Informatics Institute of Technology

Data Analysis – CMM 703

Course Work

MSc in Big Data Analytics

Submitted by:

S.Z. Raeesul Islam

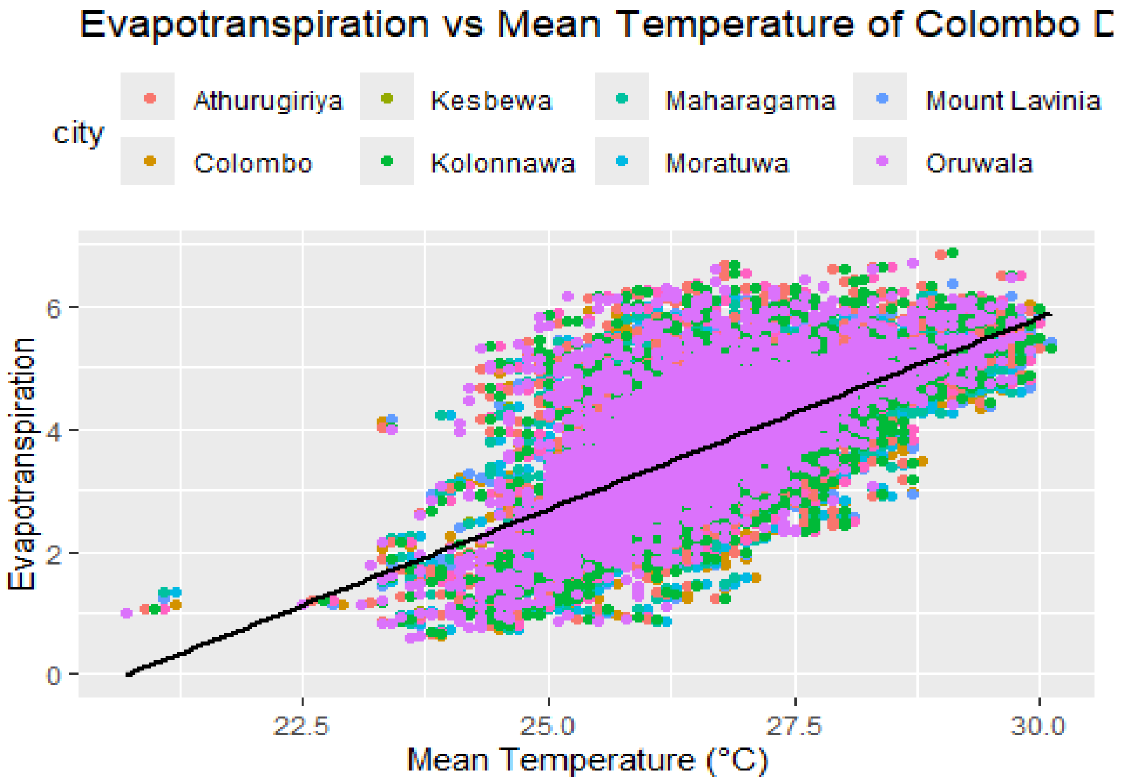
20232953/ 2409649

Contents

Source Code link: https://github.com/Raeesul25/data_analysis_R

Task 1.

```
# setup the working directory
setwd('G:/MSc in BDA/Semester 1/Data Analysis/Course Work')
getwd()

## [1] "G:/MSc in BDA/Semester 1/Data Analysis/Course Work"

# Load package
library(ggplot2)
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(caret)

## Warning: package 'caret' was built under R version 4.3.3

## Loading required package: lattice

library(shiny)

## Warning: package 'shiny' was built under R version 4.3.3

# Load the data set
weather_data = read.csv('02. SriLanka_Weather_Dataset.csv')

# Getting the data set dimensions
dim(weather_data)

## [1] 147480     24

# Defining cities of Colombo district
colomo_cities = c('Colombo', 'Mount Lavinia', 'Kesbewa', 'Moratuwa',
                  'Maharagama', 'Athurugiriya', 'Sri Jayewardenepura Kotte',
                  'Kolonnawa', 'Oruwala')

# Extract Colombo district data
colombo_dist = weather_data[weather_data$city %in% colomo_cities, ]

# Evaporation vs Temperature
# fit the linear model
lm_model = lm(et0_fao_evapotranspiration ~ temperature_2m_mean,
              data = colombo_dist)

ggplot(data = colombo_dist, aes(x = temperature_2m_mean,
                                y = et0_fao_evapotranspiration,
                                color = city)) +
  geom_point() +
  geom_smooth(method = lm, aes(linetype = "regression"),
              data = lm_model, color = "black") + # adding regression line
  labs(title = "Evapotranspiration vs Mean Temperature of Colombo District",
       x = "Mean Temperature (°C)",
       y = "Evapotranspiration") +
  theme(legend.position = "top")

## `geom_smooth()` using formula = 'y ~ x'
```

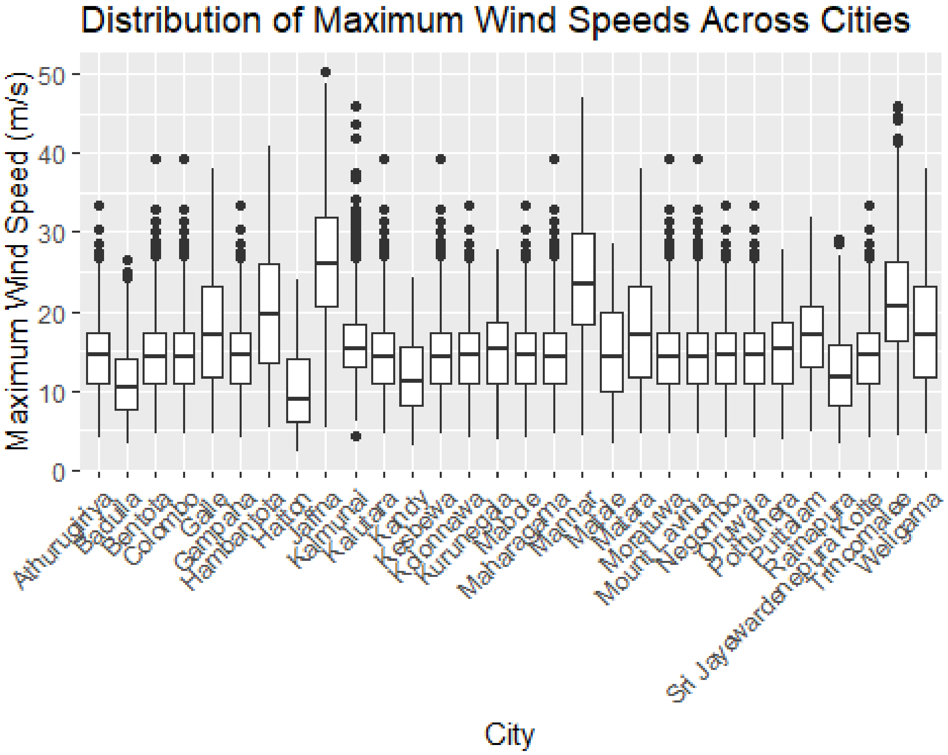


The scatterplot you sent shows the relationship between reference evapotranspiration (ET0) and mean temperature for cities in Colombo, Sri Lanka. Each data point represents a city, with the x-axis showing the mean temperature and the y-axis showing the reference evapotranspiration. There appears to be a positive correlation between mean temperature and reference evapotranspiration (ET0).

Improve the scatterplot:

To enhance the clarity of the scatterplot, several improvements can be made. First, adding labels for the axes is crucial to understanding the units of measurement for mean temperature and ET0. Secondly, jittering the data points on the x-axis can alleviate overlapping issues, improving the visibility of data spread. Finally, color coding the data points based on cities or regions could reveal patterns in ET0 rates across different locations, aiding in analysis and interpretation.

```r
# Wind Speed Distribution
ggplot(weather_data, aes(x = city, y = windspeed_10m_max)) +
  geom_boxplot() +
  labs(title = "Distribution of Maximum Wind Speeds Across Cities",
       x = "City",
       y = "Maximum Wind Speed (m/s)") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The boxplot explores the distribution of maximum wind speeds across different cities in Sri Lanka. The spread of the data points is large, indicating high variability in wind speeds between locations. The cities with the highest and lowest wind speeds are outliers, positioned far from the rest of the data. The highest wind speeds are in Hambantota and Puttalam, while the lowest wind speeds are in Badulla and Hatton.

Improve the scatterplot:

To enhance the boxplot, addressing text overlap on the x-axis is crucial. Abbreviating city names or rotating them can significantly improve readability. Additionally, incorporating the median and the interquartile range (IQR) would offer precise insights into the data's central tendency and dispersion, enhancing its informativeness.

Task 2.

Task 2.1.

Load the data and Explore basic information

Initially, load the lepto dataset and get the number of features and observations.

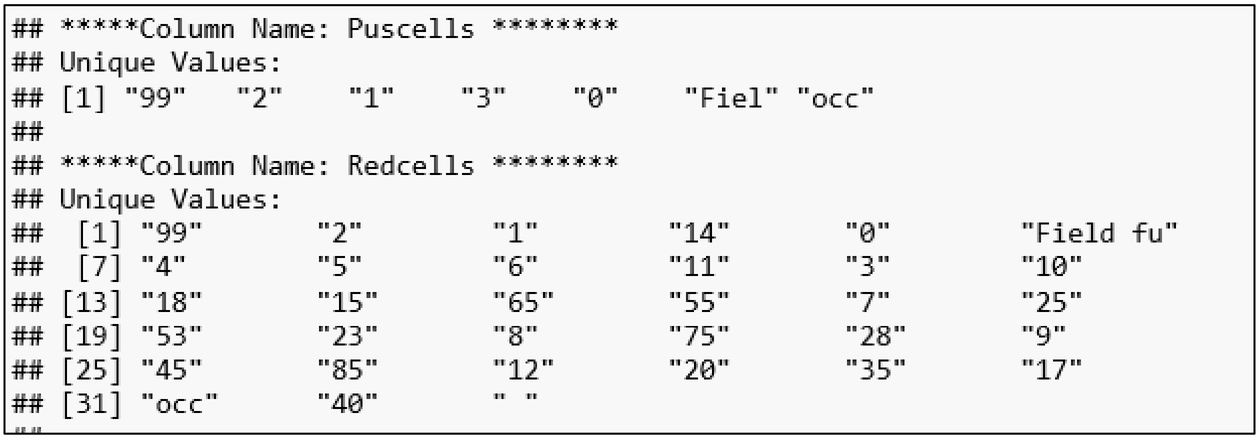
```
## Number of observations: 1734
```

```
## Number of features: 806
```

Then extract the column names and identify the type of each column.

