Predicting Applicant from Inquiries

# Libraries

list.of.packages <- c("readxl", "dplyr", "tidyr", "mosaic", "ggplot2", "ggcorrplot", "rpart","rpart.plot", "caret", "e1071", "randomForest", "xgboost", "neuralnet", "janitor", "stringr","openxlsx","rlang","lubridate","anytime")  
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]  
if(length(new.packages)) install.packages(new.packages)  
  
library(stringr)  
library(readxl)

## Warning: package 'readxl' was built under R version 4.1.2

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(tidyr)  
library(mosaic)

## Registered S3 method overwritten by 'mosaic':  
## method from   
## fortify.SpatialPolygonsDataFrame ggplot2

##   
## The 'mosaic' package masks several functions from core packages in order to add   
## additional features. The original behavior of these functions should not be affected by this.

##   
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':  
##   
## mean

## The following object is masked from 'package:ggplot2':  
##   
## stat

## The following objects are masked from 'package:dplyr':  
##   
## count, do, tally

## The following objects are masked from 'package:stats':  
##   
## binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,  
## quantile, sd, t.test, var

## The following objects are masked from 'package:base':  
##   
## max, mean, min, prod, range, sample, sum

library(ggplot2)  
library(ggcorrplot)  
library(rpart)  
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.1.2

library(caret)

## Warning: package 'caret' was built under R version 4.1.2

##   
## Attaching package: 'caret'

## The following object is masked from 'package:mosaic':  
##   
## dotPlot

library(e1071)

## Warning: package 'e1071' was built under R version 4.1.2

library(randomForest)

## Warning: package 'randomForest' was built under R version 4.1.2

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

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

library(xgboost)

## Warning: package 'xgboost' was built under R version 4.1.2

##   
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':  
##   
## slice

library(neuralnet)

## Warning: package 'neuralnet' was built under R version 4.1.2

##   
## Attaching package: 'neuralnet'

## The following object is masked from 'package:dplyr':  
##   
## compute

library(janitor)

## Warning: package 'janitor' was built under R version 4.1.2

##   
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':  
##   
## chisq.test, fisher.test

library(openxlsx)

## Warning: package 'openxlsx' was built under R version 4.1.2

library(rlang)  
library(lubridate)

## Warning: package 'lubridate' was built under R version 4.1.2

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(anytime)

## Warning: package 'anytime' was built under R version 4.1.2

# Load and clean the data

ZIP <- read.csv("Zip code Census.csv") # ZIP CODE  
file.list <- data.frame(list.files(pattern = '\*.xlsx') ) # Create dataframe of all excel files   
file.list <- data.frame(list.files(pattern = '\*.xlsx') ) %>% # Double do it bc it fixed an error  
 filter(str\_detect(file.list$list.files.pattern......xlsx..,"Pred")) # Filter so we only have the predicted fields  
Pred\_one <- read\_excel(file.list[[1,1]]) # First file we want to predict

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P1428 / R1428C16: got 'Raise.Me'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P2203 / R2203C16: got 'NACAC'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P7221 / R7221C16: got 'NACAC'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P8759 / R8759C16: got 'PCU'

Pred\_two <- read\_excel(file.list[[2,1]]) # Second  
Pred\_three <- read\_excel(file.list[[3,1]]) # Third

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P1032 / R1032C16: got 'Raise.Me'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P4133 / R4133C16: got 'NACAC'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P5213 / R5213C16: got 'Raise.Me'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P5305 / R5305C16: got 'Raise.Me'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P10933 / R10933C16: got 'Raise.Me'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P12468 / R12468C16: got 'NACAC'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P13702 / R13702C16: got 'Raise.Me'

## Warning in read\_fun(path = enc2native(normalizePath(path)), sheet\_i = sheet, :  
## Expecting logical in P14529 / R14529C16: got 'Raise.Me'

