Data Cleaning Assignment

Mercedes Joseph

Professor Middleton

Data cleaning -D206

June 12, 2023

Part I: Research Question

A.

What demographic and medical factors have a higher likelihood of impacting a patient's readmission?

B.

B1.

* CaseOrder: A variable used to maintain the original order of the raw data file.
* Customer\_id: A unique identifier assigned to each patient.
* Interaction, UID: Unique identification numbers associated with patient transactions, procedures, and admissions.
* City: The city where the patient resides, as listed on the billing statement.
* State: The state where the patient resides, as listed on the billing statement.
* County: The county where the patient resides, as listed on the billing statement.
* Zip: The zip code of the patient's residence, as listed on the billing statement.
* Lat, Lng: GPS coordinates representing the latitude and longitude of the patient's residence, as listed on the billing statement.
* Population: The population within a one-mile radius of the patient's residence, based on census data.
* Area: The type of area where the patient resides (rural, urban, suburban), based on unofficial census data.
* TimeZone: The time zone of the patient's residence, obtained from the patient's sign-up information.
* Job: The occupation of the patient or primary insurance holder, as reported in the admissions information.
* Children: The number of children in the patient's household, as reported in the admissions information.
* Age: The age of the patient, as reported in the admissions information.
* Education: The highest degree earned by the patient, as reported in the admissions information.
* Employment: The employment status of the patient, as reported in the admissions information.
* Income: The annual income of the patient or primary insurance holder, as reported at the time of admission.
* - Marital: The marital status of the patient or primary insurance holder, as reported on the admission information.
* Gender: The self-identified gender of the customer (male, female, nonbinary).
* ReAdmis: Indicates whether the patient was readmitted within a month of their release.
* VitD\_levels: The patient's vitamin D levels measured in ng/mL.
* Doc\_visits: The number of times the primary physician visited the patient during the initial hospitalization.
* Full\_meals\_eaten: The number of full meals the patient consumed while hospitalized.
* VitD\_supp: The number of times vitamin D supplements were administered to the patient.
* Soft\_drink: Indicates whether the patient habitually consumes three or more sodas per day.
* Initial\_admin: The method by which the patient was initially admitted to the hospital (emergency admission, elective admission, observation).
* HighBlood: Indicates whether the patient has high blood pressure.
* Stroke: Indicates whether the patient has had a stroke.
* Complication risk: The level of complication risk for the patient as assessed by a primary patient assessment (high, medium, low).
* Overweight: Indicates whether the patient is considered overweight based on age, gender, and height.
* Arthritis: Indicates whether the patient has arthritis.
* Diabetes: Indicates whether the patient has diabetes.
* Hyperlipidemia: Indicates whether the patient has hyperlipidemia.
* BackPain: Indicates whether the patient has chronic back pain.
* Anxiety: Indicates whether the patient has an anxiety disorder.
* Allergic\_rhinitis: Indicates whether the patient has allergic rhinitis.
* Reflux\_esophagitis: Indicates whether the patient has reflux esophagitis.
* Asthma: Indicates whether the patient has asthma.
* Services: The primary service provided by the hospital.
* Initial\_days: The number of days the patient stayed in the hospital during the initial visit.
* TotalCharge: The average amount charged to the patient per day, representing the total charges billed to patients during their hospital stay.
* Additional\_charges: The average amount charged to the patient for miscellaneous procedures, treatments, medicines, anesthesiology, and other services not included in the regular charges.
* Item1: Rating of the importance of timely admission.
* Item2: Rating of the importance of timely treatment.
* Item3: Rating of the importance of timely visits.
* Item4: Rating of the importance of reliability.
* Item5: Rating of the importance of available options.
* Item6: Rating of the importance of hours of treatment.
* Item7: Rating of the importance of having a courteous staff.
* Item8: Rating of the importance of active listening from the doctor.