```
## [1] "integer"   "numeric"   "character" "logical"
```

Based on the above output, the lepto dataset includes character columns. First, extract the character columns and get the unique values of those columns.



```
## *****Column Name: Puscells ********
## Unique Values:
## [1] "99"   "2"    "1"    "3"    "0"    "Fiel" "occ"
##
## *****Column Name: Redcells ********
## Unique Values:
##  [1] "99"       "2"        "1"        "14"       "0"        "Field fu"
##  [7] "4"        "5"        "6"        "11"       "3"        "10"
## [13] "18"       "15"       "65"       "55"       "7"        "25"
## [19] "53"       "23"       "8"        "75"       "28"       "9"
## [25] "45"       "85"       "12"       "20"       "35"       "17"
## [31] "occ"      "40"       " "
```

These character values are inconsistent so, let's convert character columns into numeric columns. Still logic columns are available. Let's extract logic columns and get unique values.

```
##   PomonaF
## "logical"
```

```
## [1] NA
```

There is a only column with a logic datatype and all values are null. So, let's remove that column.

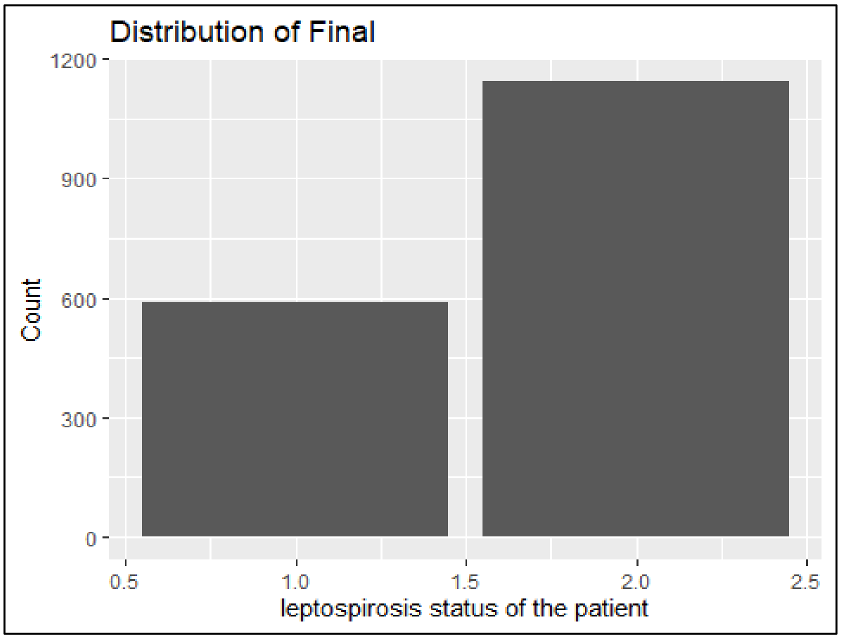
```
lepto_data = lepto_data[, -which(names(lepto_data) == "PomonaF")]
```

The serial feature is unique so, remove the serial feature also.

```
lepto_data = lepto_data[, -which(names(lepto_data) == "Serial")]
```

Analyze the Target Variable

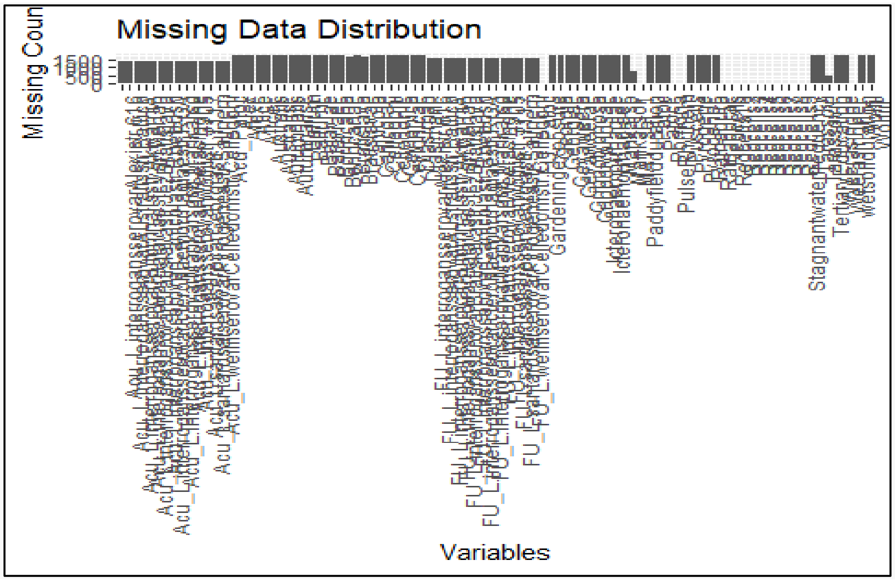
Next, analyze the Target feature. The target feature is named Final and plots the unique values in a bar plot.



Based on the bar plot, maximum observations have leptospirosis are not confirmed.

Analyze the Missing Data

Initially, extract the missing count of lepto data and create a data frame with variables and missing value counts. After that extract the missing value columns and missing value counts and plot the missing value data in a bar plot.



The above bar plot shows, most features have more than 1000 null records. Let’s remove columns with more than 50% null records.

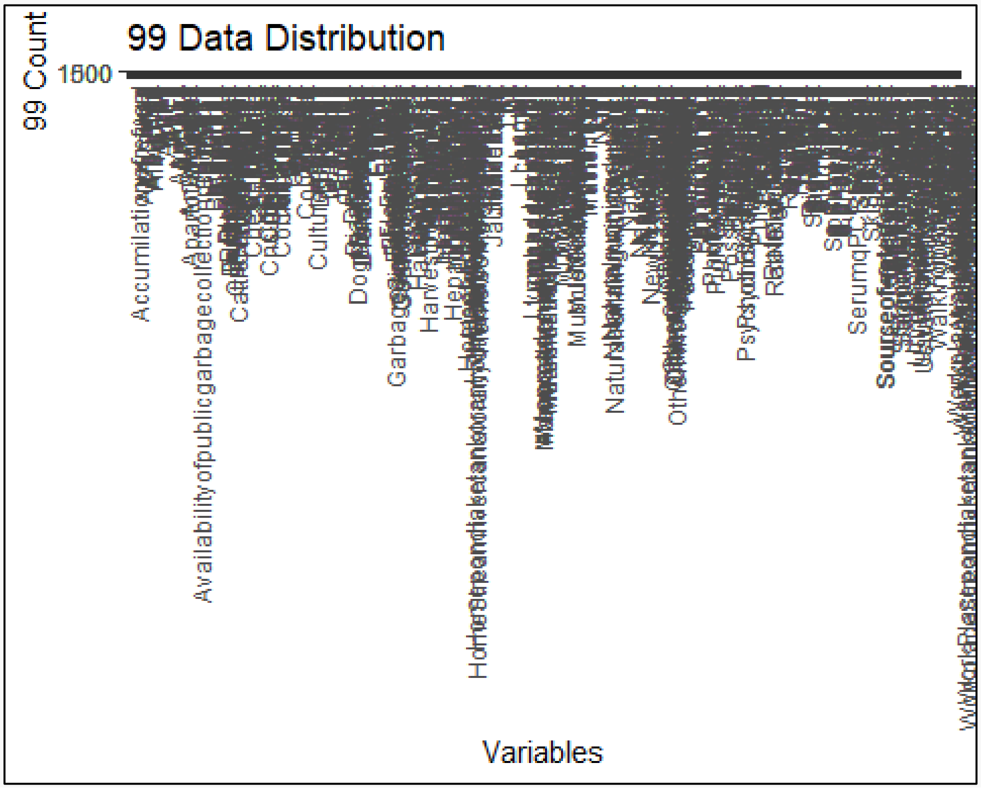
```
lepto_new = lepto_data[, !names(lepto_data) %in% many_na_cols]
```

After removing columns, there are 729 features available and still some columns have null values. Let’s replace null values with 99.

```
lepto_new = replace(lepto_new, is.na(lepto_new), 99)
```

Analyze the 99 value of Data

Initially, extract the 99 value count of lepto data and create a data frame with variables and missing value counts. After that extract the missing value columns and missing value count s and plot the missing value data in a bar plot.



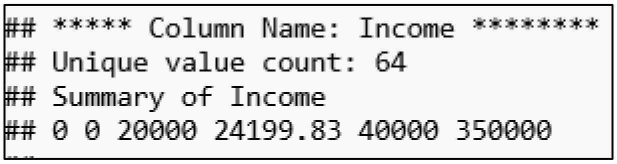
The above bar plot shows, that most features have more than 1000 records have 99 values. Let’s remove columns with more than 50% null records.