Fall\_2022 <- bind\_rows(Pred\_one,Pred\_two,Pred\_three) # Merge them  
file.list2 <- data.frame(list.files(pattern = '\*.txt') ) # Same thing as before but for .txt  
file.list2 <- data.frame(list.files(pattern = '\*.txt') ) %>% # Fixed Error  
 filter(str\_detect(file.list2$list.files.pattern......txt..,"Past|past")) # Filter for Past files  
Uncleaned\_data <- read.csv(file.list2[[1,1]], sep = "\t",na.strings = "") # Read in the file  
cleaned\_data <- Uncleaned\_data %>%  
 select(Applicant,Sex,Birthdate,Age,IPEDS.Classification,Active.Region,Active.US.5.digit.ZIP.Code,Active.Geomarket,School.Address.US.5.digit.ZIP.Code,Intended.Major,Combined.Inquiry.Date,Ping...Total.Count,Deliver.Statistics.by.Status,Deliver.Statistics...Total,Off.Campus.Visit.Attended,Off.Campus.Visit.Registered,Open.House.Registered,Open.House.Attended,Campus.Tour.Registered,Campus.Tour.Attended,Has.Test.Score,Distance.from.01845)%>% # Only keep specific columns  
filter(Applicant < 3) # Filter out errors in Applicant  
colnames(Fall\_2022)[-c(1,16)] <- colnames(cleaned\_data) # Fix naming errors  
cleaned\_data$Intended.Major[is.na(cleaned\_data$Intended.Major)] = "Undecided/Unknown"   
cleaned\_data$School.Address.US.5.digit.ZIP.Code[is.na(cleaned\_data$School.Address.US.5.digit.ZIP.Code)] = "No Recorded ZIPCode"  
 #Na means undecided or unknown so fixed that   
cleaned\_data <- cleaned\_data %>% #   
 drop\_na(Distance.from.01845) # Drop Na's in Distance   
cleaned\_data <- cleaned\_data %>% #   
 filter(Age > 15) # Filter out unreasonable ages (Outliers)  
cleaned\_data$Sex[is.na(cleaned\_data$Sex)] <- 3 # Na to 3  
cleaned\_data <- cleaned\_data %>% # Convert Gender into numbers  
 mutate(Sex = ifelse(Sex == "F", 0, # Convert F to 0   
ifelse(Sex == "M",1, # M to 1  
 ifelse(Sex != 3, 2,3))))# %>% # Other to 2  
cleaned\_data$Deliver.Statistics.by.Status <- as.numeric(cleaned\_data$Deliver.Statistics.by.Status)  
 # Make as numeric for modeling  
cleaned\_data <- cleaned\_data %>% # Create bins for distance   
 filter(Distance.from.01845 < 500) # No ouliers  
 names <- c("1","2","3","4") # Values   
 b <- c(-Inf,25.1406265 ,54.6694124 ,145.0034910 ,499.3681769 ) # Bins   
cleaned\_data$Distance.from.01845\_bin <- cut(cleaned\_data$Distance.from.01845,breaks = b, labels = names)  
 # Creation of bins  
ZIP <- as.data.frame(t(data.frame(matrix(unlist(ZIP), nrow=length(ZIP), byrow=TRUE))))   
 ZIP<- ZIP %>% # Fix formatting  
 row\_to\_names(row\_number = 1) # Fix formatting conversion errors  
ZIP$`Geographic Area Name` <- gsub("^.{0,6}","",ZIP$`Geographic Area Name`)   
ZIP[1] <- NULL # get rid of column  
ZIP <- ZIP[c(1,2,24)] # Keep specific columns  
ZIP$`Estimate!!Households!!Total` <- as.numeric(ZIP$`Estimate!!Households!!Total`)#Make numeric (this line and below)==========  
ZIP$`Estimate!!Households!!Median income (dollars)` <- as.numeric(ZIP$`Estimate!!Households!!Median income (dollars)`)

## Warning: NAs introduced by coercion

colnames(ZIP ) <- c("Geographic.Area.Name","Household.Total","Household.Median.Income")  
 # Make names simple for Random Forest  
cleaned\_data <- left\_join(cleaned\_data,ZIP,by = c("Active.US.5.digit.ZIP.Code" = "Geographic.Area.Name") )   
cleaned\_data <- na.omit(cleaned\_data) # Combine and Clean Dataset  
  