|  |  |
| --- | --- |
| *These are the variables in RStudio's this shows the variable name, data type, and examples of each variable.*   |  | | --- | | $ X *<int>* 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, …  $ CaseOrder *<int>* 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, …  $ Customer\_id *<chr>* "C412403", "Z919181", "F995323", "A879973", …  $ Interaction *<chr>* "8cd49b13-f45a-4b47-a2bd-173ffa932c2f", "d2…  $ UID *<chr>* "3a83ddb66e2ae73798bdf1d705dc0932", "176354…  $ City *<chr>* "Eva", "Marianna", "Sioux Falls", "New Rich…  $ State *<chr>* "AL", "FL", "SD", "MN", "VA", "OK", "OH", "…  $ County *<chr>* "Morgan", "Jackson", "Minnehaha", "Waseca", …  $ Zip *<int>* 35621, 32446, 57110, 56072, 23181, 74423, 4…  $ Lat *<dbl>* 34.34960, 30.84513, 43.54321, 43.89744, 37.…  $ Lng *<dbl>* -86.72508, -85.22907, -96.63772, -93.51479, …  $ Population *<int>* 2951, 11303, 17125, 2162, 5287, 981, 2558, …  $ Area *<chr>* "Suburban", "Urban", "Suburban", "Suburban"…  $ Timezone *<chr>* "America/Chicago", "America/Chicago", "Amer…  $ Job *<chr>* "Psychologist, sport and exercise", "Commun…  $ Children *<int>* 1, 3, 3, 0, NA, NA, 0, 7, NA, 2, 4, NA, 0, …  $ Age *<int>* 53, 51, 53, 78, 22, 76, 50, 40, 48, 78, 55, …  $ Education *<chr>* "Some College, less than 1 Year", "Some Col…  $ Employment *<chr>* "Full Time", "Full Time", "Retired", "Retir…  $ Income *<dbl>* 86575.93, 46805.99, 14370.14, 39741.49, 120…  $ Marital *<chr>* "Divorced", "Married", "Widowed", "Married"…  $ Gender *<chr>* "Male", "Female", "Female", "Male", "Female…  $ ReAdmis *<chr>* "No", "No", "No", "No", "No", "No", "No", "…  $ VitD\_levels *<dbl>* 17.80233, 18.99464, 17.41589, 17.42008, 16.…  $ Doc\_visits *<int>* 6, 4, 4, 4, 5, 6, 6, 7, 6, 7, 6, 7, 5, 5, 4…  $ Full\_meals\_eaten *<int>* 0, 2, 1, 1, 0, 0, 0, 2, 3, 1, 3, 3, 3, 0, 2…  $ VitD\_supp *<int>* 0, 1, 0, 0, 2, 0, 0, 0, 0, 2, 0, 0, 1, 0, 1…  $ Soft\_drink *<chr>* NA, "No", "No", "No", "Yes", "No", NA, NA, …  $ Initial\_admin *<chr>* "Emergency Admission", "Emergency Admission…  $ HighBlood *<chr>* "Yes", "Yes", "Yes", "No", "No", "No", "Yes…  $ Stroke *<chr>* "No", "No", "No", "Yes", "No", "No", "No", …  $ Complication\_risk *<chr>* "Medium", "High", "Medium", "Medium", "Low"…  $ Overweight *<int>* 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0…  $ Arthritis *<chr>* "Yes", "No", "No", "Yes", "No", "Yes", "Yes…  $ Diabetes *<chr>* "Yes", "No", "Yes", "No", "No", "Yes", "Yes…  $ Hyperlipidemia *<chr>* "No", "No", "No", "No", "Yes", "No", "Yes",…  $ BackPain *<chr>* "Yes", "No", "No", "No", "No", "Yes", "Yes"…  $ Anxiety *<int>* 1, NA, NA, NA, 0, 0, 1, 0, NA, 0, 0, 0, 1, …  $ Allergic\_rhinitis *<chr>* "Yes", "No", "No", "No", "Yes", "Yes", "No"…  $ Reflux\_esophagitis *<chr>* "No", "Yes", "No", "Yes", "No", "No", "Yes"…  $ Asthma *<chr>* "Yes", "No", "No", "Yes", "No", "No", "No",…  $ Services *<chr>* "Blood Work", "Intravenous", "Blood Work", …  $ Initial\_days *<dbl>* 10.585770, 15.129562, 4.772177, 1.714879, 1…  $ TotalCharge *<dbl>* 3191.049, 4214.905, 2177.587, 2465.119, 188…  $ Additional\_charges *<dbl>* 17939.403, 17612.998, 17505.192, 12993.437, …  $ Item1 *<int>* 3, 3, 2, 3, 2, 4, 4, 1, 3, 5, 3, 4, 4, 3, 4…  $ Item2 *<int>* 3, 4, 4, 5, 1, 5, 3, 2, 3, 5, 3, 5, 4, 3, 5…  $ Item3 *<int>* 2, 3, 4, 5, 3, 4, 3, 2, 2, 5, 4, 5, 2, 3, 5…  $ Item4 *<int>* 2, 4, 4, 3, 3, 4, 2, 5, 3, 3, 4, 5, 4, 5, 4…  $ Item5 *<int>* 4, 4, 3, 4, 5, 3, 3, 4, 3, 4, 2, 3, 5, 4, 5…  $ Item6 *<int>* 3, 4, 4, 5, 3, 5, 4, 2, 3, 2, 5, 3, 2, 5, 3…  $ Item7 *<int>* 3, 3, 3, 5, 4, 4, 5, 4, 4, 3, 2, 3, 4, 3, 3…  $ Item8 *<int>* 4, 3, 3, 5, 3, 6, 5, 2, 2, 2, 3, 3, 3, 2, 5… |   Part II: Data Cleaning   1. Propose a plan that includes the relevant techniques and specific steps needed to assess the quality of the data in the data set.   C1. The plan to clean the data set:  I will first upload all packages and their libraries to help assist with the data cleaning process.   * + install. Packages("tidyverse") & library(tidyverse) – Tidyverse provides a collection of packages for data manipulation, visualization, and analysis, offering powerful tools for working with data.   + install.packages("stats") & library(stats) - `stats`: stats are a core package in R that offers various statistical functions and distributions for basic statistical analysis.   + install.packages("corrplot") & library(corrplot) - Corrplot package enables the visualization of correlation matrices, helping to explore relationships between variables in a dataset.   + install.packages("FactoMineR") & library (FactoMineR) - FactoMineR facilitates multivariate exploratory data analysis and dimensionality reduction techniques, such as PCA and correspondence analysis.   + install.packages("factoextra") & library(factoextra) - Factoextra package complements `FactoMineR` by providing functions for extracting and visualizing information from multivariate analysis results.   + Library(ggplot2) & (tidyr)- I used ggplot2 and Tidyr to help visualize and analyze the data. With ggplot2, I created different types of plots, like scatter plots and bar plots, to understand patterns and identify outliers in the data. Tidyr helped me organize and transform the data, making it easier to create meaningful visualizations. By using these libraries together, I gained insights into the data, which helped me make decisions based on the observed trends and anomalies.   I plan on taking a 6-step data cleaning plan first check for duplicate rows, irrelevant column names, and any misleading information that exists in the raw data frame. My second task is handling missing values in the data set with either the mean, median, or mode of each column. The third step is standardizing the variables I plan on turning most Char values into numeric just to make the data easier to interpret and clean. The fourth step is after transforming the Char values into numeric creating a new data frame with all the cleaned data. In my fifth step I plan on running a series of statistical modeling like boxplots, histograms, etc. to determine outliers for all quantitative data adjusting the outliers accordingly. In my sixth step I will perform PCA to find which Dimensions have more influence on the data set.  C2.  I used these methods to clean the dataset to ensure its quality and reliability for analysis. The `tidyverse` package, developed by Wickham et al. (2019), provided efficient tools for organizing, visualizing, and analyzing the data. I relied on the `stats` package for basic statistical analysis, such as calculating descriptive statistics and conducting hypothesis tests. The `corrplot` package helped me visualize relationships between variables, while the `FactoMineR` package allowed for dimensionality reduction and identifying influential variables. The `factoextra` package assisted `FactoMineR` by providing additional analysis and visualization functions. To handle missing data, I utilized the `mice` package for effective imputation. This comprehensive data cleaning approach ensures accurate results for further analysis. In terms of my six-step data cleaning plan, it aimed to address common issues in the dataset. I began by checking for duplicate rows, removing irrelevant column names, and scrutinizing misleading information. Then, I handled missing values by employing appropriate imputation methods, standardized variables to maintain consistency, addressed outliers through statistical modeling techniques, and performed PCA to identify influential dimensions.  