```
lepto_new1 = lepto_new[, !names(lepto_new) %in% many_99_cols]
```

After removing columns, there are 194 features available and still, some columns have a value of 99. Let’s replace 99 with 0.

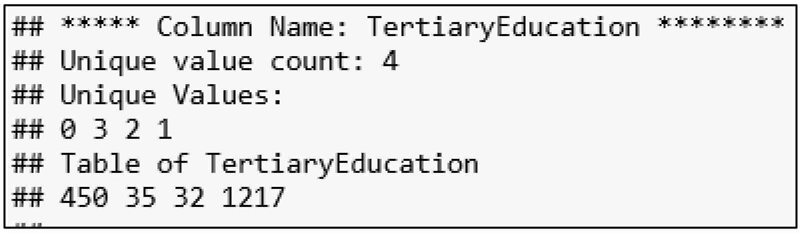
Analyze variables

Analyze the remaining features of lepto data and extract information from each column based on the condition using the analyze_vars function. If unique count is more than 12 then get the summary of the column.



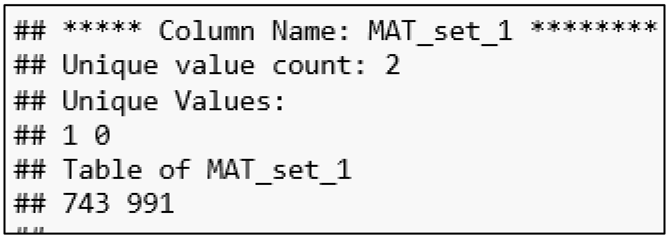
```
## ***** Column Name: Income ********
## Unique value count: 64
## Summary of Income
## 0 0 20000 24199.83 40000 350000
```

If unique count is less than 12 then get the table of the column.

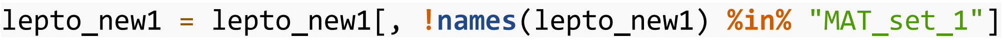


```
## ***** Column Name: TertiaryEducation ********
## Unique value count: 4
## Unique Values:
## 0 3 2 1
## Table of TertiaryEducation
## 450 35 32 1217
```

Based on the above analysis, the MAT_set_1 feature has 1 and 0 values. 0 represents unknown and there is no use for the model. so, remove that column.



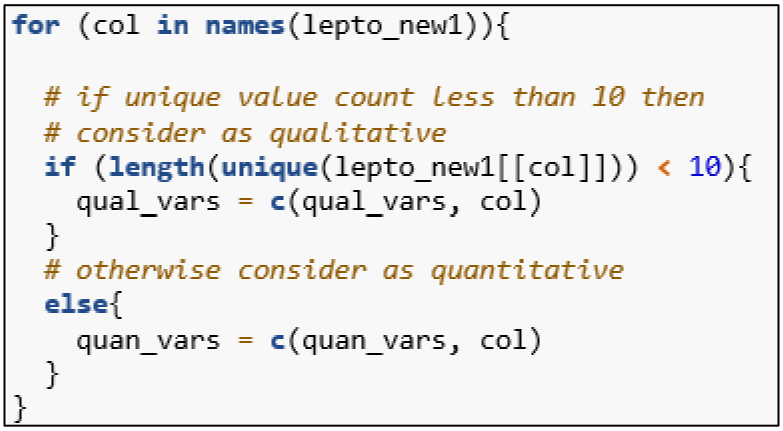
```
## ***** Column Name: MAT_set_1 ********
## Unique value count: 2
## Unique Values:
## 1 0
## Table of MAT_set_1
## 743 991
```



```r
lepto_new1 = lepto_new1[, !names(lepto_new1) %in% "MAT_set_1"]
```

Identify the qualitative and quantitative features

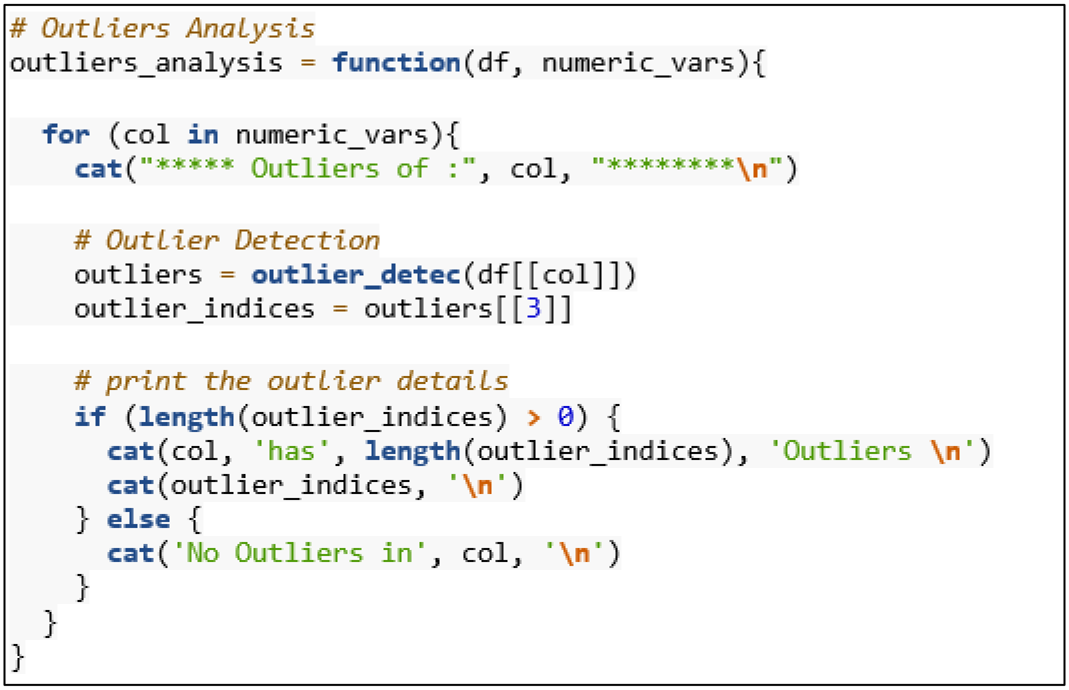
Identify the qualitative and quantitative features using the following conditions. If the length of unique values is less than 10 then append into qualitative otherwise quantitative.



```r
for (col in names(lepto_new1)){

  # if unique value count less than 10 then
  # consider as qualitative
  if (length(unique(lepto_new1[[col]])) < 10){
    qual_vars = c(qual_vars, col)
  }
  # otherwise consider as quantitative
  else{
    quan_vars = c(quan_vars, col)
  }
}
```

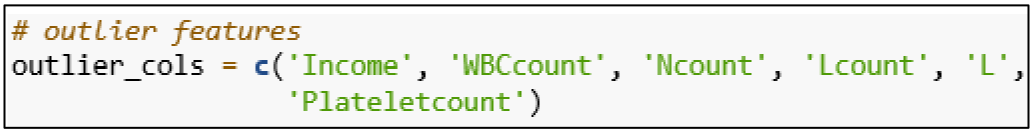
Identify the Outliers

Identify the outliers for numeric variables and get the summary of the outlier analysis using the outlier_analysis function.

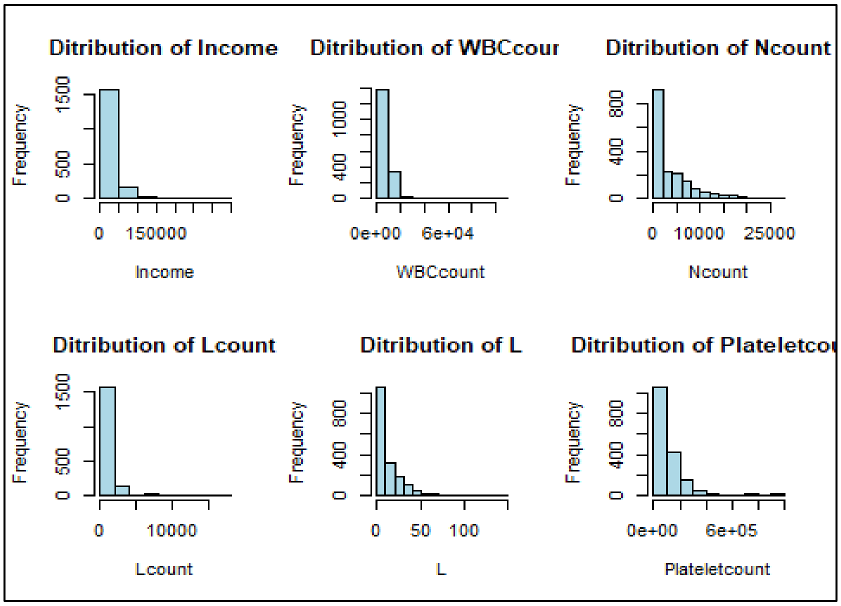


```
# Outliers Analysis
outliers_analysis = function(df, numeric_vars){

  for (col in numeric_vars){
    cat("***** Outliers of :", col, "********\n")

    # Outlier Detection
    outliers = outlier_detec(df[[col]])
    outlier_indices = outliers[[3]]

    # print the outlier details
    if (length(outlier_indices) > 0) {
      cat(col, 'has', length(outlier_indices), 'Outliers \n')
      cat(outlier_indices, '\n')
    } else {
      cat('No Outliers in', col, '\n')
    }
  }
}
```

Based on the analysis, extract the following columns that have outliers. Plot the histogram and check the data distribution of outlier variables.



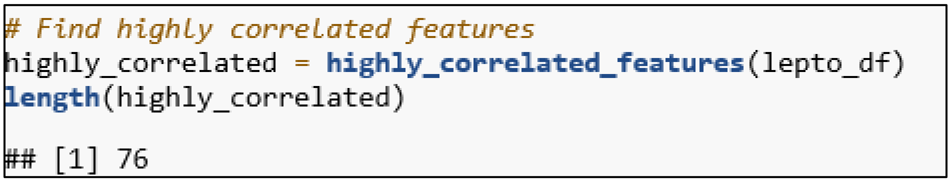
```
# outlier features
outlier_cols = c('Income', 'WBCcount', 'Ncount', 'Lcount', 'L',
                 'Plateletcount')
```



All the columns' data are distributed in the right-skewed. A robust imputation technique such as median imputation could be a good choice for these outliers. Let's do the outlier imputation using the outlier_imputation function.

Correlation Analysis

Let's do the correlation analysis for the remaining features of the lepto data. In this phase, extract features with a correlation value of more than 0.95 using the highly_correlated_features function and remove those features.