cleaned\_data$Ping...Total.Count <- log10(cleaned\_data$Ping...Total.Count + 1) # Normalize data  
# IPEDS CLASSIFICATION  
X = model.matrix(~0+IPEDS.Classification, data = cleaned\_data)  
X <- as.data.frame(X)  
X$not.enough.population <- X$`IPEDS.ClassificationTwo or more races` + X$IPEDS.ClassificationOther + X$`IPEDS.ClassificationAmerican Indian or Alaska Native` + X$`IPEDS.ClassificationNonresident Alien` +X$`IPEDS.ClassificationNative Hawaiian or Other Pacific` + X$`IPEDS.ClassificationMulti-Racial`  
X$`IPEDS.ClassificationHispanic of any race` <- X$IPEDS.ClassificationHispanic +X$`IPEDS.ClassificationHispanic of any race`  
X$IPEDS.ClassificationHispanic <- NULL  
X$`IPEDS.ClassificationTwo or more races` <- NULL  
X$IPEDS.ClassificationOther <- NULL  
X$`IPEDS.ClassificationAmerican Indian or Alaska Native` <- NULL  
X$`IPEDS.ClassificationNonresident Alien` <- NULL  
X$`IPEDS.ClassificationNative Hawaiian or Other Pacific` <- NULL  
X$`IPEDS.ClassificationMulti-Racial` <- NULL  
cleaned\_data <- cbind(cleaned\_data,X)  
cleaned\_data$IPEDS.Classification <- NULL  
cleaned\_data$IPEDS.ClassificationRace.Ethnicity.Unknown <- cleaned\_data$`IPEDS.ClassificationRace/Ethnicity Unknown`  
cleaned\_data$`IPEDS.ClassificationRace/Ethnicity Unknown` <- NULL  
cleaned\_data$IPEDS.ClassificationHispanic.of.any.race <- cleaned\_data$`IPEDS.ClassificationHispanic of any race`  
cleaned\_data$`IPEDS.ClassificationHispanic of any race` <- NULL  
cleaned\_data$IPEDS.ClassificationBlack.or.African.American <- cleaned\_data$`IPEDS.ClassificationBlack or African American`  
cleaned\_data$`IPEDS.ClassificationBlack or African American` <- NULL  
# INQUIRY DATE  
cleaned\_data$year = lubridate::year(mdy(cleaned\_data$Combined.Inquiry.Date))  
cleaned\_data$yday = yday(mdy(cleaned\_data$Combined.Inquiry.Date))  
cleaned\_data$quarter = quarter(mdy(cleaned\_data$Combined.Inquiry.Date))  
cleaned\_data$month = lubridate::month(mdy(cleaned\_data$Combined.Inquiry.Date))  
cleaned\_data$day = lubridate::day(mdy(cleaned\_data$Combined.Inquiry.Date))  
cleaned\_data$weekdays = weekdays(anydate(cleaned\_data$Combined.Inquiry.Date))  
cleaned\_data$month = as.factor(cleaned\_data$month)  
cleaned\_data$weekdays = factor(cleaned\_data$weekdays,levels = c("Monday", "Tuesday", "Wednesday","Thursday","Friday","Saturday",'Sunday'))  
cleaned\_data$year = as.factor(cleaned\_data$year)  
cleaned\_data$quarter = as.factor(cleaned\_data$quarter)  
cleaned\_data$week = format(anydate(cleaned\_data$Combined.Inquiry.Date), "%V")  
cleaned\_data$week = as.integer(cleaned\_data$week)  
Year = as.data.frame(model.matrix(~0+year, data = cleaned\_data))  
Quarter = as.data.frame(model.matrix(~0+quarter, data = cleaned\_data))  
Month = as.data.frame(model.matrix(~0+month, data = cleaned\_data))  
cleaned\_data <- cbind(cleaned\_data,Year,Quarter,Month)  
cleaned\_data$year <- NULL  
cleaned\_data$quarter <- NULL  
cleaned\_data$month <- NULL  
  
X = model.matrix(~0+Distance.from.01845\_bin, data = cleaned\_data)  
X <- as.data.frame(X)  
cleaned\_data <- cbind(cleaned\_data,X)  
cleaned\_data$Distance.from.01845\_bin <- NULL  
cleaned\_data$Distance.from.01845 <- NULL  
  
train <- cleaned\_data %>% sample\_frac(size = .75) # Train Test Split   
test <- cleaned\_data %>% setdiff(train)  
  