C3. Justify your selected programming language and any libraries and packages that will support the data-cleaning process.  To ensure the success of this project, I will be using the coding language R. R is specifically designed for statistical computing and has many advantages for data analysis and statistical modeling. It offers built-in functions that are helpful for data manipulation, visualization, and statistical analysis. R's easy to understand, making it accessible to users with various levels of programming experience, explore and visualize my data throughout the cleaning process. R studios environment “provides a good environment for reproducible data cleaning” (De Jonge, E., & Van Der Loo, M., 2013). To assist with cleaning the data, I will be using the Tidyverse, Stats, Corrplot, FactoMineR, and Factoextra packages in R.  The Tidyverse package provides tools for data manipulation and follows clean data principles. The Stats package offers various statistical functions, while Corrplot helps visualize relationships between variables. FactoMineR is essential for exploring complex data, and Factoextra complements it by providing additional analysis and visualization capabilities. Using these packages, I can effectively clean and analyze the data, ensuring it is ready for further analysis and interpretation. Overall, these data-cleaning steps resulted in a cleaner dataset with no duplicates, missing values, or irrelevant information. The library ggplot2 was also used to create boxplots and bar graphs.  C4.  install.packages("tidyverse")  library(tidyverse)  install.packages("stats")  library(stats)  install.packages("corrplot")  library(corrplot)  install.packages("FactoMineR")  library(FactoMineR)  install.packages("factoextra")  library(factoextra)  library(ggplot2)  library(ggplot2)  library(tidyr)  #upload packages and libraries  MD <- read\_csv("C:/Users/merce/Downloads/medical\_raw\_data.csv")  #uploading raw data  View(MD)  #View data frame  str(MD)  #Viewing data types and their examples  duplicates <- duplicated(MD)  #Checking data frame for duplicates  print(MD[duplicates, ])  #Print duplicate row 0 found  MD <- MD[, -1]  #Delete the first column due to it being repetitive  MD <- MD %>%  mutate(index = CaseOrder) %>%  select(-CaseOrder)  #Setting Index  missing\_counts <- colSums(is.na(MD))  #Checking the missing values of each column  print(missing\_counts)  #Show the sum of missing value  colnames(MD)[colnames(MD) == "Item1"] <- "Timely admission"  colnames(MD)[colnames(MD) == "Item2"] <- "Timely treatment"  colnames(MD)[colnames(MD) == "Item3"] <- "Timely visits"  colnames(MD)[colnames(MD) == "Item4"] <- "Reliability"  colnames(MD)[colnames(MD) == "Item5"] <- "Options"  colnames(MD)[colnames(MD) == "Item6"] <- "Hours of treatment"  colnames(MD)[colnames(MD) == "Item7"] <- "Courteous staff"  colnames(MD)[colnames(MD) == "Item8"] <- "Evidence of active listening from doctor"  #Change Item 1-8 names to relative descriptions  colnames(MD)  #View all column names  convert\_to\_numeric <- function(x) {  ifelse(x == "Yes", 1, 0)  }  #Convert columns that use variables Yes and No to numeric  MD[, c("HighBlood", "Stroke", "Complication\_risk", "Arthritis", "Diabetes",  "Hyperlipidemia", "BackPain", "Allergic\_rhinitis", "Reflux\_esophagitis", "Asthma", "ReAdmis", "Soft\_drink")] <- lapply(MD[, c("HighBlood", "Stroke", "Complication\_risk", "Arthritis", "Diabetes", "Hyperlipidemia”, "BackPain", "Allergic\_rhinitis", "Reflux\_esophagitis", "Asthma", "ReAdmis", "Soft\_drink")], convert\_to\_numeric)  #Converting Yes/NO to numeric  MD$Zip <- as.character(MD$Zip)  #Convert zip codes to character type  MD$Zip <- str\_pad(MD$Zip, width = 5, pad = "0")  #Add leading zeros to zip code  ggplot(data = MD, aes(x = Lng, y = Lat)) +  geom\_point() +  labs(x = "Longitude", y = "Latitude") +  theme\_bw()  # lat and lng  children\_median <- median(MD$Children, na.rm = TRUE)  #Replace null values for median  MD$Children[is.na(MD$Children)] <- children\_median  # Replace missing values with the mean  median\_income <- median(MD$Income, na.rm = TRUE)  # Calculate the mean of the non-missing values  MD$Income[is.na(MD$Income)] <- median\_income  # Replace missing values with the mean  mean\_Age <- mean(MD$Age, na.rm = TRUE)  # Calculate the mean of the non-missing values  MD$Age[is.na(MD$Age)] <- mean\_Age  #Replace missing values with the mean  mean\_Initial\_days <- mean(MD$Initial\_days, na.rm = TRUE)  # Calculate the mean of the non-missing values  MD$Initial\_days[is.na(MD$Initial\_days)] <- mean\_Initial\_days  #Replace missing values with the mean  missing\_sum <- colSums(is.na(MD))  # Calculate the sum of missing values  print(missing\_sum)  # Print the sum of missing values  # Print the sum of missing values  MD$Age <- round(MD$Age)  #Round the variable age  print(MD$Age)  #Print the new rounded age column  Mode <- function(x) {  ux <- unique(x)  ux[which.max(tabulate(match(x, ux)))]  }  #Function to calculate mode  mode\_overweight <- Mode(MD$Overweight)  #calculate mode for overweight column  MD$Overweight[is.na(MD$Overweight)] <- mode\_overweight  #Replace null values with mode  mode\_Anxiety <- Mode(MD$Anxiety)  #calculate mode for anxiety  MD$Anxiety[is.na(MD$Anxiety)] <- Mode(MD$Anxiety)  #Impute mode in anxiety column  mode\_Soft\_drink <- Mode(MD$Soft\_drink)  #Calculate mode for soft drink  MD$Soft\_drink[is.na(MD$Soft\_drink)] <- Mode(MD$Soft\_drink)  #Fill in NA value with mode  View(MD)  #View MD  variables\_of\_interest <- c("Lat", "Lng", "Population", "Children", "Age", "Income",  "ReAdmis", "VitD\_levels", "Doc\_visits", "Full\_meals\_eaten",  "VitD\_supp", "Soft\_drink", "HighBlood", "Stroke",  "Complication\_risk", "Overweight", "Arthritis", "Diabetes",  "Hyperlipidemia", "BackPain", "Anxiety", "Allergic\_rhinitis",  "Reflux\_esophagitis", "Asthma", "Initial\_days", "TotalCharge",  "Additional\_charges", "Timely admission", "Timely treatment",  "Timely visits", "Reliability", "Options", "Hours of treatment")  #Create new dataframe  df <- MD[, variables\_of\_interest]  #Create new data frame  non\_numeric\_cols <- sapply(df, function(x) !is.numeric(x))  #Identify the columns in df that are not numeric  df[!non\_numeric\_cols] <- lapply(df[!non\_numeric\_cols], as.numeric)  #Convert the non nuemric column excluding char or factors  detect\_outliers <- function(x) {  q1 <- quantile(x, 0.25, na.rm = TRUE)  q3 <- quantile(x, 0.75, na.rm = TRUE)  iqr <- q3 - q1  lower\_fence <- q1 - 1.5 \* iqr  upper\_fence <- q3 + 1.5 \* iqr  outliers <- x[x < lower\_fence | x > upper\_fence]  return(outliers)  }  # Detect outliers in each column  outliers\_list <- lapply(df, detect\_outliers)  # Identify columns with outliers  columns\_with\_outliers <- names(df)[sapply(outliers\_list, length) > 0]  #Identify columns with outliers  print(columns\_with\_outliers)  #print column outliers  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Lat)) +  labs(x = "", y = "Lat") +  theme\_bw() +  ggtitle("Box Plot of Lat")  #Boxplot Lat  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Lng)) +  labs(x = "", y = "Lng") +  theme\_bw() +  ggtitle("Box Plot of Lng")  #Boxplot for LNG  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Population)) +  labs(x = "", y = "Population") +  theme\_bw() +  ggtitle("Box Plot of Population")  #Boxplot for Population  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Income)) +  labs(x = "", y = "Income") +  