```
# Find highly correlated features
highly_correlated = highly_correlated_features(lepto_df)
length(highly_correlated)

## [1] 76
```

76 features are highly correlated with other features and remove those features.

After removing highly correlated features, there are 193 features available.

Task 2.2.

In the Final feature, 1 represented confirmed leptospirosis and 2 represented not confirmed leptospirosis. so, replace 'Final' feature value 2 into 0

```
lepto_df$Final[lepto_df$Final == 2] = 0
```

let's take backup of the data before do the data transformation

```
# Data Transformation
# backup the data
scaled_data = lepto_df

# Apply logarithmic transformation
scaled_data[, -ncol(scaled_data)] = log(scaled_data[, -ncol(scaled_data)] +
1)

# check are there any null records in lepto new data
colnames(scaled_data)[colSums(is.na(scaled_data)) > 0]

# if there are is null records, let's replace those null values with 0
scaled_data = replace(scaled_data, is.na(scaled_data), 0)

# check are there any null records in lepto new data
colnames(scaled_data)[colSums(is.na(scaled_data)) > 0]

# Split the Data
# Determine number of rows for training and testing
```

```
n_train = round(nrow(scaled_data) * 0.8)
n_test = nrow(scaled_data) - n_train

## Number of Training Records :  1387

## Number of Testing Records :  347

# Randomly shuffle the data
shuffled_data = scaled_data[sample(nrow(scaled_data)), ]

# Split data into training and testing sets
train_data = shuffled_data[1:n_train, ]
test_data = shuffled_data[(n_train + 1):(n_train + n_test), ]

# target feature has two values so, we need do binary classification
# logistic regression is good for binary classification

# build full logistic model
model = glm(Final ~ ., data = train_data, family=binomial(link=logit))
summary(model)

# Forward selection
forward_model = step(model, direction = "forward", trace = 0)
summary(forward_model)

# Backward selection
backward_model = step(model, direction = "backward", trace = 0)
summary(backward_model)

# backward model has low AIC value 1500.
# so, best model backward model
final_model = backward_model
summary(final_model)
```

Task 2.3.

```
# make a prediction for testing data
y_pred = predict(final_model, newdata = test_data, type = 'response')

# Change values based on condition
y_pred = ifelse(y_pred >= 0.5, 1, 0)
y_pred = factor(y_pred)

# get target values of test data
y_test = test_data$Final
y_test = factor(y_test)


# Compute confusion matrix
conf_matrix = confusionMatrix(y_pred, y_test)
```

```r
# Extract performance metrics
accuracy = conf_matrix$overall['Accuracy']
precision = conf_matrix$byClass['Precision']
recall = conf_matrix$byClass['Recall']
f1_score = conf_matrix$byClass['F1']

# Print the performance metrics
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.726224783861671"
```

```r
print(paste("Precision:", precision))
```

```
## [1] "Precision: 0.774590163934426"
```

```r
print(paste("Recall:", recall))
```

```
## [1] "Recall: 0.825327510917031"
```

```r
print(paste("F1 Score:", f1_score))
```

```
## [1] "F1 Score: 0.799154334038055"
```

Task 2.4.

```r
# identify non clinical features

non_clinical = c("Year", "Month", "Hospital", "Sample", "ICU", "OPD", "Sex",
                 "Age", "Ethnicity", "Income", "Education",
                 "TertiaryEducation", "Prophylactics", "Pasttreatments",
                 "Pastantibiotics", "Chronicillness", "Possibleexposure",
                 "Final")

# extract non clinical features
train_non_clinical = train_data[, names(train_data) %in% non_clinical]

# Print dimensions of non clinical data
print(dim(train_non_clinical))
```

```
## [1] 1387   18
```

```r
# build full logistic model
model_nc = glm(Final ~ ., data = train_non_clinical,
               family=binomial(link=logit))
summary(model_nc)

# Forward selection
forward_model_nc = step(model_nc, direction = "forward", trace = 0)
summary(forward_model_nc)
```

```r
# Backward selection
backward_model_nc = step(model_nc, direction = "backward", trace = 0)
summary(backward_model_nc)

# backward model has low AIC value 1687.8
# so, best model backward model
final_model_nc = backward_model_nc
summary(final_model_nc)

# AIC of non clinical data is less than AIC of whole lepto dataset.

# make a prediction for non clinical testing data
y_pred_nc = predict(final_model_nc, newdata = test_data,
                    type = 'response')

# Change values based on condition
y_pred_nc = ifelse(y_pred_nc >= 0.5, 1, 0)
y_pred_nc = factor(y_pred_nc)

# check the levels
levels(y_pred)
```

```
## [1] "0" "1"
```

```r
# Compute confusion matrix
conf_matrix_nc = confusionMatrix(y_pred_nc, y_test)

# Extract performance metrics
accuracy_nc = conf_matrix_nc$overall['Accuracy']
precision_nc = conf_matrix_nc$byClass['Precision']
recall_nc = conf_matrix_nc$byClass['Recall']
f1_score_nc = conf_matrix_nc$byClass['F1']

# Print the performance metrics
print(paste("Accuracy:", accuracy_nc))
```

```
## [1] "Accuracy: 0.62536023054755"
```

```r
print(paste("Precision:", precision_nc))
```

```
## [1] "Precision: 0.668941979522184"
```

```r
print(paste("Recall:", recall_nc))
```

```
## [1] "Recall: 0.85589519650655"
```

```r
print(paste("F1 Score:", f1_score_nc))
```

```
## [1] "F1 Score: 0.75095785440613"
```

```r
# Discussion
compare_df = data.frame(
  Dataset = c("Whole Data", "Non-Clinical"),
  Accuracy = c(accuracy, accuracy_nc),
  Precision = c(precision, precision_nc),
  Recall = c(recall, recall_nc),
  F1_Score = c(f1_score, f1_score_nc)
)
compare_df
```

```
##        Dataset  Accuracy Precision    Recall  F1_Score
## 1   Whole Data 0.7262248 0.7745902 0.8253275 0.7991543
## 2 Non-Clinical 0.6253602 0.6689420 0.8558952 0.7509579
```

```r
# whole data model better performance than non clinical data
# based on the accuracy, precision, recall, and f1 score of.
```

Task 3.

```r
# function Outlier Detection
outlier_detec = function(x){

  # Calculate quadrilles and IQR
  q1 = quantile(x, 0.25)
  q3 = quantile(x, 0.75)
  iqr = q3 - q1

  # Calculate lower and upper bounds for outliers
  lower_bound = q1 - 1.5 * iqr
  upper_bound = q3 + 1.5 * iqr

  # Identify outlier indices
  outlier_indices = which(x < lower_bound | x > upper_bound)

  return(list(lower_bound, upper_bound, outlier_indices))
}


# function for summarize variable
summarize_vars = function(df){

  # identify the continuous and non continuous features
  cont_vars = character()
  non_cont_vars = character()

  for (col in names(df)){
    unique_count = length(unique(df[[col]]))

    # if unique value count more than 4 then
    # consider as numerical
    if (unique_count > 4){
      cont_vars = c(cont_vars, col)
    }
    # otherwise consider as categorical
    else{
      non_cont_vars = c(non_cont_vars, col)
    }
  }

  return(list(cont_vars, non_cont_vars))
}


# function for convert categorical into numerical
```

```r
label_encoding = function(df, qualitative){
  for (col in qualitative){
    # extract unique values
    unique_values = unique(df[[col]])

    category_mapping = setNames(1:length(unique_values), unique_values)
    df[[col]] = category_mapping[df[[col]]]
  }
  return(df)
}


# function for best predictive model with performance evaluation
predictive_model = function(df, response) {

  response_len = length(unique(df[[response]]))
  col_type = sapply(df[response] ,class)

  # Check response variable type
  if (response_len == 2 & col_type == "integer") {
    type = "Binary"
  }
  else if(col_type == "numeric"){
    type = "Continuous"
  }
  else{
    print("Response variable must be continuous or binary")
    return("This function execution has ended")
  }

  # Determine number of rows for training and testing
  n_train = round(nrow(df) * 0.8)
  n_test = nrow(df) - n_train

  # Split data into training and testing sets
  train_data = df[1:n_train, ]
  test_data = df[(n_train + 1):(n_train + n_test), ]

  # Build the model
  if (type == "Binary") {
    # logistic model
    model_name = "Logistic Regression"
    model = glm(train_data[[response]] ~ ., data = train_data,
                family=binomial(link=logit))
  }else{
    # linear model
    model_name = "Linear Regression"
    model = lm(train_data[[response]] ~ ., data = train_data)
  }
  # Perform forward selection
  forward_model = step(model, direction = "forward", trace = 0)

  best_model = forward_model
  model_selection = "Forward Selection"

  # AIC value
  aic_value = AIC(best_model)

  # Get best features from the chosen model
  best_features = names(coef(best_model))[2:length(names(coef(best_model)))]

  # get target values of test data
  y_test = test_data[[response]]

  # model evaluation
  if (type == "Binary") {
    # make a prediction for testing data
    y_pred = predict(best_model, newdata = test_data,
                     type = 'response')

    # Change values based on condition
    y_pred = ifelse(y_pred >= 0.5, 1, 0)

    # Calculate the number of correct predictions
    correct_predictions = sum(y_test == y_pred)

    # Calculate accuracy
    accuracy = correct_predictions / length(y_test)

  } else {
    # make a prediction for testing data
    y_pred = predict(best_model, newdata = test_data)

