D209: Data Mining I

Task 2

Lucas Benish

Western Governors University

# Part 1: Research Question

# Describe the purpose of this data mining report by doing the following:

## Propose **one** question relevant to a real-world organizational situation that you will answer using **one** of the following prediction methods:

## decision trees

## random forests

## advanced regression (i.e., lasso or ridge regression)

The business question that I am interested in answering is “Can we predict how many days a patient will stay at the hospital based on comorbidities/other patient factors?” I intend to use random forest regression for this task due to the continuous nature of the explanatory variable and the ability of random forests to create robust models.

## Define **one** goal of the data analysis. Ensure that your goal is reasonable within the scope of the scenario and is represented in the available data.

One goal of my analysis is to create a prediction model using a random forest to determine how many days a patient will stay in the hospital that is not overfit to the data as my linear model was in D208. This will be accomplished via splitting the data into training and test datasets and utilizing cross-validation on the training split. This is well within the scope of the scenario, as data regarding length of stay (Initial\_days) and many patient factors are available within the dataset.

# Part II: Method Justification

# Explain the reasons for your chosen classification method from part A1 by doing the following:

## Explain how the prediction method you chose analyzes the selected data set. Include expected outcomes.

I chose random forest as my prediction model due to random forests being able to create strong prediction models and the flexibility the model allows due to tuning. The random forest model will create random decision trees which will then be combined to create an ensemble model. The outputs of the random forest will be average leaves/decisions based on all inputs (Yiu, 2019). The random forest will accomplish this using bootstrap aggregation, where each decision tree within the random forest is created from a random sample of the data, and each tree is then made based upon random features, or split decisions, so that each tree is made from different features. For the overall ensemble, the final leaf nodes will be determined based on averages/majority rule.

## Summarize **one** assumption of the chosen prediction method

One assumption of a random forest model is that there is not a high level of correlation between explanatory variables (Yiu, 2019). Like linear regression models, if there is a high level of collinearity between explanatory variables and/or the response variable, some explanatory variables may carry more weight in the model or be redundant.

## List the packages or libraries you have chosen for Python or R, and justify how *each* item on the list supports the analysis

The libraries that I have chosen to use within R are as follows:

readr: used for importing dataset from csv

dplyr: utilized for easier syntax, piping functions/code

ggplot2: used for visualizing relationships between explanatory variables

visdat: used for visualizing missingness during exploration

naniar: used for cleaning missingness found during exploration

caret: used for performing regression analysis with cross validation

ranger: used for performing random forest regression

# Part III: Data Preparation

# Perform data preparation for the chosen data set by doing the following:

## Describe **one** data preprocessing goal relevant to the prediction method from part A1.

One data preprocessing goal for this task is to make sure that none of my explanatory variables have a strong, linear relationship with each other. This is an assumption of random forest regression, that none of the variables are strongly correlated to each other.

## Identify the initial data set variables that you will use to perform the analysis for the prediction question from part A1, and group *each* variable as continuous or categorical.

The variables that I am going to include within my model have been chosen based on the significant variables noted in D208 Task 1 – Linear Regression. Because a reduced linear regression model was already created using just those variables determined to be significant, they will be used as a basis for this model. This will eliminate the need to visualize the relationships between every potential explanatory variable and allow me to just focus on these variables.

The variables that I have chosen to include, with their designation after, are as follows:

Explanatory variables:

ReAdmis: categorical

Initial\_admin: categorical

Complication\_risk: categorical

Arthritis: categorical

Anxiety: categorical

Allergic\_rhinitis: categorical

HighBlood: categorical

Full\_meals\_eaten: continuous

The response variable I am exploring is:

Initial\_days: continuous

## Explain the steps used to prepare the data for the analysis. Identify the code segment for *each* step.

Similarly to D209 Task 1, much of the cleaning process is a standard cleaning process. Pre-processing for this task is minimal until the model is being built, at which point hyperparameter tuning will take place. The initial steps include establishing the R studio environment, importing the dataset, visualizing the missingness of the data and treating any missingness, and determining presence of duplicate values. This was accomplished with:

#setting up environment

library(dplyr)

library(ggplot2)

library(naniar)

library(visdat)

library(readr)

library(ranger)

library(caret)

#import dataset

medical\_clean <- read\_csv("C:/Users/lgben/OneDrive/Desktop/MSDA/D209 - Data Mining I/Task 2 - Predictive Analysis/medical\_clean.csv")

View(medical\_clean)

#assessing missingness and duplicates

vis\_miss(medical\_clean)

str(medical\_clean)

sum(duplicated(medical\_clean))

#general glimpse of dataset

summary(medical\_clean)

#copy data set for outliers if needed

medical\_clean2 <- medical\_clean

The packages that I included are described above. It was noted during the above code section that no missingness nor duplicates were present within my dataset. I then determined the presence of outliers and treated the outliers. This was accomplished via the following code:

#z-score columns

medical\_clean2$children\_z <- scale(x=medical\_clean2$Children)

medical\_clean2$age\_z <- scale(x=medical\_clean2$Age)

medical\_clean2$income\_z <- scale(x=medical\_clean2$Income)

medical\_clean2$vitd\_levels\_z <- scale(x=medical\_clean2$VitD\_levels)

medical\_clean2$doc\_visits\_z <- scale(x=medical\_clean2$Doc\_visits)

medical\_clean2$full\_meals\_eaten\_z <- scale(x=medical\_clean2$Full\_meals\_eaten)

medical\_clean2$vitd\_supp\_z <- scale(x=medical\_clean2$vitD\_supp)

medical\_clean2$initial\_days\_z <- scale(x=medical\_clean2$Initial\_days)

medical\_clean2$totalcharge\_z <- scale(x=medical\_clean2$TotalCharge)

medical\_clean2$additional\_charges\_z <- scale(x=medical\_clean2$Additional\_charges)

#outlier vectors

children\_outliers <- which(medical\_clean2$children\_z >3 | medical\_clean2$children\_z < -3)

age\_outliers <- which(medical\_clean2$age\_z >3 | medical\_clean2$age\_z < -3)

income\_outliers <- which(medical\_clean2$income\_z >3 | medical\_clean2$income\_z < -3)

vitd\_levels\_outliers <- which(medical\_clean2$vitd\_levels\_z >3 | medical\_clean2$vitd\_levels\_z < -3)

doc\_visits\_outliers <- which(medical\_clean2$doc\_visits\_z >3 | medical\_clean2$doc\_visits\_z < -3)

full\_meals\_eaten\_outliers <- which(medical\_clean2$full\_meals\_eaten\_z >3 | medical\_clean2$full\_meals\_eaten\_z < -3)

vitd\_supp\_outliers <- which(medical\_clean2$vitd\_supp\_z >3 | medical\_clean2$vitd\_supp\_z < -3)

initial\_days\_outliers <- which(medical\_clean2$initial\_days\_z >3 | medical\_clean2$initial\_days\_z < -3)

total\_charge\_outliers <- which(medical\_clean2$totalcharge\_z >3 | medical\_clean2$totalcharge\_z < -3)

additional\_charges\_outliers <- which(medical\_clean2$additional\_charges\_z >3 | medical\_clean2$additional\_charges\_z < -3)

#treating outliers

unique\_outliers <- unique(c(children\_outliers, doc\_visits\_outliers, full\_meals\_eaten\_outliers, income\_outliers, vitd\_levels\_outliers, vitd\_supp\_outliers))

medical\_clean3 <- medical\_clean2[-unique\_outliers, ]

medical\_clean3 <- subset(medical\_clean3, select = -c(age\_z, income\_z, vitd\_levels\_z, doc\_visits\_z, full\_meals\_eaten\_z, vitd\_supp\_z, initial\_days\_z, totalcharge\_z, additional\_charges\_z, children\_z))