################ SAME EXACT CODE AS ABOVE BUT ON THE OTHER DATASET #############################  
  
Fall2022\_clean <- Fall\_2022 %>%  
 select(Ref,Applicant,Sex,Birthdate,Age,IPEDS.Classification,Active.Region,Active.US.5.digit.ZIP.Code,Active.Geomarket,School.Address.US.5.digit.ZIP.Code,Intended.Major,Combined.Inquiry.Date,Deliver.Statistics.by.Status,Deliver.Statistics...Total,Off.Campus.Visit.Attended,Off.Campus.Visit.Registered,Open.House.Registered,Open.House.Attended,Campus.Tour.Registered,Campus.Tour.Attended,Has.Test.Score,Distance.from.01845,Ping...Total.Count) %>%  
 filter(Applicant < 3)  
Fall2022\_clean$Intended.Major[is.na(Fall2022\_clean$Intended.Major)] = "Undecided/Unknown"  
Fall2022\_clean$School.Address.US.5.digit.ZIP.Code[is.na(Fall2022\_clean$School.Address.US.5.digit.ZIP.Code)] = "No Recorded ZIPCode"  
Fall2022\_clean <- Fall2022\_clean %>%  
 drop\_na(Distance.from.01845)  
Fall2022\_clean <- Fall2022\_clean %>%  
 filter(Age > 15)   
Fall2022\_clean$Sex[is.na(Fall2022\_clean$Sex)] <- 3  
Fall2022\_clean <- Fall2022\_clean %>%   
 mutate(Sex = ifelse(Sex == "F", 0,   
ifelse(Sex == "M",1,   
 ifelse(Sex != 3, 2,3))))  
Fall2022\_clean$Ping...Total.Count <- as.numeric(Fall2022\_clean$Ping...Total.Count)  
Fall2022\_clean$Ping...Total.Count <- log10(Fall2022\_clean$Ping...Total.Count + 1)  
Fall2022\_clean$Deliver.Statistics.by.Status <- as.numeric(Fall2022\_clean$Deliver.Statistics.by.Status)  
Fall2022\_clean <- Fall2022\_clean%>% filter(Distance.from.01845 < 500)  
 names <- c("1","2","3","4")  
 b <- c(-Inf,25.1406265 ,54.6694124 ,145.0034910 ,499.3681769 )  
Fall2022\_clean$Distance.from.01845\_bin <- cut(Fall2022\_clean$Distance.from.01845,breaks = b, labels = names)  
Fall2022\_clean <- left\_join(Fall2022\_clean,ZIP,by = c("Active.US.5.digit.ZIP.Code" = "Geographic.Area.Name") )  
Fall2022\_clean <- na.omit(Fall2022\_clean)  
X = model.matrix(~0+IPEDS.Classification, data = Fall2022\_clean)  
X <- as.data.frame(X)  
X$not.enough.population <- X$`IPEDS.ClassificationTwo or more races` + X$IPEDS.ClassificationOther + X$`IPEDS.ClassificationAmerican Indian or Alaska Native` + X$`IPEDS.ClassificationNonresident Alien` +X$`IPEDS.ClassificationNative Hawaiian or Other Pacific` + X$`IPEDS.ClassificationMulti-Racial`  
X$`IPEDS.ClassificationHispanic of any race` <- X$IPEDS.ClassificationHispanic +X$`IPEDS.ClassificationHispanic of any race`  
X$IPEDS.ClassificationHispanic <- NULL  
X$`IPEDS.ClassificationTwo or more races` <- NULL  
X$IPEDS.ClassificationOther <- NULL  
X$`IPEDS.ClassificationAmerican Indian or Alaska Native` <- NULL  
X$`IPEDS.ClassificationNonresident Alien` <- NULL  
X$`IPEDS.ClassificationNative Hawaiian or Other Pacific` <- NULL  
X$`IPEDS.ClassificationMulti-Racial` <- NULL  
Fall2022\_clean <- cbind(Fall2022\_clean,X)  
Fall2022\_clean$IPEDS.Classification <- NULL  
Fall2022\_clean$IPEDS.ClassificationRace.Ethnicity.Unknown <- Fall2022\_clean$`IPEDS.ClassificationRace/Ethnicity Unknown`  
Fall2022\_clean$`IPEDS.ClassificationRace/Ethnicity Unknown` <- NULL  
Fall2022\_clean$IPEDS.ClassificationHispanic.of.any.race <- Fall2022\_clean$`IPEDS.ClassificationHispanic of any race`  
Fall2022\_clean$`IPEDS.ClassificationHispanic of any race` <- NULL  
Fall2022\_clean$IPEDS.ClassificationBlack.or.African.American <- Fall2022\_clean$`IPEDS.ClassificationBlack or African American`  
Fall2022\_clean$`IPEDS.ClassificationBlack or African American` <- NULL  
  