    # Mean Squared Error (MSE) for continuous response
    accuracy = mean((y_pred - y_test)^2)
  }

  return(list(type = type, model_selection = model_selection,
              final_model = best_model, model_name = model_name,
              AIC_value = aic_value, accuracy = accuracy))
}


# Define the UI of R shiny dashborad
ui = fluidPage(
  titlePanel("Data Exploration and Modeling"),
  sidebarLayout(
    sidebarPanel(
      fileInput("file", "Upload Data CSV"),
      textInput("text", "Enter Text"),
      actionButton("process", "Process Data")
    ),

    mainPanel(
      tabsetPanel(
        tabPanel("Data Types", tableOutput("combined")),
        tabPanel("Missing Values",tableOutput("missing_values")),
        tabPanel("Outliers", tableOutput("outliers")),
        tabPanel("Plots", uiOutput("plots")),
        tabPanel("Summary", tableOutput("summary"))
      )
    )
  )
)

# Define the server logic
server = function(input, output) {

  # R function for Data Analysis
  data_analysis = function(df, response) {

    column_names = setdiff(names(df), response)
    qualitative_vars = list()
    quantitative_vars = list()
    missing_count = list()
    outlier_list = list()

    # Iterate through each column of the df
    for (col in names(df)) {

      if (is.numeric(df[, col])) {
        quantitative_vars[[col]] = "Quantitative"
        # Count missing values
        missing_count[[col]] = sum(is.na(df[[col]]))
        # Impute with mean
        df[[col]] = replace(df[[col]], is.na(df[[col]]),
                            mean(df[[col]], na.rm = TRUE))
        # Identify outliers
        outliers = outlier_detec(df[[col]])
        outlier_indices = outliers[[3]]

        # Print information about outliers (optional)
        if (length(outlier_indices) > 0) {
          outlier_list[[col]] = length(outlier_indices)
```

```r
        }
      }
      else {
        qualitative_vars[[col]] = "Qualitative"
        # Count missing values
        missing_count[[col]] = sum(is.na(df[[col]]))
        # Get mode of column
        mode_val = names(sort(table(df[[col]]), decreasing = TRUE)[1])
        df[[col]][is.na(df[[col]])] = mode_val
      }
    }

    # calling the summarize variable function
    summary_analysis = summarize_vars(df)

    # identify the continuous and non continuous features
    cont_vars = summary_analysis[[1]]
    non_cont_vars = summary_analysis[[2]]

    # store plots
    plots = list()

    # histogram plot for numerical variables
    for (col in cont_vars){
      plots[[col]] = ggplot(df, aes(x = df[[col]])) +
        geom_histogram(fill = "skyblue", color = "black") +
        labs(title = paste("Histogram of", col),
             x = col,
             y = "Frequency")
    }

    # Bar plot plot for categorical variables
    for (col in non_cont_vars){
      plots[[col]] = ggplot(df, aes(x = factor(df[[col]]))) +
        geom_bar(fill = "skyblue", color = "black") +
        labs(title = paste("Bar Plot of", col),
             x = col,
             y = "Frequency")
    }

    encoded_df = label_encoding(df, names(qualitative_vars))

    # calling the function of predictive model
    model_data = predictive_model(encoded_df, response)

    combined = append(qualitative_vars, quantitative_vars)
    type = model_data$type
    model_name = model_data$model_name
    model_selection = model_data$model_selection
    AIC_value = model_data$AIC_value
    accuracy = model_data$accuracy

    summary_list = list()
    summary_list[["Number of Observations"]] = nrow(df)
    summary_list[["Number of Features"]] = ncol(df)
    summary_list[["Qualitative Features"]] = length(names(qualitative_vars))
    summary_list[["Quantitative Features"]] =
length(names(quantitative_vars))
    summary_list[["Features with Outlier"]] = length(names(outlier_list))
    summary_list[["Type of Target Feature"]] = type
    summary_list[["Suitable Model Name"]] = model_name
    summary_list[["Selected Best Model"]] = model_selection
    summary_list[["AIC of Best Model"]] = AIC_value
    summary_list[["Model Evaluation"]] = accuracy

    return(list(comlist = combined,
                qualitative = qualitative_vars,
                quantitative = quantitative_vars,
                missing_values = missing_count,
                outlier_values = outlier_list,
                plots = plots,
                type = model_data$type,
                model_name = model_data$model_name,
                model_selection = model_data$model_selection,
                AIC = model_data$AIC_value,
                accuracy = model_data$accuracy,
                summary_data = summary_list))
  }

  # Reactive expression to process data when button is clicked
  data_processed = eventReactive(input$process, {
    req(input$file)
    data = read.csv(input$file$datapath)
    text = input$text
    data_analysis(data, text)
  })

  # for data types
  output$combined = renderTable({
    combined = data_processed()$comlist
    df = data.frame(
      "Column Name" = names(combined),
      "Data Type" = unlist(combined),
      stringsAsFactors = FALSE
    )
    df
  }, rownames = FALSE)
  # for missing values
  output$missing_values = renderTable({
    missing = data_processed()$missing_values
    df = data.frame(
      "Column Name" = names(missing),
      "Missing Count" = unlist(missing),
      stringsAsFactors = FALSE
    )
    if (length(missing) == 0) {
      paste("No Missing Values")
    }else{df}

  }, rownames = FALSE)

  # for outliers
  output$outliers = renderTable({
    outliers = data_processed()$outlier_values
    df = data.frame(
      "Column Name" = names(outliers),
      "Outliers Count" = unlist(outliers),
      stringsAsFactors = FALSE
    )
    if (length(missing) == 0) {
      paste("No Outliers detected")
    }else{df}

  }, rownames = FALSE)

  # for plots
  output$plots = renderUI({
    plots = data_processed()$plots
    plot_list = lapply(names(data_processed()$plots), function(col) {
      plotOutput(paste0("plot_", col))
    })

    # Arrange the plots in columns and rows
    do.call(tagList, lapply(seq_along(data_processed()$plots), function(i) {
      fluidRow(column(width = 6, plot_list[i]))
    }))
  })

  # Render each plot
  observe({
    plots = data_processed()$plots
    for (col in names(data_processed()$plots)) {
      output[[paste0("plot_", col)]] = renderPlot({
        data_processed()$plots[[col]]
      })
    }
  })

  # for model data
  output$summary = renderTable({
    summary = data_processed()$summary_data
    df = data.frame(
      "Type" = names(summary),
      "Data" = unlist(summary),
      stringsAsFactors = FALSE )
    if (length(missing)== 0) {
      paste("No Missing Values")
    }
    else{df}
  }, rownames = FALSE)
}

# Run the application
shinyApp(ui = ui, server = server)
```

Appendix

Appendix of Task 02.

```
# ********** Task 2.1. **********
```

```
# Load the data set
lepto_data = read.csv('02. lepto_data.csv')


# Number of observations and features:
cat("Number of observations:", nrow(lepto_data), "\n")
```

```
cat("Number of features:", ncol(lepto_data), "\n")

# extract column names
cols = names(lepto_data)


# Extract column names and their types
col_types = sapply(lepto_data, class)
unique(col_types)

# Extract 'character' features only
char_vars = col_types[col_types == 'character']

# Checking what are the unique values and
# how many rows for each unique value has in character features
for (col in names(char_vars)) {
  # extract unique values of character features
  char_unique = unique(lepto_data[, col])


  # Print information about the column
  cat("*****Column Name:", col, "********\n")
  cat("Unique Values:\n")
  print(char_unique)
  cat("\n")
}
```

```
for (col in names(char_vars)){
  lepto_data[[col]] = as.numeric(lepto_data[[col]])
}
```

```
# recheck column type
unique(sapply(lepto_data, class))
```

```
## [1] "integer" "numeric" "logical"
```

```
# Extract 'logical' features only
log_vars = col_types[col_types == 'logical']
log_vars
```

```r
# only PomonaF feature is logical, get unique value of PomonaF
unique(lepto_data$PomonaF)

# print dimension of lepto data
print(dim(lepto_data))
## [1] 1734  804

# Analyze Target Variable
# table of Final
table(lepto_data$Final)

##
##    1    2
##  591 1143

# Summary of the "Final" variable
summary(lepto_data$Final)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   2.000   1.659   2.000   2.000

# Plot the distribution using bar chart
ggplot(lepto_data, aes(x = Final)) +
  geom_bar(stat = "count") +
  labs(title = "Distribution of Final",
       x = "leptospirosis status of the patient",
       y = "Count")

# Analyze the Missing data
# get the missing counts of each feature
missing_data = colSums(is.na(lepto_data))

# create a dataframe for missing data
missing_df = data.frame(Variables = names(missing_data),
                        Missing_Count = missing_data)

# get only null records
missing_df = missing_df[missing_df$Missing_Count != 0, ]

# visualize bar plot for null counts
ggplot(data = missing_df, aes(x = Variables, y = Missing_Count)) +
  geom_bar(stat = "identity") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Missing Data Distribution",
       x = "Variables",
       y = "Missing Count")

# get 50% of record counts
c = nrow(lepto_data) * 0.5

# get records with more than 50% NA values
```

```r
many_na_cols = missing_df$Variables[missing_df$Missing_Count > c]

# Remove features with more than 1000 null values
lepto_new = lepto_data[, !names(lepto_data) %in% many_na_cols]

# print dimension of new lepto data
print(dim(lepto_new))
```

```
## [1] 1734  729
```

```r
# still there are some null values
# so, let's replace those null values with 99
lepto_new = replace(lepto_new, is.na(lepto_new), 99)

# check are there any null records in lepto new data
colnames(lepto_new)[colSums(is.na(lepto_new)) > 0]
```

```
## character(0)
```

```r
# Analyze features with 99 values
# get the missing counts of each feature
data_99 = colSums(lepto_new == 99)

# create a dataframe for missing data
df_99 = data.frame(Variables = names(data_99),
                   Count_99 = data_99)