#removing outlier vectors/objects from environment

remove(additional\_charges\_outliers, age\_outliers, children\_outliers, doc\_visits\_outliers, full\_meals\_eaten\_outliers, income\_outliers, initial\_days\_outliers, total\_charge\_outliers, unique\_outliers, vitd\_levels\_outliers, vitd\_supp\_outliers)

It was determined during the above code section that there were 467 outliers present within the dataset. These were determined via z-scores. Anything above 3 standard deviations or below -3 standard deviations from the mean were added to a vector of “unique outliers.” This vector included all observations that contained at least one outlier. Next, all outliers were removed. Because the number of outliers totaled less than 5% of the overall dataset, I decided to simply remove them all. My goal was not to introduce personal bias by deciding to keep some outliers while removing others. After outliers were removed from the dataset, the z-score columns and vectors relating to outliers were removed from the programming environment. With removal of the z-score columns, a new data frame was again created.

After removal of the z-score columns and additional vectors, I decided to remove the columns containing variables that would not be a part of the random forest modeling process. The variables that will be included are those included in D208 Task 1’s reduced linear model. This reduced linear model contained only variables that were determined to be statistically significant within the initial linear model that included all explanatory variables. This decision was made to reduce the processing power, and time to create the model, required for modeling with the random forest. These columns were removed with the following code:

#removing unnecessary explanatory variables

medical\_clean4 <- subset(medical\_clean3, select = c(ReAdmis, Initial\_admin, Complication\_risk, Arthritis, Anxiety, Allergic\_rhinitis, HighBlood, Full\_meals\_eaten, Initial\_days))

Following removal of the non-significant explanatory variables, I proceeded to visualize the remaining explanatory variables and response variable. I did this with univariate visualizations, as well as bivariate visualizations. No large correlations were noted between any of the explanatory variables and the response variable, which is an assumption of the random forest model. This was accomplished by the following code:

#visualizing only those pertinent variables from prior linear regression

#univariate viz

ggplot(medical\_clean4, aes(x=Initial\_days)) + geom\_histogram(binwidth = 1)

ggplot(medical\_clean4, aes(x=ReAdmis)) + geom\_bar()

ggplot(medical\_clean4, aes(x=Initial\_admin)) + geom\_bar()

ggplot(medical\_clean4, aes(x=Complication\_risk)) + geom\_bar()

ggplot(medical\_clean4, aes(x=Arthritis)) + geom\_bar()

ggplot(medical\_clean4, aes(x=Anxiety)) + geom\_bar()

ggplot(medical\_clean4, aes(x=Allergic\_rhinitis)) + geom\_bar()

ggplot(medical\_clean4, aes(x=HighBlood)) + geom\_bar()

ggplot(medical\_clean4, aes(x=Full\_meals\_eaten)) + geom\_histogram(binwidth=1)

#bivariate viz

ggplot(medical\_clean4, aes(x=ReAdmis, y=Initial\_days)) + geom\_boxplot()

ggplot(medical\_clean4, aes(x=Initial\_admin, y=Initial\_days)) +geom\_boxplot()

ggplot(medical\_clean4, aes(x=Complication\_risk, y=Initial\_days)) + geom\_boxplot()

ggplot(medical\_clean4, aes(x=Arthritis, y=Initial\_days)) + geom\_boxplot()

ggplot(medical\_clean4, aes(x=Anxiety, y=Initial\_days)) + geom\_boxplot()

ggplot(medical\_clean4, aes(x=Allergic\_rhinitis, y=Initial\_days)) + geom\_boxplot()

ggplot(medical\_clean4, aes(x=HighBlood, y=Initial\_days)) + geom\_boxplot()

ggplot(medical\_clean4, aes(x=Full\_meals\_eaten, y=Initial\_days)) + geom\_point()

## Provide a copy of the cleaned data set.

A copy of the final, cleaned dataset will be attached. This was done following completion of the above cleaning/exploration steps, and utilized the following code:

#writing cleaned/pre-processed medical\_clean4 to folder

write.csv(medical\_clean4, 'C:\\Users\\lgben\\OneDrive\\Desktop\\MSDA\\D209 - Data Mining I\\Task 2 - Predictive Analysis\\medical\_clean4.csv')

# Part IV: Analysis

# Perform the data analysis and report on the results by doing the following:

## Split the data into training and test data sets and provide the file(s)

The next step that I performed was to begin the actual modeling portion of the analysis. Prior to creating the model, I set the seed for reproducibility. Training and testing datasets were then created from my cleaned dataset, medical\_clean4. These were named medical\_clean\_train and medical\_clean\_test, respectively. To verify that the split was performed with the correct number of rows, a variable, n\_80\_20\_split, was created for comparison. These new datasets were then written to a folder on my desktop. All of this was accomplished with the following code:

#set seed for future needs

set.seed(42)

#train-test-split

sample\_rows <- sample(nrow(medical\_clean4), nrow(medical\_clean4)\*0.8)

n\_80\_20\_split <- round(nrow(medical\_clean4) \* 0.8, 0)

medical\_clean\_train <- medical\_clean4[sample\_rows, ]

medical\_clean\_test <- medical\_clean4[-sample\_rows, ]

#writing split data sets to folder

write.csv(medical\_clean\_train, 'C:\\Users\\lgben\\OneDrive\\Desktop\\MSDA\\D209 - Data Mining I\\Task 2 - Predictive Analysis\\medical\_clean\_train.csv')

write.csv(medical\_clean\_test, 'C:\\Users\\lgben\\OneDrive\\Desktop\\MSDA\\D209 - Data Mining I\\Task 2 - Predictive Analysis\\medical\_clean\_test.csv')

## Describe the analysis technique you used to appropriately analyze the data. Include screenshots of the intermediate calculations you performed.

The modeling technique that I used to analyze my data is a random forest model. This model creates multiple decision trees, using bootstrap aggregation and random feature splits from the data, to create an aggregate model (Donges, 2022). This aggregate is used to create predictions based on majority rule and to minimize the out of sample error, or root mean square error (RMSE). RMSE is a measure of the average error, or difference, of the predicted values from their actual values. This is measured in the same units as the actual values. In this case, it would be the average days that the model is off on its prediction of Initial\_days spent in the hospital.

Intermediate steps performed during this part of the analysis include creating and modifying a tuningGrid, which contains hyperparameters for the random forest model (DataCamp, n.d.). The hyperparameters were modified to allow training with different values within the model, with the best model being selected afterwards. The hyperparameters I decided to tune for this model included mtry, splitrule, and minimum node size. Mtry is the number of different variables randomly sampled at each split, splitrule is determines how a branch is split, and minimum node size is how many observations must be included at minimum in each final leaf.

Screenshots of the code for the intermediate steps, implementing the tuning grid, will be included here.