Fall2022\_clean$year = lubridate::year(mdy(Fall2022\_clean$Combined.Inquiry.Date))  
Fall2022\_clean$yday = yday(mdy(Fall2022\_clean$Combined.Inquiry.Date))  
Fall2022\_clean$quarter = quarter(mdy(Fall2022\_clean$Combined.Inquiry.Date))  
Fall2022\_clean$month = lubridate::month(mdy(Fall2022\_clean$Combined.Inquiry.Date))  
Fall2022\_clean$day = lubridate::day(mdy(Fall2022\_clean$Combined.Inquiry.Date))  
Fall2022\_clean$weekdays = weekdays(anydate(Fall2022\_clean$Combined.Inquiry.Date))  
Fall2022\_clean$month = as.factor(Fall2022\_clean$month)  
Fall2022\_clean$weekdays = factor(Fall2022\_clean$weekdays,levels = c("Monday", "Tuesday", "Wednesday","Thursday","Friday","Saturday",'Sunday'))  
Fall2022\_clean$year = as.factor(Fall2022\_clean$year)  
Fall2022\_clean$quarter = as.factor(Fall2022\_clean$quarter)  
Fall2022\_clean$week = format(anydate(Fall2022\_clean$Combined.Inquiry.Date), "%V")  
Fall2022\_clean$week = as.integer(Fall2022\_clean$week)  
Year = as.data.frame(model.matrix(~0+year, data = Fall2022\_clean))  
Quarter = as.data.frame(model.matrix(~0+quarter, data = Fall2022\_clean))  
Month = as.data.frame(model.matrix(~0+month, data = Fall2022\_clean))  
Fall2022\_clean <- cbind(Fall2022\_clean,Year,Quarter,Month)  
Fall2022\_clean$year <- NULL  
Fall2022\_clean$quarter <- NULL  
Fall2022\_clean$month <- NULL  
  
X = model.matrix(~0+Distance.from.01845\_bin, data = Fall2022\_clean)  
X <- as.data.frame(X)  
Fall2022\_clean <- cbind(Fall2022\_clean,X)  
Fall2022\_clean$Distance.from.01845\_bin <- NULL  
Fall2022\_clean$Distance.from.01845 <- NULL

# Random Forest Model

Random\_Forest <- randomForest(as.factor(Applicant) ~., data = train, ntree = 800, mtry = 6, nodesize = 14, maxnodes = 24)  
 # Creation of Random Forest Model  
prediction1 <- predict(Random\_Forest, test[-c(1)],type = "prob")[,2] # Probabilities from test   
prediction1[prediction1 < .65] = 0 # Set threshold  
prediction1[prediction1 >= .65] = 1   
confusion <- confusionMatrix(table(test$Applicant, prediction1)) # Confusion Matrix  
confusion

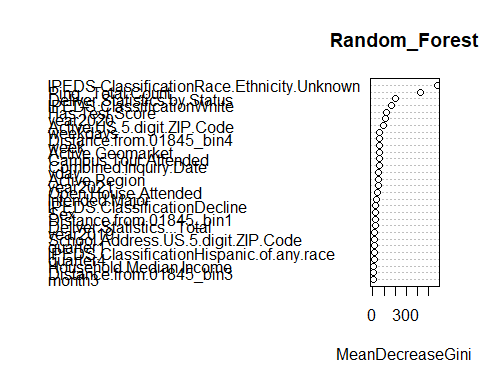
## Confusion Matrix and Statistics  
##   
## prediction1  
## 0 1  
## 0 4033 66  
## 1 1327 710  
##   
## Accuracy : 0.773   
## 95% CI : (0.7623, 0.7834)  
## No Information Rate : 0.8735   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.3938   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.7524   
## Specificity : 0.9149   
## Pos Pred Value : 0.9839   
## Neg Pred Value : 0.3486   
## Prevalence : 0.8735   
## Detection Rate : 0.6573   
## Detection Prevalence : 0.6680   
## Balanced Accuracy : 0.8337   
##   
## 'Positive' Class : 0   
##

prediction <-predict(Random\_Forest, Fall2022\_clean[-c(1,2)],type = "prob")[,2] # Probabilities from Actual data   
prediction[prediction < .65] = 0 # Threshold  
prediction[prediction >= .65] = 1  
confusion2 <- confusionMatrix(table(Fall2022\_clean$Applicant, prediction)) # Confusion Matrix  
confusion2