theme\_bw() +  ggtitle("Box Plot of Income")  #Boxplot for Income  boxplot\_TotalCharge <- boxplot(df$TotalCharge)  #boxplot for total charge  boxplot\_Additional\_charges <- boxplot(df$Additional\_charges)  #Additional Charge boxplot  boxplot\_Timely\_admission <- boxplot(df$`Timely admission`)  #boxplot for TA  boxplot\_Timely\_treatment <- boxplot(df$`Timely treatment`)  #Boxplot TT  boxplot\_Timely\_visits <- boxplot(df$`Timely visits`)  #Boxplot TV  boxplot\_Reliability <- boxplot(df$Reliability)  #boxplot Reliabilty  boxplot\_Options <- boxplot(df$Options)  #Boxplot options  boxplot\_Hours\_of\_treatment <- boxplot(df$`Hours of treatment`)  #boxplot Hours of Treatment  percentage\_outliers <- length(columns\_with\_outliers) / nrow(MD) \* 100  # Find the percentage of outliers  percentage\_remaining <- 100 - percentage\_outliers  #Calculate the percentage remaining  cat("Percentage of outliers:", percentage\_outliers, "%\n")  cat("Percentage of data remaining:", percentage\_remaining, "%\n")  #print results  Unclean <- read\_csv("C:/Users/merce/Downloads/medical\_raw\_data.csv")  #Unclean data  columns <- c("Children", "Soft\_drink", "Anxiety", "Income", "Overweight", "Initial\_days", "Age")  # columns for unclean histogram  Unclean[columns] <- lapply(Unclean[columns], function(x) as.numeric(x, na.rm = TRUE))  #Turn char values to numeric  par(mfrow = c(2, 4))  #set up layout  for (col in columns) {  values <- Unclean[[col]]  values <- values[!is.na(values)]  # Filter out missing values  if (!is.null(values) && length(values) > 0)  hist(values, main = col, xlab = col, col = "lightblue")  }  #Create histogram for unclean  columns <- c("Children", "Soft\_drink", "Anxiety", "Income", "Overweight", "Initial\_days", "Age")  #Select column from new\_MD  par(mfrow = c(2, 4))  #set the layout of the subplots  for (col in columns) {  hist(MD[[col]], main = col, xlab = col, col = "lightblue")  }  No matter how I copy and paste the cod still show up with a formatting issue. I have attached a r script that will let you view the data quality assessment code in its entirety. File named Data quality Assessment.  #Histogram for cleaned data MD Part III: Data Cleaning  D.  D1.  The first step I took was checking to see if I had any duplicate columns in my data frame with the following code:    The code gave back the following response of 0 duplicates being found.    I then checked the code for irrelevant columns, finding the first column being unnamed just being a repeat of the column named Case order. I then deleted the first column and made Case Order the Index for the data set using the following code.    Next to determine what values were missing I used the missing count function to count the missing values in column. This is the code I used:    The code gave me the following:  Children 2588  Age 2414  Income 2464  Soft\_drink 2467  Overweight 982  Anxiety 984  Initial days 1056  I then changed the Item 1- 8 names to the respective description:    I wanted to make sure that the Zip codes format was uniform and added back any leading zeros so that the number of digits is all the same.    I changed all the categorical data to numerical with the following code: Yes= 1 and No =0      The first section of the code to turn categorical data to numeric  The second section of the code to change categorical data to numerical.  To fill in missing values I either used the median, mode, or mean based on the type of data.  I used Mode to fill missing values of categorical data.    I filled the missing values of Normal data with the column’s median    For skewed data I used the mean to fill in missing data in columns. I also rounded the age so that there were no data quality issues.    The code below allowed me to create a map using ggplot2:      The map of Lng and Lat has detected outliers, but most have landed on the United States map. I will not be adjusting the outliers due to the Lng and Lat reflects the person residence which is self-reported.  Next Detecting outliers  I ran a code to detect which variables had outliers.    This was the code used ^    The code listed the names of the columns that had outliers detected.  [1] "Lat" "Lng" "Population"   [4] "Children" "Income" "VitD\_levels"   [7] "Full\_meals\_eaten" "VitD\_supp" "Soft\_drink"  [10] "Stroke" "TotalCharge" "Additional\_charges" [13] "Timely admission" "Timely treatment" "Timely visits"  [16] "Reliability" "Options" "Hours of treatment"  These were the columns that were detected to have outliers. I then created boxplots to represent each quantitative column to get a better view of the outliers. I did not create the boxplot for binary values such as Stroke, vitamin D supplements, and full meals eaten.    This was the code that create the boxplot for population.  A picture containing text, screenshot, diagram, number  Description automatically generated  Create a box plot in population. There is outlier but the outliers ae based on information that cannot be altered and does not influence the data in any way.  A picture containing text, font, screenshot, line  Description automatically generated  Code that created the boxplot for Lat    The boxplot has outliers but most of them fall on the map that was shown above. I will not be deleting the outlier since most fall in range and the patients self-reported their locations.    Code that created the boxplot for Lng  A picture containing text, diagram, line, screenshot  Description automatically generated  The boxplot has outliers but most of them fall on the map that was shown above. I will not be deleting the outlier since most fall in range and the patients self-reported their locations.  A screenshot of a computer  Description automatically generated with medium confidence  This code created the box plot for the population.    The population boxplot shows outliers that will not be changed due to this being a fact that was included in the patients file it does not need to be fixed.  A picture containing text, font, screenshot, line  Description automatically generated  This code creates the box plot Income.  A picture containing text, diagram, line, screenshot  Description automatically generated  The boxplot shows outliers, but these outliers seem legitimate and not an error due to the patient’s elf reporting this information.    This code created the box plot for Total charge.    The boxplot for Total Charge shows a great number of outliers but I do not want to extract this data. This is the total amount charged based on the hospital’s records and these amounts can vary based on number of days stayed, treatment received, and initial procedures taken.    Code to create boxplot for Additional Charges Column  A picture containing screenshot, text, rectangle, diagram  Description automatically generated  The boxplot for additional charges does show outlier but again these were values based on the hospital’s records of what the charged the patient. I do not think these outliers are an area just based on the influence of the patient’s care while at the hospital.    Boxplot was created to reflect the values of Timely admission survey answers.  A picture containing diagram, rectangle, technical drawing, sketch  Description automatically generated  This boxplot does display outlier, but they are self-reported responses therefore they are not errors, most likely a difference of opinion when it comes to a patient’s admission experience.    