Graphical user interface, text

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

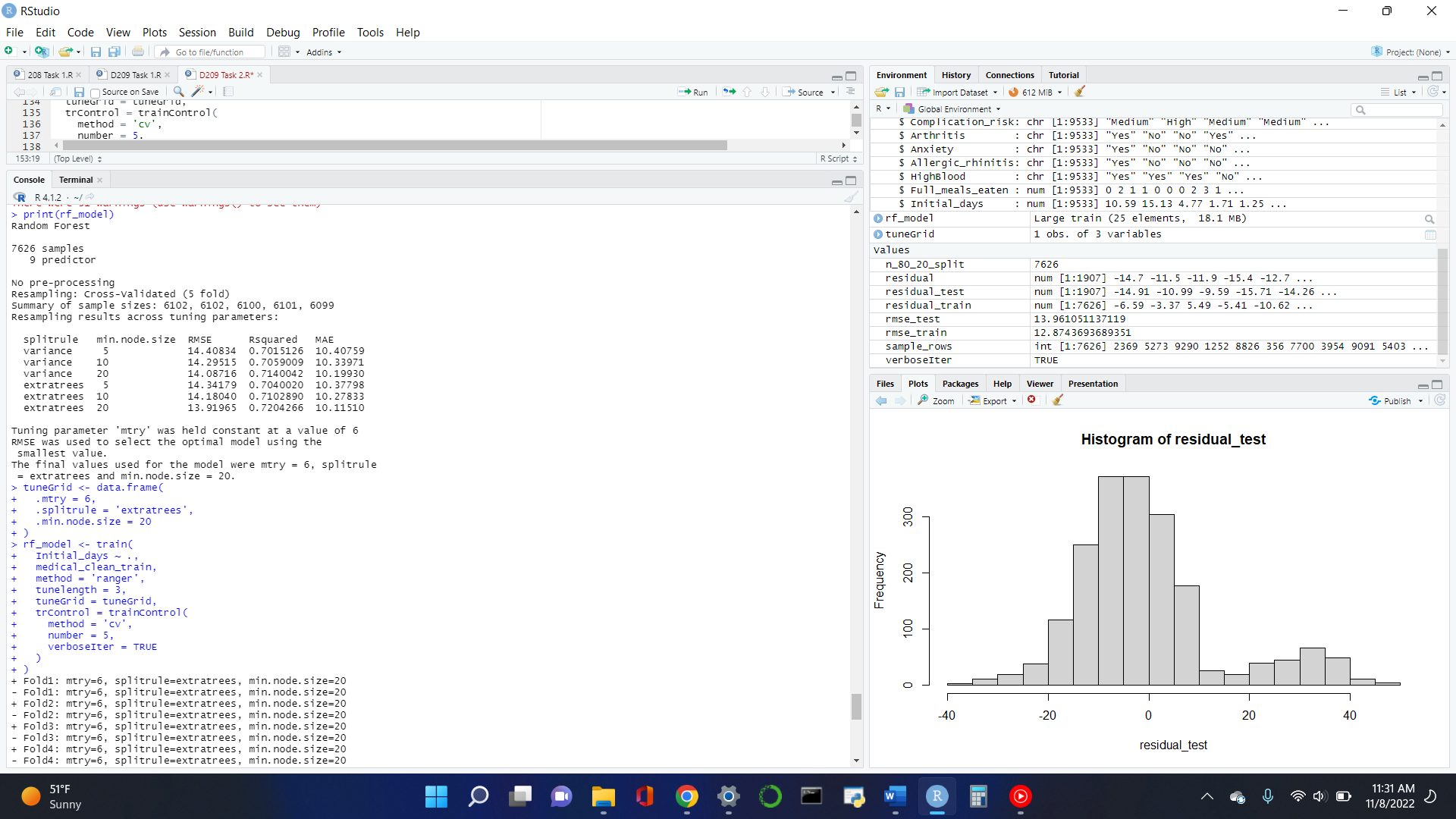
The output from the parameter tuning portion can be seen here:

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

The output of the model made with the final parameters in the tuningGrid is below: 

## Provide the code used to perform the prediction analysis from part D2.

The code that I utilized to perform the analysis is as follows:

#modeling

rf\_model <- train(

Initial\_days ~ .,

medical\_clean\_train,

method = 'ranger',

trControl = trainControl(

method = 'cv',

number = 5,

verboseIter = TRUE))

plot(rf\_model)

#hyperparameter tuning

#introducing tuning grid and retrain

tuneGrid <- expand.grid(

.mtry = 6,

.splitrule = c('variance', 'extratrees'),

.min.node.size = c(5, 10, 20))

rf\_model <- train(

Initial\_days ~ .,

medical\_clean\_train,

method = 'ranger',

tunelength = 3,

tuneGrid = tuneGrid,

trControl = trainControl(

method = 'cv',

number = 5,

verboseIter = TRUE))

print(rf\_model)

plot(rf\_model)