## Confusion Matrix and Statistics  
##   
## prediction  
## 0 1  
## 0 18987 686  
## 1 1273 7818  
##   
## Accuracy : 0.9319   
## 95% CI : (0.9289, 0.9348)  
## No Information Rate : 0.7044   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8397   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.9372   
## Specificity : 0.9193   
## Pos Pred Value : 0.9651   
## Neg Pred Value : 0.8600   
## Prevalence : 0.7044   
## Detection Rate : 0.6601   
## Detection Prevalence : 0.6839   
## Balanced Accuracy : 0.9282   
##   
## 'Positive' Class : 0   
##

# Model Analysis for report

varImpPlot(Random\_Forest)



Import <- as.data.frame(importance(Random\_Forest))  
Import %>%  
 mutate(MeanDecreaseGini = format(MeanDecreaseGini, scientific = FALSE)) %>%  
 arrange(desc(MeanDecreaseGini))

## MeanDecreaseGini  
## IPEDS.ClassificationRace.Ethnicity.Unknown 574.37152208  
## Ping...Total.Count 428.07328229  
## Deliver.Statistics.by.Status 201.53402884  
## IPEDS.ClassificationWhite 163.41998862  
## Has.Test.Score 126.14600044  
## year2020 112.01102279  
## Active.US.5.digit.ZIP.Code 96.51755664  
## weekdays 62.14284763  
## Distance.from.01845\_bin4 61.30915680  
## week 57.78965261  
## Active.Geomarket 56.25732572  
## Campus.Tour.Attended 55.96842021  
## Combined.Inquiry.Date 55.53555549  
## yday 52.84695025  
## Active.Region 51.03561615  
## year2021 47.77250699  
## Open.House.Attended 37.59341146  
## Intended.Major 32.20856056  
## IPEDS.ClassificationDecline 23.94901178  
## Sex 20.02313273  
## Distance.from.01845\_bin1 19.88372151  
## Deliver.Statistics...Total 18.95901193  
## year2019 15.94952518  
## School.Address.US.5.digit.ZIP.Code 14.91319017  
## quarter1 13.36709040  
## IPEDS.ClassificationHispanic.of.any.race 12.06477767  
## quarter4 10.55835801  
## Household.Median.Income 4.91480436  
## Distance.from.01845\_bin3 4.79635573  
## month3 3.65302518  
## Distance.from.01845\_bin2 3.23336203  
## not.enough.population 3.07279390  
## Off.Campus.Visit.Attended 3.05269544  
## Birthdate 2.77363435  
## month12 2.74528174  
## Campus.Tour.Registered 2.51736603  
## day 2.27456640  
## quarter3 1.80765489  
## Household.Total 1.79480095  
## month10 1.46814826  
## Open.House.Registered 1.31830584  
## month1 1.10804658  
## IPEDS.ClassificationBlack.or.African.American 1.00162197  
## month11 0.76714007  
## quarter2 0.71441649  
## Age 0.64066989  
## month4 0.59213315  
## year2018 0.44811224  
## month2 0.44635880  
## month8 0.43612691  
## month9 0.42790144  
## IPEDS.ClassificationAsian 0.31745170  
## month6 0.13078117  
## month7 0.05766141  
## month5 0.05347346  
## year2016 0.02500993  
## Off.Campus.Visit.Registered 0.01916392  
## year2017 0.01838459  
## year2015 0.01184478

Import$Type <- row.names(Import)  
rownames(Import) <- NULL

# Exporting Results

Ref <- data.frame("Ref" = Fall\_2022$Ref, "Actually Applied" = Fall\_2022$Applicant )  
Probabilities <- data.frame("Ref" = Fall2022\_clean$Ref, "Application Prediction" = prediction, "Application Probability" = predict(Random\_Forest, Fall2022\_clean, type = "prob")[,2])  
Final\_Dataset <- left\_join(Ref,Probabilities) %>%  
 arrange(desc(Application.Probability))