Boxplot created to reflect the survey answers of Timely Treatment    Timely treatment has a minimal number of outliers and are also patient reported no errors so these outlier values will not be extracted.    Boxplot created for Timely Visits column.    The box plot shows patient survey answer that I will not be changing due to this not being an error but a reflection of the patient’s experience. The outliers are of minimal value and do not greatly affecting the data’s quality.    Code that created boxplot for reliability.  A picture containing rectangle, diagram, screenshot, line  Description automatically generated  The boxplot shows outliers on its minimum and its maximum. These are based on the survey from patients and are not errors just difference of opinion they will not be extracted.    Boxplot for Options was created from the code above.  A picture containing diagram, rectangle, screenshot, text  Description automatically generated  The box plot displays outliers that are patient reported that can be caused by a difference of opinion and not error no change will be made.    Code to create hours of treatment box plot.    This boxplot shows outliers that are due to patient’s responses to the survey and do not significantly change the data set so the outliers will not be extracted.  A screenshot of a computer  Description automatically generated with medium confidence  The code for the histogram to see if the data skewed after I filled in the missing variables.  A screenshot of a graph  Description automatically generated with low confidence  The data did not skew and still looks like the original data set with just a growth in the middle of Initial days, Age, and Income. The imputation was a success the above photo is how the data represents itself in a histogram following imputation.    Code that created unclean histogram.  This is the original histogram of the unclean data.    No major changes in the data even after imputation.  I decided not to delete the outliers because they reflect patient’s experience. As well as when I did the calculation that I will show down below the outliers made up less than .20% of the data. Meaning it would not significantly change any of the data quality. It is such a small percentage and important for patient demographics treating the outliers will make no significant difference.    The code used    The results percentage of outlier 0.18% and data remaining after outlier extraction 99.82%  D2.  I used a method to check for duplicates in both rows and columns of the dataset. By utilizing the `duplicated () ` function, I found that there were no duplicate columns in the data. However, I did notice a repetitive column named "Unnamed" that duplicated the "Case Order" column, so I removed it. To maintain the organization of the data, I made "Case Order" the index by using the `mutate () ` function and renaming the column as "index". To identify missing values, I applied the `colSums()’ function, which helped me determine the number of null values in each column. This allowed me to identify the columns with missing data. I then renamed the last eight columns using the `colnames()` function to provide more appropriate and descriptive names.    To ensure consistent formatting, I ensured that all ZIP codes had leading zeros. This step ensured that the formatting of the variable remained consistent across the dataset. Taking the information, I found from Webinar 2 I applied the missing values according to this rule “If normal, mean is acceptable; for skewed data it is suggested to utilize the median; for categorical data use the mode” (Middelton, 2022). To handle missing values in certain columns, such as "overweight," "anxiety," and "soft drink," I calculated the mode using a custom function. This allowed me to fill in the missing values with the mode of each respective column. Similarly, I utilized the median to calculate missing values in the "income" and "children" columns. For the "age" and "initial day" columns, I calculated the mean and filled in the missing values accordingly. These methods proved effective in filling the missing values, and I learned about them from Webinar 2. To re-express categorical data appropriately, I transformed "yes" values to 1 and "no" values to 0. This ensured consistent representation of the categorical variables in a binary format. In order to gain a visual understanding of the distribution of longitude ("Lng") and latitude ("Lat") values, I employed the `ggplot` package to create a visualization on a map. This allowed me to better identify any potential outliers in these variables. By using the `lapply()` function, I detected columns with outliers and printed their names for further investigation. I then created boxplots for each of these columns to assess whether any treatment of the outliers was necessary. Overall, the steps helped in cleaning and preparing the dataset, ensuring data integrity, and addressing missing values and outliers.  D3.  In summary, I went through several steps to clean and prepare the dataset. Firstly, I checked for duplicate rows and columns, ensuring that the data was free of redundancy. Fortunately, there were no duplicate columns, but I did come across a repetitive column named "Unnamed" that duplicated the "Case Order" column.  duplicates <- duplicated(MD)  #Checking data frame for duplicates  print(MD[duplicates, ])    *The results*  MD <- MD[, -1] #Delete the first column due to it being repetitive  MD <- MD %>%  mutate(index = CaseOrder) %>%  Select(-CaseOrder) #Setting Index    *Proof that the first column was dropped and the second turned into an index.*  I promptly removed it to maintain data accuracy. Next, I made "Case Order" the index, which allowed me to preserve the original organization of the data. By using the `mutate () ` function, I successfully designated "Case Order" as the new index column, renaming it as "index. "To address missing values, I employed the `colSums()` function, which helped me identify columns with null values. This enabled me to target specific columns for further attention. I also took the opportunity to improve column names, utilizing the `colnames()` function to provide more descriptive and meaningful names to the last eight columns. Consistency in formatting was a priority, so I ensured that all ZIP codes had leading zeros. This uniformity enhanced the dataset's overall quality and appearance. Handling missing values was crucial, so I applied appropriate functions to fill in the gaps. Utilizing custom functions, I calculated the mode for columns such as "overweight," "anxiety," and "soft drink," and filled missing values accordingly. For "income" and "children" columns, I relied on the median, while the mean helped me address missing values in "age" and "initial day" columns. This approach allowed for a more complete and more reliable dataset. missing\_sum <- colSums(is.na(MD)) # Calculate the sum of missing values print(missing\_sum)  Code for to check for missing values    *Results after code was ran*  Re-expressing categorical data was essential to maintain consistency and improve analysis. I transformed "yes" values to 1 and "no" values to 0, aligning them with a binary representation for accurate interpretation.  To gain visual insights into the geographical distribution of data, I employed the powerful `ggplot` package to create visualizations of "Lng" and "Lat" values. This mapping approach facilitated the identification of potential outliers and improved comprehension of the data.  0  ggplot(data = MD, aes(x = Lng, y = Lat)) + geom\_point() + labs(x = "Longitude", y = "Latitude") + theme\_bw() # lat and lng    *The outliers seen on the map.