#modify tuning grid to optimal RMSE parameters and retrain

tuneGrid <- data.frame(

.mtry = 6,

.splitrule = 'extratrees',

.min.node.size = 20)

rf\_model <- train(

Initial\_days ~ .,

medical\_clean\_train,

method = 'ranger',

#tunelength removed due to error messages

tuneGrid = tuneGrid,

trControl = trainControl(

method = 'cv',

number = 5,

verboseIter = TRUE))

# Part V: Data Summary and Implications

# Summarize your data analysis by doing the following:

## Explain the accuracy and the mean squared error (MSE) of your prediction model.

As discussed briefly above, the accuracy metric of interest during this modeling process is root mean squared error (RMSE). This is a measure of the average difference between predicted values and actual values. It is measured in the same units as the predicted variable, in this case initial days spent in the hospital. It is calculated by taking the square root of the average squared difference, or sqrt(mean((actual-prediction)^2).

For this specific model, the best model was noted to have an mtry of 6, splitrule of ‘extratrees,’ and minimum node size of 20. The RMSE during the model fitting was noted to be 13.92. This means that on average, the model was off on its predictions by almost 14 days.

RMSE was calculated manually for both the training and test data sets. For the training set, the calculated RMSE was slightly better than what model stated, with rmse\_train being 12.87. The calculated RMSE of the testing data set was noted to be 13.96. This demonstrates that the model performed slightly better on the training dataset, which would be expected as the test data was data the model had not seen before.

## Discuss the results and implications of your prediction analysis

The implication of this model is that we can now begin to predict how long a patient may stay in the hospital based on specific patient factors that were included within the model. If we were to input a new dataframe containing a patient’s, or multiple patients’, specific values for the factors in the model, the model would predict the number of days that each patient would stay. This output could then be utilized to help predict staffing needs, or room requirements, at the hospital based on expected length of stay.

## Discuss **one** limitation of your data analysis

One limitation of the analysis that I performed is that I did not include all potential explanatory variables. Because a reduced linear model had already been created in a prior course, I decided to use only the significant variables from that model. It is very likely that this may have led to a slightly different random forest model due the possible different decision splits that could have been taken with other explanatory variables.

## Recommend a course of action for the real-world organizational situation from part A1 based on your results and implications discussed in part E2.

To recap, the question that I asked above, is “can we predict how many days a patient may stay at the hospital based on comorbidities/other patient factors?” Based on the output of the model, I would say that we can predict a patient’s length of stay.

I would recommend utilizing this model daily or weekly, to estimate patient length of stays and room utilization. This can be used to estimate staffing needs based on estimated patient volumes/total length of stays. I would further recommend continued data gathering and recreating the model in the future utilizing new data. This could be completed every 6 to 12 months if desired. It may be noted that the model may change in the future based on different patient factors/comorbidities, and therefore change its estimates.

# Part VI: Demonstration

# Provide a Panopto video recording that includes a demonstration of the functionality of the code used for the analysis and a summary of the programming environment.

## *Note: The audiovisual recording should feature you visibly presenting the material (i.e., not in voiceover or embedded video) and should simultaneously capture both you and your multimedia presentation*

# Record the web sources used to acquire data or segments of third-party code to support the analysis. Ensure the web sources are reliable.

1. DataCamp. (n.d.). *CreateGrid: Tuning Parameter Grid*. RDocumentation. Retrieved November 8, 2022, from <https://www.rdocumentation.org/packages/caret/versions/4.47/topics/createGrid>
2. Donges, N. (2022, September 28). *Random forest classifier: A complete guide to how it works in Machine Learning*. Built In. Retrieved November 8, 2022, from <https://builtin.com/data-science/random-forest-algorithm#procon>
3. Yiu, T. (2019, June 12). *Understanding random forest - towardsdatascience.com*. Towards Data Science. Retrieved November 8, 2022, from <https://towardsdatascience.com/understanding-random-forest-58381e0602d2>

# Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.

# Demonstrate professional communication in the content and presentation of your submission.