# get only null records
df_99 = df_99[df_99$Count_99 != 0, ]

# visualize bar plot for null counts
ggplot(data = df_99, aes(x = Variables, y = Count_99)) +
  geom_bar(stat = "identity") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "99 Data Distribution",
       x = "Variables",
       y = "99 Count")

# get records with more than 50% 99 values
many_99_cols = df_99$Variables[df_99$Count_99 > c]
length(many_99_cols)
```

```
## [1] 535
```

```r
# Remove features with more than 1000 null values
lepto_new1 = lepto_new[, !names(lepto_new) %in% many_99_cols]

# print dimension of new lepto data
print(dim(lepto_new1))
```

```
## [1] 1734  194
```

```
# still there are some 99 values
# so, let's replace those 99 values with 0
lepto_new1[lepto_new1 == 99] = 0

# check are there any 99 values in lepto new data
colnames(lepto_new1)[colSums(lepto_new1 == 99) > 0]
```

```
## character(0)
```

```
# function for analyze given variable
analyze_vars = function(df){

  for (col in names(df)){
    unique_count = length(unique(df[[col]]))

    cat("***** Column Name:", col, "********\n")
    cat('Unique value count:', length(unique(df[, col])), '\n')

    # if unique value count more than 12 then
    # consider as numerical
    if (unique_count > 12){
      # summary statistic of the variable
      cat('Summary of', col, '\n')
      cat(summary(df[[col]]), '\n')
      cat('\n')
    }
    # otherwise consider as categorical
    else{
      # Analyze variable
      cat("Unique Values:\n")
      cat(unique(df[, col]), '\n')
      cat('Table of', col, '\n')
      cat(table(df[[col]]), '\n')
      cat('\n')
    }
  }
}

# calling the analyze function
analyze_vars(lepto_new1)
```

```
## ***** Column Name: MAT_set_1 ********
## Unique value count: 2
## Unique Values:
## 1 0
## Table of MAT_set_1
## 743 991
##
```

```
## ***** Column Name: Final ********
## Unique value count: 2
## Unique Values:
## 2 1
## Table of Final
## 591 1143
```

```r
# MAT_set_1 feature has 0 and 1 value. its mean only one value
# so, remove the MAT_set_1 feature
lepto_new1 = lepto_new1[, !names(lepto_new1) %in% "MAT_set_1"]

# print dimension of new lepto data
print(dim(lepto_new1))
```

```
## [1] 1734  193
```

```r
# identify the qualitative and quantitative features
qual_vars = character()
quan_vars = character()

for (col in names(lepto_new1)){

  # if unique value count less than 10 then
  # consider as qualitative
  if (length(unique(lepto_new1[[col]])) < 10){
    qual_vars = c(qual_vars, col)
  }
  # otherwise consider as quantitative
  else{
    quan_vars = c(quan_vars, col)
  }
}
```

```r
# Outlier Detection
outlier_detec = function(x){

  # Calculate quadrilles and IQR
  q1 = quantile(x, 0.25)
  q3 = quantile(x, 0.75)
  iqr = q3 - q1

  # Calculate lower and upper bounds for outliers
  lower_bound = q1 - 1.5 * iqr
  upper_bound = q3 + 1.5 * iqr

  # Identify outlier indices
  outlier_indices = which(x < lower_bound | x > upper_bound)

  return(list(lower_bound, upper_bound, outlier_indices))
}
```

```r
# Outliers Analysis
outliers_analysis = function(df, numeric_vars){

  for (col in numeric_vars){
    cat("***** Outliers of :", col, "********\n")

    # Outlier Detection
    outliers = outlier_detec(df[[col]])
    outlier_indices = outliers[[3]]

    # print the outlier details
    if (length(outlier_indices) > 0) {
      cat(col, 'has', length(outlier_indices), 'Outliers \n')
      cat(outlier_indices, '\n')
    } else {
      cat('No Outliers in', col, '\n')
    }
  }
}

# calling outlier detection function
outliers_analysis(lepto_new1, quan_vars)
```

```r
# outlier features
outlier_cols = c('Income', 'WBCcount', 'Ncount', 'Lcount', 'L',
                 'Plateletcount')

# plot the histogram to check the distribution
par(mfrow = c(2, 3))
for (col in outlier_cols){
  hist(lepto_new1[[col]], main = paste("Ditribution of", col),
       xlab = col,
       col = "lightblue",
       border = "black")
}

# Outlier columns are distributed in right skewed, a robust imputation
# technique such as median imputation could be a good choice
# Outlier Imputation
outlier_imputation = function(df){
  for (col in outlier_cols){
    cat("***** Outliers Imputation for :", col, "********\n")

    # Calculate median
    median_value = median(df[[col]], na.rm = TRUE)

    # outlier detection
    outliers = outlier_detec(df[[col]])
    lower_limit = outliers[[1]]
    upper_limit = outliers[[2]]
    cat('Lower limit :', lower_limit, '\n')
    cat('Upper limit :', upper_limit, '\n')

    # impute for lower outliers
    df[[col]][df[[col]] < lower_limit] = median_value

    # impute for upper outliers
    df[[col]][df[[col]] > upper_limit] = median_value

    cat("***** Done Outliers Imputation for :", col, "********\n\n")
  }

  return(df)
}

# calling outlier imputation function
lepto_df = outlier_imputation(lepto_new1)
```

```
## ***** Outliers Imputation for : Income ********
## Lower limit : -60000
## Upper limit : 1e+05
## ***** Done Outliers Imputation for : Income ********
##
## ***** Outliers Imputation for : WBCcount ********
## Lower limit : -13571.25
## Upper limit : 22618.75
## ***** Done Outliers Imputation for : WBCcount ********
##
## ***** Outliers Imputation for : Ncount ********
## Lower limit : -8205
## Upper limit : 13675
## ***** Done Outliers Imputation for : Ncount ********
##
## ***** Outliers Imputation for : Lcount ********
## Lower limit : -1736.25
## Upper limit : 2893.75
## ***** Done Outliers Imputation for : Lcount ********
##
## ***** Outliers Imputation for : L ********
## Lower limit : -25.58728
## Upper limit : 42.64547
## ***** Done Outliers Imputation for : L ********
##
## ***** Outliers Imputation for : Plateletcount ********
## Lower limit : -219000
## Upper limit : 365000
## ***** Done Outliers Imputation for : Plateletcount ********
```

```
# Correlation analysis
# Function to identify highly correlated features
highly_correlated_features <- function(df, threshold = 0.95) {

  # Calculate correlation matrix
  corr_matrix = cor(df)

  # Exclude self-correlations on the diagonal
  diag(corr_matrix) = 0

  # Get indices of highly correlated features
  highly_correlated_indices = which(abs(corr_matrix) > threshold,
                                    arr.ind = TRUE)

  # Convert indices to feature names
  features = rownames(corr_matrix)[highly_correlated_indices[, 1]]

  # Remove duplicates
  features = unique(features)

  return(features)
}

# Find highly correlated features
highly_correlated = highly_correlated_features(lepto_df)
length(highly_correlated)
```

```
## [1] 76
```

```
# remove highly correlated features
lepto_df = lepto_df[, !names(lepto_df) %in% highly_correlated]

# print dimension of new Lepto data
print(dim(lepto_df))
```

```
## [1] 1734  117
```

```
# Print dimensions of training and testing sets
print(dim(train_data))
```

```
## [1] 1387  117
```

```
print(dim(test_data))
```

```
## [1] 347 117
```

```
# check the levels
levels(y_test)
```

```
## [1] "0" "1"
```

```r
levels(y_pred)
```

```
## [1] "0" "1"
```

```r
# build full logistic model
model = glm(Final ~ ., data = train_data, family=binomial(link=logit))
summary(model)
```