*  Finally, I used the `lapply()` function to detect columns with outliers and printed their names for further investigation. Creating boxplots for each identified column aided in determining whether the outlier's needed treatment or further attention.  Code to detect outliers:  df <- MD[, variables\_of\_interest]  #Create new data frame  non\_numeric\_cols <- sapply(df, function(x) !is.numeric(x)) #Identify the columns in df that are not numeric df[!non\_numeric\_cols] <- lapply(df[!non\_numeric\_cols], as.numeric) #Convert the non nuemric column excluding char or factors  detect\_outliers <- function(x) { + q1 <- quantile(x, 0.25, na.rm = TRUE) + q3 <- quantile(x, 0.75, na.rm = TRUE) + iqr <- q3 - q1 + lower\_fence <- q1 - 1.5 \* iqr + upper\_fence <- q3 + 1.5 \* iqr + outliers <- x[x < lower\_fence | x > upper\_fence] + return(outliers) + } # Detect outliers in each column outliers\_list <- lapply(df, detect\_outliers) # Identify columns with outliers columns\_with\_outliers <- names(df)[sapply(outliers\_list, length) > 0] #Identify columns with outliers print(columns\_with\_outliers)    *These are the numerical values with outliers that I made boxplots for above.*  Through these steps, I successfully cleaned the dataset, ensuring data integrity, addressing missing values, and identifying potential outliers. The dataset is now well-prepared for subsequent analysis and interpretation. Now that the data has been cleaned, it means that all the missing values have been taken care of, and the column names have been changed to make them easier to understand. This makes it simpler for anyone looking at the data from outside to know what each column represents. The cleaned dataset now provides a complete profile for each customer. This is useful for detecting specific patient details or characteristics. The improvements made to the dataset make it easier to analyze and gain valuable insights, which can support better decision-making. Overall, by cleaning and organizing the data, I have created a dataset that is ready for further analysis and can be easily understood and utilized by researchers, analysts, and stakeholders.  D4. Annotated Code  install.packages("tidyverse")  library(tidyverse)  install.packages("stats")  library(stats)  install.packages("corrplot")  library(corrplot)  install.packages("FactoMineR")  library(FactoMineR)  install.packages("factoextra")  library(factoextra)  library(ggplot2)  library(ggplot2)  library(tidyr)  #upload packages and libraries  MD <- read\_csv("C:/Users/merce/Downloads/medical\_raw\_data.csv")  #uploading raw data  View(MD)  #View data frame  str(MD)  #Viewing data types and their examples  duplicates <- duplicated(MD)  #Checking data frame for duplicates  print(MD[duplicates, ])  #Print duplicate row 0 found  MD <- MD[, -1]  #Delete the first column due to it being repetitive  MD <- MD %>%  mutate(index = CaseOrder) %>%  select(-CaseOrder)  #Setting Index  missing\_counts <- colSums(is.na(MD))  #Checking the missing values of each column  print(missing\_counts)  #Show the sum of missing value  colnames(MD)[colnames(MD) == "Item1"] <- "Timely admission"  colnames(MD)[colnames(MD) == "Item2"] <- "Timely treatment"  colnames(MD)[colnames(MD) == "Item3"] <- "Timely visits"  colnames(MD)[colnames(MD) == "Item4"] <- "Reliability"  colnames(MD)[colnames(MD) == "Item5"] <- "Options"  colnames(MD)[colnames(MD) == "Item6"] <- "Hours of treatment"  colnames(MD)[colnames(MD) == "Item7"] <- "Courteous staff"  colnames(MD)[colnames(MD) == "Item8"] <- "Evidence of active listening from doctor"  #Change Item 1-8 names to relative descriptions  colnames(MD)  #View all column names  convert\_to\_numeric <- function(x) {  ifelse(x == "Yes", 1, 0)  }  #Convert columns that use variables Yes and No to numeric  MD[, c("HighBlood", "Stroke", "Complication\_risk", "Arthritis", "Diabetes",  "Hyperlipidemia", "BackPain", "Allergic\_rhinitis", "Reflux\_esophagitis", "Asthma", "ReAdmis", "Soft\_drink")] <- lapply(MD[, c("HighBlood", "Stroke", "Complication\_risk", "Arthritis", "Diabetes",  "Hyperlipidemia", "BackPain", "Allergic\_rhinitis", "Reflux\_esophagitis", "Asthma", "ReAdmis", "Soft\_drink")], convert\_to\_numeric)  #Converting Yes/NO to numeric  MD$Zip <- as.character(MD$Zip)  #Convert zip codes to character type  MD$Zip <- str\_pad(MD$Zip, width = 5, pad = "0")  #Add leading zeros to zip code  ggplot(data = MD, aes(x = Lng, y = Lat)) +  geom\_point() +  labs(x = "Longitude", y = "Latitude") +  theme\_bw()  # lat and lng  children\_median <- median(MD$Children, na.rm = TRUE)  #Replace null values for median  MD$Children[is.na(MD$Children)] <- children\_median  # Replace missing values with the mean  median\_income <- median(MD$Income, na.rm = TRUE)  # Calculate the mean of the non-missing values  MD$Income[is.na(MD$Income)] <- median\_income  # Replace missing values with the mean  mean\_Age <- mean(MD$Age, na.rm = TRUE)  # Calculate the mean of the non-missing values  MD$Age[is.na(MD$Age)] <- mean\_Age  #Replace missing values with the mean  mean\_Initial\_days <- mean(MD$Initial\_days, na.rm = TRUE)  # Calculate the mean of the non-missing values  MD$Initial\_days[is.na(MD$Initial\_days)] <- mean\_Initial\_days  #Replace missing values with the mean  missing\_sum <- colSums(is.na(MD))  # Calculate the sum of missing values  print(missing\_sum)  # Print the sum of missing values  # Print the sum of missing values  MD$Age <- round(MD$Age)  #Round the variable age  print(MD$Age)  #Print the new rounded age column  Mode <- function(x) {  ux <- unique(x)  ux[which.max(tabulate(match(x, ux)))]  }  #Function to calculate mode  mode\_overweight <- Mode(MD$Overweight)  #calculate mode for overweight column  MD$Overweight[is.na(MD$Overweight)] <- mode\_overweight  #Replace null values with mode  mode\_Anxiety <- Mode(MD$Anxiety)  #calculate mode for anxiety  MD$Anxiety[is.na(MD$Anxiety)] <- Mode(MD$Anxiety)  #Impute mode in anxiety column  mode\_Soft\_drink <- Mode(MD$Soft\_drink)  #Calculate mode for soft drink  MD$Soft\_drink[is.na(MD$Soft\_drink)] <- Mode(MD$Soft\_drink)  #Fill in NA value with mode  View(MD)  #View MD  variables\_of\_interest <- c("Lat", "Lng", "Population", "Children", "Age", "Income",  "ReAdmis", "VitD\_levels", "Doc\_visits", "Full\_meals\_eaten",  "VitD\_supp", "Soft\_drink", "HighBlood", "Stroke",  "Complication\_risk", "Overweight", "Arthritis", "Diabetes",  "Hyperlipidemia", "BackPain", "Anxiety", "Allergic\_rhinitis",  "Reflux\_esophagitis", "Asthma", "Initial\_days", "TotalCharge",  "Additional\_charges", "Timely admission", "Timely treatment",  "Timely visits", "Reliability", "Options", "Hours of treatment")  #Create new dataframe  df <- MD[, variables\_of\_interest]  #Create new data frame  non\_numeric\_cols <- sapply(df, function(x) !is.numeric(x))  #Identify the columns in df that are not numeric  df[!non\_numeric\_cols] <- lapply(df[!non\_numeric\_cols], as.numeric)  #Convert the non nuemric column excluding char or factors  detect\_outliers <- function(x) {  q1 <- quantile(x, 0.25, na.rm = TRUE)  q3 <- quantile(x, 0.75, na.rm = TRUE)  iqr <- q3 - q1  lower\_fence <- q1 - 1.5 \* iqr  upper\_fence <- q3 + 1.5 \* iqr  outliers <- x[x < lower\_fence | x > upper\_fence]  return(outliers)  }  # Detect outliers in each column  outliers\_list <- lapply(df, detect\_outliers)  # Identify columns with outliers  columns\_with\_outliers <- names(df)[sapply(outliers\_list, length) > 0]  #Identify columns with outliers  print(columns\_with\_outliers)  #print column outliers  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Lat)) +  labs(x = "", y = "Lat") +  theme\_bw() +  ggtitle("Box Plot of Lat")  #Boxplot Lat  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Lng)) +  labs(x = "", y = "Lng") +  theme\_bw() +  ggtitle("Box Plot of Lng")  #Boxplot for LNG  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Population)) +  labs(x = "", y = "Population") +  theme\_bw() +  ggtitle("Box Plot of Population")  #Boxplot for Population  ggplot(MD) +  geom\_boxplot(aes(x = "", y = Income)) +  labs(x = "", y = "Income") +  theme\_bw() +  ggtitle("Box Plot of Income")  #Boxplot for Income  boxplot\_TotalCharge <- boxplot(df$TotalCharge)  #boxplot for total charge  boxplot\_Additional\_charges <- boxplot(df$Additional\_charges)  #Additional Charge boxplot  boxplot\_Timely\_admission <- boxplot(df$`Timely admission`)  #boxplot for TA  boxplot\_Timely\_treatment <- boxplot(df$`Timely treatment`)  #Boxplot TT  boxplot\_Timely\_visits <- boxplot(df$`Timely visits`)  #Boxplot TV  boxplot\_Reliability <- boxplot(df$Reliability)  #boxplot Reliabilty  boxplot\_Options <- boxplot(df$Options)  #Boxplot options  boxplot\_Hours\_of\_treatment <- boxplot(df$`Hours of treatment`)  #boxplot Hours of Treatment  Unclean <- read\_csv("C:/Users/merce/Downloads/medical\_raw\_data.csv")  #Unclean data  columns <- c("Children", "Soft\_drink", "Anxiety", "Income", "Overweight", "Initial\_days", "Age")  # columns for unclean histogram  Unclean[columns] <- lapply(Unclean[columns], function(x) as.numeric(x, na.rm = TRUE))  #Turn char values to numeric  par(mfrow = c(2, 4))  #set up layout  for (col in columns) {  values <- Unclean[[col]]  values <- values[!is.na(values)]  # Filter out missing values  if (!is.null(values) && length(values) > 0)  hist(values, main = col, xlab = col, col = "lightblue")  }  #Create histogram for unclean  columns <- c("Children", "Soft\_drink", "Anxiety", "Income", "Overweight", "Initial\_days", "Age")  #Select column from new\_MD  par(mfrow = c(2, 4))  #set the layout of the subplots  for (col in columns) {  hist(MD[[col]], main = col, xlab = col, col = "lightblue")  }  #Histogram for cleaned data MD  df <- MD[, c("Income", "Lat", "Lng", "VitD\_levels", "Initial\_days", "Additional\_charges", "TotalCharge")]  #selected variables for eigen values  pca\_result <- prcomp(df, scale. = TRUE)  #scale pca  eigen\_values <- pca\_result$sdev^2  #extract pca  print(eigen\_values)  # Print the eigenvalues  pc\_numbers <- 1:length(eigen\_values)  #create sequence  barplot(eigen\_values, names.arg = pc\_numbers,  xlab = "Principal Component", ylab = "Eigenvalue",  main = "Scree Plot")  #create scree plot  percentage\_outliers <- length(columns\_with\_outliers) / nrow(MD) \* 100  # Find the percentage of outliers  percentage\_remaining <- 100 - percentage\_outliers  #Calculate the percentage remaining  cat("Percentage of outliers:", percentage\_outliers, "%\n")  cat("Percentage of data remaining:", percentage\_remaining, "%\n")  #print results  file\_path <- "C:/Users/merce/Downloads/MD.csv"  #file path  write.csv(MD, file = file\_path, row.names = FALSE)  I have attached the Rscript File done just incase of formatting issues from copy and paste.  #Create csv  D5. The file is attached below. The csv file is named MD and is listed in the attachments. The code to create the csv file is down below:    D6.  The main limitation of the cleaning process is the absence of an opportunity to discuss the data variables with someone knowledgeable about the current condition of the hospital. This lack of expertise prevents gaining valuable insights into how to handle outliers effectively and whether it is necessary to remove them. Additionally, filling in missing values with assumed appropriate values might not accurately reflect the true nature of the null values. Having the perspective of an expert would have provided more clarity and guidance, addressed these limitations and ensured a more accurate and informed data cleaning process.  D7.  One of the main challenges that can arise when attempting to answer the research question is the reliance on assumptions regarding the treatment of certain variables due to a limited understanding of their significance. This lack of knowledge can lead to potential biases or inaccuracies in the analysis and interpretation of the data. To overcome this challenge, it is crucial to consult with subject matter experts or individuals with domain expertise who can provide insights into the importance and nuances of the variables under investigation. Their expertise can help ensure that the variables are appropriately handled and interpreted, enhancing the validity and reliability of the research findings.  E.  E1.  I applied PCA to all continuous numerical variables, including income, latitude, longitude, VitD\_levels, initial days, additional charges, and total charges. By utilizing PCA, I aimed to reduce the dimensionality of the dataset and capture the most significant patterns and variations within these variables.  Code that displayed eigen values for the created data set df. The data set consists of all continuous variables in the data set MD.    Loading matrix    The code that created the scree plot :      The scree plot is above.  E2.  In deciding which dimensions to keep for further analysis, I focused on the first six dimensions as they exhibited strong variance. These six dimensions collectively explained approximately 99% of the total variance in the data, making them crucial for capturing the most significant patterns and information. By retaining these dimensions, I ensured that I retained most of the variability present in the dataset, allowing for a comprehensive representation of the underlying data structure. Moreover, these dimensions likely contain the most relevant and influential features that contribute to the overall variability and characteristics of the data. Keeping these six dimensions provides a concise yet informative representation of the dataset, facilitating more efficient and effective analyses and interpretations.    E3.  Applying (PCA) offers numerous benefits in data analysis by giving a data analyst the ability to see patterns in a data set. Principal component analysis allows for the detection of modes (information reduction) by transforming a set of correlated variables into a smaller set of uncorrelated variables called principal components. The component values that are close to 1.0 are “components that capture the greatest amount of variance in the data” (Roweis, 1997). PCA helps simplify the analysis and visualization of complex datasets, creating the identification of dominant patterns, and enhances understanding of the viewers. Th six variables that make up each PC are income, latitude, longitude, VitD\_levels, initial days, additional charges, and total charges.  The code used to perform PCA as follows:  df <- MD[, c("Income", "Lat", "Lng", "VitD\_levels", "Initial\_days", "Additional\_charges", "TotalCharge")]  #selected variables for eigen values  PCA.pca <- prcomp(df, center = TRUE, scale. = TRUE)  #Perform PCA  loading\_matrix <- PCA.pca$rotation  #loading matrix  print(loading\_matrix)  #print loading matrix  singular\_values <- PCA.pca$sdev^2 |