## Joining, by = "Ref"

Final\_Dataset$Application.Prediction[is.na(Final\_Dataset$Application.Prediction)] = "Not.enough.data"  
Final\_Dataset <- left\_join(Final\_Dataset,Ref)

## Joining, by = c("Ref", "Actually.Applied")

Final\_Dataset$Application.Probability[is.na(Final\_Dataset$Application.Probability)] = "Not.enough.data"  
write.csv(Final\_Dataset, "Fall\_2022\_Predicted.csv")  
write.csv(Import, "Variable Importance.csv")  
  
sink(file = "Prediction Statistics")  
cat("The measurements to evalute a model changes each time new data is presented, Which is why this document exists. \nIn order relpicate the evaluation of this model, I will showcase the most important statistics to be aware of. \n \n")

## The measurements to evalute a model changes each time new data is presented, Which is why this document exists.   
## In order relpicate the evaluation of this model, I will showcase the most important statistics to be aware of.   
##

print(as.matrix(confusion, what = "classes"))

## [,1]  
## Sensitivity 0.7524254  
## Specificity 0.9149485  
## Pos Pred Value 0.9838985  
## Neg Pred Value 0.3485518  
## Precision 0.9838985  
## Recall 0.7524254  
## F1 0.8527328  
## Prevalence 0.8735332  
## Detection Rate 0.6572686  
## Detection Prevalence 0.6680248  
## Balanced Accuracy 0.8336869

cat("\nTotal Predicted Applicant:")

##   
## Total Predicted Applicant:

print(sum(as.numeric(prediction)))

## [1] 8504

cat("\n\nThe key takeaways here should be Balanced Accuarcy, Precision, and Recall \nBalanced Accuracy: This value is the overall accuracy of the predictions. \nPrecision: Out of all that are predicted to apply, what percent of those actually apply.\nRecall: Out of all those who do apply, how many are predicted to apply.\n\nSummary:\nIn terms of Balanced Accuracy, a Balanced Accuracy greater than 75% is significant, yet for this model more can \nbe found from the Precision and Recall. I believe there is more meaning behind a higher Precision. If Precision is \nhigh we can assume when a inquiry is predicted to apply, they will apply. This gives us definite answer of the \nminimum amount of applicant Merrimack will recieve. While we can be certain we have a high precision, how many \ninquiries that will apply did we miss? This is what Recall evaluates. A lower Recall means we have missed a lot of \nApplicants while a higher Recall means we did not miss many potential Applicants. (In this case a lower recall \ndoesn't mean a statistically insignificant model but rather there are more inquiries to applicants we missed). \n\n  
   
When the model predicts an inquiry on file to be a future applicant, I recommend putting EXTRA effort into them. Especially if they\n haven't applied yet (Recall). Since we have high precision, they SHOULD be applying, yet they have not. I recommend any actions taken towards \n the pursuit be documented and quantitized as I would love to analyze such impacts our actions have towards the application rate! ")

##   
##   
## The key takeaways here should be Balanced Accuarcy, Precision, and Recall   
## Balanced Accuracy: This value is the overall accuracy of the predictions.   
## Precision: Out of all that are predicted to apply, what percent of those actually apply.  
## Recall: Out of all those who do apply, how many are predicted to apply.  
##   
## Summary:  
## In terms of Balanced Accuracy, a Balanced Accuracy greater than 75% is significant, yet for this model more can   
## be found from the Precision and Recall. I believe there is more meaning behind a higher Precision. If Precision is   
## high we can assume when a inquiry is predicted to apply, they will apply. This gives us definite answer of the   
## minimum amount of applicant Merrimack will recieve. While we can be certain we have a high precision, how many   
## inquiries that will apply did we miss? This is what Recall evaluates. A lower Recall means we have missed a lot of   
## Applicants while a higher Recall means we did not miss many potential Applicants. (In this case a lower recall   
## doesn't mean a statistically insignificant model but rather there are more inquiries to applicants we missed).   
##   
##   
##   
## When the model predicts an inquiry on file to be a future applicant, I recommend putting EXTRA effort into them. Especially if they  
## haven't applied yet (Recall). Since we have high precision, they SHOULD be applying, yet they have not. I recommend any actions taken towards   
## the pursuit be documented and quantitized as I would love to analyze such impacts our actions have towards the application rate!

sink(file = NULL)