```
## 
## Call:
## glm(formula = Final ~ ., family = binomial(link = logit), data = train_dat
a)
## 
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)    
## (Intercept)           -5.646e+03  2.043e+03  -2.763 0.005719 ** 
## Year                   7.418e+02  2.685e+02   2.763 0.005728 ** 
## Month                  1.658e-01  1.421e-01   1.167 0.243361    
## Hospital               8.269e-01  2.217e-01   3.730 0.000192 ***
## Sample                -1.325e+00  9.461e-01  -1.401 0.161316    
## ICU                    1.268e+00  9.201e-01   1.379 0.168029    
## OPD                   -5.424e-01  9.449e-01  -0.574 0.565969    
## Sex                   -5.834e-01  4.908e-01  -1.189 0.234569    
## Age                    4.509e-01  1.314e-01   3.432 0.000599 ***
## Ethnicity             -5.179e-01  5.198e-01  -0.996 0.319133    
## Income                 1.643e-02  2.234e-02   0.736 0.461961    
## Education             -1.853e-01  1.488e-01  -1.246 0.212789    
## TertiaryEducation      3.407e-01  3.166e-01   1.076 0.281746    
## Prophylactics         -8.908e-02  4.519e-01  -0.197 0.843750    
## Pasttreatments        -2.606e-01  4.481e-01  -0.582 0.560858    
## Pastantibiotics        4.091e-01  3.406e-01   1.201 0.229679    
## Chronicillness         6.536e-01  4.047e-01   1.615 0.106293    
## Possibleexposure      -1.616e+00  4.278e-01  -3.778 0.000158 ***
## Feveronset             1.672e+00  1.099e+00   1.521 0.128344    
## Headacheonset          3.120e-02  7.002e-01   0.045 0.964456    
## Musclepainonset       -9.847e-01  9.921e-01  -0.993 0.320948    
## Cnsuffusiononset      -6.244e-01  7.930e-01  -0.787 0.431047    
## Jaundiceonset          8.189e-01  8.376e-01   0.978 0.328230    
## SOBonset               9.988e-01  9.903e-01   1.009 0.313200    
## Coughonset             4.053e-01  8.143e-01   0.498 0.618717    
## Chestpainonset         2.659e-01  1.010e+00   0.263 0.792299    
## Nauseaonset            4.963e-01  8.005e-01   0.620 0.535245    
## Vomitingonset         -1.637e+00  6.204e-01  -2.638 0.008340 ** 
## Diarrhoeaonset        -1.054e+00  7.113e-01  -1.482 0.138248    
## Prostrationonset       1.497e+00  7.187e-01   2.083 0.037295 *  
## Rigorsonset           -3.767e-01  8.004e-01  -0.471 0.637903    
## Photophobiaonset      -1.648e+00  1.131e+00  -1.457 0.145026    
## Chillsonset            9.980e-01  9.202e-01   1.085 0.278127    
## Muscletendernessonset -5.669e-01  8.100e-01  -0.700 0.484014    
## Feverad                -7.856e-01  8.824e-01  -0.890 0.373324    
## Headachead              5.982e-02  7.053e-01   0.085 0.932407    
## Chillsad               -1.646e-01  9.304e-01  -0.177 0.859603    
## Rigorsad                3.573e-01  9.008e-01   0.397 0.691655    
## Musclepainad           -3.649e-01  9.199e-01  -0.397 0.691584    
## Muscletendernessad      1.707e-01  8.731e-01   0.195 0.845014    
## Nauseaad                9.579e-01  7.827e-01   1.224 0.220994    
## Vomitingadmission       1.024e-01  5.792e-01   0.177 0.859710    
## Cnsuffusionad           4.820e-01  8.472e-01   0.569 0.569377    
## Prostrationad          -1.884e-01  7.220e-01  -0.261 0.794119    
## Diarrhoeaad            -3.668e-01  7.706e-01  -0.476 0.634054    
## Jaundicead             -4.600e-01  8.843e-01  -0.520 0.602954    
## Hepatictendernessad     1.798e-01  5.654e-01   0.318 0.750489    
## Photophobiaad           2.674e+00  1.234e+00   2.167 0.030254 *  
## Neckstiffnessad        -2.731e-01  7.282e-01  -0.375 0.707668    
## Coughad                -1.199e+00  9.047e-01  -1.326 0.184988    
## SOBadd                 -4.778e-01  1.025e+00  -0.466 0.641259    
## Chestpainad            -5.308e-01  1.020e+00  -0.521 0.602641    
## Bleedingad              3.486e-01  4.234e-01   0.823 0.410343    
## Headache2               2.061e+00  1.181e+00   1.746 0.080874 .  
## Headache3              -3.343e+00  1.159e+00  -2.885 0.003919 ** 
## Headache4               1.578e+00  8.418e-01   1.874 0.060938 .  
## Headache5              -6.104e-01  7.183e-01  -0.850 0.395427    
## Fever2                 -1.418e+00  8.915e-01  -1.590 0.111792    
## Fever3                  9.140e-01  1.067e+00   0.857 0.391601    
## Fever4                 -6.914e-02  1.001e+00  -0.069 0.944938    
## Fever5                 -4.972e-01  8.195e-01  -0.607 0.544054    
## Chills2                 5.142e-01  1.188e+00   0.433 0.664992    
## Chills3                -2.695e+00  1.380e+00  -1.953 0.050872 .  
## Chills4                 1.339e+00  1.275e+00   1.050 0.293697    
## Chills5                 1.674e+00  9.911e-01   1.689 0.091178 .  
## Rigors2                -6.758e-01  1.096e+00  -0.617 0.537531    
## Rigors3                 1.488e+00  1.353e+00   1.100 0.271541    
## Rigors4                -9.301e-01  1.169e+00  -0.796 0.426082    
## Rigors5                -3.189e-01  8.845e-01  -0.361 0.718432    
## Musclepain2             1.103e-02  1.224e+00   0.009 0.992810    
## Musclepain3             1.509e+00  1.276e+00   1.183 0.236855    
## Musclepain4            -1.513e-01  1.225e+00  -0.124 0.901679    
## Musclepain5             2.659e-01  1.025e+00   0.259 0.795282    
## Mustender4              2.784e-01  9.823e-01   0.283 0.776828    
## Mustender5             -1.041e+00  9.796e-01  -1.062 0.288086    
## Nausea2                -1.897e+00  1.097e+00  -1.728 0.083946 .  
## Nausea3                 1.852e+00  1.117e+00   1.658 0.097250 .  
## Nausea4                 8.659e-02  9.900e-01   0.087 0.930302    
## Nausea5                -1.786e+00  8.503e-01  -2.100 0.035716 *  
## Vomiting2               1.840e+00  7.569e-01   2.431 0.015054 *  
## Vomiting3              -1.095e+00  7.877e-01  -1.390 0.164602    
## Vomiting4              -8.425e-01  7.754e-01  -1.087 0.277243    
## Vomiting5               7.642e-01  7.088e-01   1.078 0.280956    
## Consuf4                -1.242e+00  9.512e-01  -1.305 0.191785    
## Consuf5                1.101e+00  9.334e-01   1.180 0.238045
## Prostration4          -4.251e-02  8.737e-01  -0.049 0.961197
## Prostration5          -7.160e-01  8.235e-01  -0.870 0.384571
## diarrhea4              4.567e-01  8.906e-01   0.513 0.608072
## diarrhea5              6.537e-01  9.020e-01   0.725 0.468604
## Jaundice4              3.160e-01  7.166e-01   0.441 0.659249
## Jaundice5             -8.239e-01  7.608e-01  -1.083 0.278820
## hepatictender2        -1.095e+00  5.919e-01  -1.850 0.064327 .
## hepatictender3         1.238e+00  6.921e-01   1.789 0.073624 .
## hepatictender4        -1.354e+00  7.356e-01  -1.840 0.065717 .
## hepatictender5         6.627e-01  6.471e-01   1.024 0.305733
## Photophobia4           1.968e+00  1.126e+00   1.748 0.080424 .
## Photophobia5          -1.689e+00  1.045e+00  -1.617 0.105939
## Neckstiffness2        -3.635e-01  7.411e-01  -0.490 0.623781
## Neckstiffness3        -9.630e-02  8.940e-01  -0.108 0.914220
## Neckstiffness4        -2.365e-01  9.968e-01  -0.237 0.812426
## Neckstiffness5        -1.682e-01  8.662e-01  -0.194 0.846034
## Cough4                -1.190e+00  1.018e+00  -1.169 0.242233
## Cough5                 2.577e+00  9.666e-01   2.667 0.007663 **
## SOB4                   6.486e-01  7.951e-01   0.816 0.414613
## Chestpain4             1.853e-01  7.676e-01   0.241 0.809264
## Bleeding4             -5.282e-01  6.208e-01  -0.851 0.394855
## Bleeding5              3.816e-01  6.287e-01   0.607 0.543845
## WBCcount               6.210e-02  4.162e-02   1.492 0.135718
## Ncount                -8.979e-02  1.444e-01  -0.622 0.533973
## N                     -2.672e-02  2.760e-01  -0.097 0.922869
## Lcount                 6.191e-02  7.132e-02   0.868 0.385370
## L                     -2.404e-02  1.495e-01  -0.161 0.872303
## Plateletcount         -2.299e-02  6.364e-02  -0.361 0.717902
## PCV                    1.564e-01  1.758e-01   0.890 0.373604
## WBC_first_day         -2.122e-01  1.390e-01  -1.527 0.126714
## WPqPCRDiagnosis       -1.815e+00  1.730e-01 -10.488  < 2e-16 ***
## Isolate               -3.562e-01  5.079e-02  -7.012 2.35e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1406.4  on 1270  degrees of freedom
## AIC: 1640.4
##
## Number of Fisher Scoring iterations: 5
```

```r
# Forward selection
forward_model = step(model, direction = "forward", trace = 0)
summary(forward_model)
```

