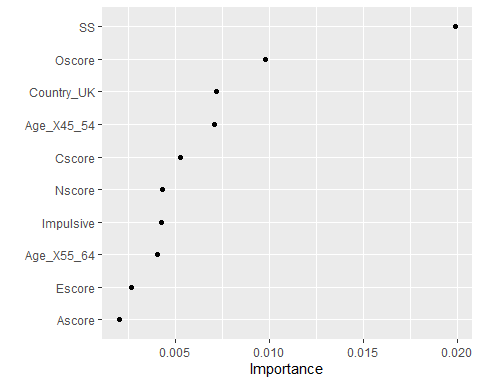
### D.

best\_rf = select\_best(rf\_res\_tuned, "accuracy")  
  
final\_rf = finalize\_workflow(  
 train\_wflow,  
 best\_rf  
)  
  
final\_rf

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 1 Recipe Step  
##   
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = 6  
## trees = 100  
## min\_n = 16  
##   
## Engine-Specific Arguments:  
## importance = permutation  
##   
## Computational engine: ranger

#fit the finalized workflow to our training data  
final\_rf\_fit = fit(final\_rf, train)  
  
final\_rf\_fit %>% pull\_workflow\_fit() %>% vip(geom = "point")

## Warning: `pull\_workflow\_fit()` was deprecated in workflows 0.2.3.  
## ℹ Please use `extract\_fit\_parsnip()` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



## Question 7: To four decimal places, what is the accuracy of your model on the training set?

### 0.9165

## Question 8: To four decimal places, what is the naive accuracy (training set)?

### 0.6707

## Question 9: To four decimal places, what is your model’s accuracy on the testing set?

### 0.6966

trainpredrf = predict(final\_rf\_fit, train)  
head(trainpredrf)

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

confusionMatrix(trainpredrf$.pred\_class, train$Nicotine,   
 positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 871 97  
## No 13 337  
##   
## Accuracy : 0.9165   
## 95% CI : (0.9003, 0.9309)  
## No Information Rate : 0.6707   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8013   
##   
## Mcnemar's Test P-Value : 2.498e-15   
##   
## Sensitivity : 0.9853   
## Specificity : 0.7765   
## Pos Pred Value : 0.8998   
## Neg Pred Value : 0.9629   
## Prevalence : 0.6707   
## Detection Rate : 0.6608   
## Detection Prevalence : 0.7344   
## Balanced Accuracy : 0.8809   
##   
## 'Positive' Class : Yes   
##

testpredrf = predict(final\_rf\_fit, test)  
head(testpredrf)

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

confusionMatrix(testpredrf$.pred\_class, test$Nicotine,   
 positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 327 119  
## No 53 68  
##   
## Accuracy : 0.6966   
## 95% CI : (0.657, 0.7343)  
## No Information Rate : 0.6702   
## P-Value [Acc > NIR] : 0.09699   
##   
## Kappa : 0.2462   
##   
## Mcnemar's Test P-Value : 7.188e-07   
##   
## Sensitivity : 0.8605   
## Specificity : 0.3636   
## Pos Pred Value : 0.7332   
## Neg Pred Value : 0.5620   
## Prevalence : 0.6702   
## Detection Rate : 0.5767   
## Detection Prevalence : 0.7866   
## Balanced Accuracy : 0.6121   
##   
## 'Positive' Class : Yes   
##

## Question 10 The difference in accuracy on the training and testing sets implies?

## A. Overfitting does not appear to be occurring

## B. Overfitting is likely occurring

## C. It is not clear whether or not overfitting is occurring

### A