**References**

Daniel T. Larose, & Chantal D. Larose. (2019). *Data Science Using Python and R*. Wiley.

De Jonge, E., & Van Der Loo, M. (2013). *An introduction to data cleaning with R*. Heerlen: Statistics Netherlands.

Middleton, K. (n.d.). Webinar 2. Western Govern University SharePoint. Retrieved from[https://westerngovernorsuniversity.sharepoint.com/sites/D206DataCleaning/Shared%20Documents/Forms/AllItems.aspx?id=%2Fsites%2FD206DataCleaning%2FShared%20Documents%2FWebinar%202%20Getting%20Started%20with%20Missing%20Data%2Epdf&parent=%2Fsites%2FD206DataCleaning%2FShared%20Documents&p=true&ga=1]

Middleton, K. (n.d.). Webinar 3. Western Govern University SharePoint. Retrieved from [<https://westerngovernorsuniversity.sharepoint.com/sites/D206DataCleaning/Shared%20Documents/Forms/AllItems.aspx?id=%2Fsites%2FD206DataCleaning%2FShared%20Documents%2FWebinar%203%20Getting%20Started%20with%20Re%2Dexpression%20of%20Categorical%20Variables%2Epdf&parent=%2Fsites%2FD206DataCleaning%2FShared%20Documents&p=true&ga=1>]

Middleton, K. (n.d.). Webinar 4. Western Govern University SharePoint. Retrieved from [<https://westerngovernorsuniversity.sharepoint.com/sites/D206DataCleaning/Shared%20Documents/Forms/AllItems.aspxid=%2Fsites%2FD206DataCleaning%2FShared%20Documents%2FWebinar%204%20Getting%20Started%20with%20PCA%2Epdf&parent=%2Fsites%2FD206DataCleaning%2FShared%20Documents&p=true&ga=1>]

Roweis, S. (1997). EM algorithms for PCA and SPCA. *Advances in neural information processing systems*, *10*.

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., ... & Grolemund, G. (2019). Welcome to the tidyverse. Journal of Open-Source Software, 4(43), 1686. doi: 10.21105/joss.01686