```
##
## Call:
```

```
## glm(formula = Final ~ Year + Month + Hospital + Sample + ICU +
##     OPD + Sex + Age + Ethnicity + Income + Education + TertiaryEducation +
##     Prophylactics + Pasttreatments + Pastantibiotics + Chronicillness +
##     Possibleexposure + Feveronset + Headacheonset + Musclepainonset +
##     Cnsuffusiononset + Jaundiceonset + SOBonset + Coughonset +
##     Chestpainonset + Nauseaonset + Vomitingonset + Diarrhoeaonset +
##     Prostrationonset + Rigorsonset + Photophobiaonset + Chillsonset +
##     Muscletendernessonset + Feverad + Headachead + Chillsad +
##     Rigorsad + Musclepainad + Muscletendernessad + Nauseaad +
##     Vomitingadmission + Cnsuffusionad + Prostrationad + Diarrhoeaad +
##     Jaundicead + Hepatictendernessad + Photophobiaad + Neckstiffnessad +
##     Coughad + SOBadd + Chestpainad + Bleedingad + Headache2 +
##     Headache3 + Headache4 + Headache5 + Fever2 + Fever3 + Fever4 +
##     Fever5 + Chills2 + Chills3 + Chills4 + Chills5 + Rigors2 +
##     Rigors3 + Rigors4 + Rigors5 + Musclepain2 + Musclepain3 +
##     Musclepain4 + Musclepain5 + Mustender4 + Mustender5 + Nausea2 +
##     Nausea3 + Nausea4 + Nausea5 + Vomiting2 + Vomiting3 + Vomiting4 +
##     Vomiting5 + Consuf4 + Consuf5 + Prostration4 + Prostration5 +
##     diarrhea4 + diarrhea5 + Jaundice4 + Jaundice5 + hepatictender2 +
##     hepatictender3 + hepatictender4 + hepatictender5 + Photophobia4 +
##     Photophobia5 + Neckstiffness2 + Neckstiffness3 + Neckstiffness4 +
##     Neckstiffness5 + Cough4 + Cough5 + SOB4 + Chestpain4 + Bleeding4 +
##     Bleeding5 + WBCcount + Ncount + N + Lcount + L + Plateletcount +
##     PCV + WBC_first_day + WPqPCRDiagnosis + Isolate, family = binomial(lin
k = logit),
##     data = train_data)
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -5.646e+03  2.043e+03  -2.763 0.005719 **
## Year                   7.418e+02  2.685e+02   2.763 0.005728 **
## Month                  1.658e-01  1.421e-01   1.167 0.243361
## Hospital               8.269e-01  2.217e-01   3.730 0.000192 ***
## Sample                -1.325e+00  9.461e-01  -1.401 0.161316
## ICU                    1.268e+00  9.201e-01   1.379 0.168029
## OPD                   -5.424e-01  9.449e-01  -0.574 0.565969
## Sex                   -5.834e-01  4.908e-01  -1.189 0.234569
## Age                    4.509e-01  1.314e-01   3.432 0.000599 ***
## Ethnicity             -5.179e-01  5.198e-01  -0.996 0.319133
## Income                 1.643e-02  2.234e-02   0.736 0.461961
## Education             -1.853e-01  1.488e-01  -1.246 0.212789
## TertiaryEducation      3.407e-01  3.166e-01   1.076 0.281746
## Prophylactics         -8.908e-02  4.519e-01  -0.197 0.843750
## Pasttreatments        -2.606e-01  4.481e-01  -0.582 0.560858
## Pastantibiotics        4.091e-01  3.406e-01   1.201 0.229679
## Chronicillness         6.536e-01  4.047e-01   1.615 0.106293
## Possibleexposure      -1.616e+00  4.278e-01  -3.778 0.000158 ***
## Feveronset             1.672e+00  1.099e+00   1.521 0.128344
## Headacheonset          3.120e-02  7.002e-01   0.045 0.964456
## Musclepainonset       -9.847e-01  9.921e-01  -0.993 0.320948
## Cnsuffusiononset      -6.244e-01  7.930e-01  -0.787 0.431047
## Jaundiceonset          8.189e-01  8.376e-01   0.978 0.328230
## SOBonset               9.988e-01  9.903e-01   1.009 0.313200
## Coughonset             4.053e-01  8.143e-01   0.498 0.618717
## Chestpainonset         2.659e-01  1.010e+00   0.263 0.792299
## Nauseaonset            4.963e-01  8.005e-01   0.620 0.535245
## Vomitingonset         -1.637e+00  6.204e-01  -2.638 0.008340 **
## Diarrhoeaonset        -1.054e+00  7.113e-01  -1.482 0.138248
## Prostrationonset       1.497e+00  7.187e-01   2.083 0.037295 *
## Rigorsonset           -3.767e-01  8.004e-01  -0.471 0.637903
## Photophobiaonset      -1.648e+00  1.131e+00  -1.457 0.145026
## Chillsonset            9.980e-01  9.202e-01   1.085 0.278127
## Muscletendernessonset -5.669e-01  8.100e-01  -0.700 0.484014
## Feverad               -7.856e-01  8.824e-01  -0.890 0.373324
## Headachead             5.982e-02  7.053e-01   0.085 0.932407
## Chillsad              -1.646e-01  9.304e-01  -0.177 0.859603
## Rigorsad               3.573e-01  9.008e-01   0.397 0.691655
## Musclepainad          -3.649e-01  9.199e-01  -0.397 0.691584
## Muscletendernessad     1.707e-01  8.731e-01   0.195 0.845014
## Nauseaad               9.579e-01  7.827e-01   1.224 0.220994
## Vomitingadmission      1.024e-01  5.792e-01   0.177 0.859710
## Cnsuffusionad          4.820e-01  8.472e-01   0.569 0.569377
## Prostrationad         -1.884e-01  7.220e-01  -0.261 0.794119
## Diarrhoeaad           -3.668e-01  7.706e-01  -0.476 0.634054
## Jaundicead            -4.600e-01  8.843e-01  -0.520 0.602954
## Hepatictendernessad    1.798e-01  5.654e-01   0.318 0.750489
## Photophobiaad          2.674e+00  1.234e+00   2.167 0.030254 *
## Neckstiffnessad       -2.731e-01  7.282e-01  -0.375 0.707668
## Coughad               -1.199e+00  9.047e-01  -1.326 0.184988
## SOBadd                -4.778e-01  1.025e+00  -0.466 0.641259
## Chestpainad           -5.308e-01  1.020e+00  -0.521 0.602641
## Bleedingad             3.486e-01  4.234e-01   0.823 0.410343
## Headache2              2.061e+00  1.181e+00   1.746 0.080874 .
## Headache3             -3.343e+00  1.159e+00  -2.885 0.003919 **
## Headache4              1.578e+00  8.418e-01   1.874 0.060938 .
## Headache5             -6.104e-01  7.183e-01  -0.850 0.395427
## Fever2                -1.418e+00  8.915e-01  -1.590 0.111792
## Fever3                 9.140e-01  1.067e+00   0.857 0.391601
## Fever4                -6.914e-02  1.001e+00  -0.069 0.944938
## Fever5                -4.972e-01  8.195e-01  -0.607 0.544054
## Chills2                5.142e-01  1.188e+00   0.433 0.664992
## Chills3               -2.695e+00  1.380e+00  -1.953 0.050872 .
## Chills4                1.339e+00  1.275e+00   1.050 0.293697
## Chills5                1.674e+00  9.911e-01   1.689 0.091178 .
## Rigors2               -6.758e-01  1.096e+00  -0.617 0.537531
## Rigors3                1.488e+00  1.353e+00   1.100 0.271541
## Rigors4               -9.301e-01  1.169e+00  -0.796 0.426082
## Rigors5               -3.189e-01  8.845e-01  -0.361 0.718432
## Musclepain2            1.103e-02  1.224e+00   0.009 0.992810
## Musclepain3            1.509e+00  1.276e+00   1.183 0.236855
## Musclepain4          -1.513e-01  1.225e+00  -0.124 0.901679    
## Musclepain5           2.659e-01  1.025e+00   0.259 0.795282    
## Mustender4            2.784e-01  9.823e-01   0.283 0.776828    
## Mustender5           -1.041e+00  9.796e-01  -1.062 0.288086    
## Nausea2              -1.897e+00  1.097e+00  -1.728 0.083946 .  
## Nausea3               1.852e+00  1.117e+00   1.658 0.097250 .  
## Nausea4               8.659e-02  9.900e-01   0.087 0.930302    
## Nausea5              -1.786e+00  8.503e-01  -2.100 0.035716 *  
## Vomiting2             1.840e+00  7.569e-01   2.431 0.015054 *  
## Vomiting3            -1.095e+00  7.877e-01  -1.390 0.164602    
## Vomiting4            -8.425e-01  7.754e-01  -1.087 0.277243    
## Vomiting5             7.642e-01  7.088e-01   1.078 0.280956    
## Consuf4              -1.242e+00  9.512e-01  -1.305 0.191785    
## Consuf5               1.101e+00  9.334e-01   1.180 0.238045    
## Prostration4         -4.251e-02  8.737e-01  -0.049 0.961197    
## Prostration5         -7.160e-01  8.235e-01  -0.870 0.384571    
## diarrhea4             4.567e-01  8.906e-01   0.513 0.608072    
## diarrhea5             6.537e-01  9.020e-01   0.725 0.468604    
## Jaundice4             3.160e-01  7.166e-01   0.441 0.659249    
## Jaundice5            -8.239e-01  7.608e-01  -1.083 0.278820    
## hepatictender2       -1.095e+00  5.919e-01  -1.850 0.064327 .  
## hepatictender3        1.238e+00  6.921e-01   1.789 0.073624 .  
## hepatictender4       -1.354e+00  7.356e-01  -1.840 0.065717 .  
## hepatictender5        6.627e-01  6.471e-01   1.024 0.305733    
## Photophobia4          1.968e+00  1.126e+00   1.748 0.080424 .  
## Photophobia5         -1.689e+00  1.045e+00  -1.617 0.105939    
## Neckstiffness2       -3.635e-01  7.411e-01  -0.490 0.623781    
## Neckstiffness3       -9.630e-02  8.940e-01  -0.108 0.914220    
## Neckstiffness4       -2.365e-01  9.968e-01  -0.237 0.812426    
## Neckstiffness5       -1.682e-01  8.662e-01  -0.194 0.846034    
## Cough4               -1.190e+00  1.018e+00  -1.169 0.242233    
## Cough5                2.577e+00  9.666e-01   2.667 0.007663 ** 
## SOB4                  6.486e-01  7.951e-01   0.816 0.414613    
## Chestpain4            1.853e-01  7.676e-01   0.241 0.809264    
## Bleeding4            -5.282e-01  6.208e-01  -0.851 0.394855    
## Bleeding5             3.816e-01  6.287e-01   0.607 0.543845    
## WBCcount              6.210e-02  4.162e-02   1.492 0.135718    
## Ncount               -8.979e-02  1.444e-01  -0.622 0.533973    
## N                    -2.672e-02  2.760e-01  -0.097 0.922869    
## Lcount                6.191e-02  7.132e-02   0.868 0.385370    
## L                    -2.404e-02  1.495e-01  -0.161 0.872303    
## Plateletcount        -2.299e-02  6.364e-02  -0.361 0.717902    
## PCV                   1.564e-01  1.758e-01   0.890 0.373604    
## WBC_first_day        -2.122e-01  1.390e-01  -1.527 0.126714    
## WPqPCRDiagnosis      -1.815e+00  1.730e-01 -10.488  < 2e-16 ***
## Isolate              -3.562e-01  5.079e-02  -7.012 2.35e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1406.4  on 1270  degrees of freedom
## AIC: 1640.4
##
## Number of Fisher Scoring iterations: 5
```

```
# Backward selection
backward_model = step(model, direction = "backward", trace = 0)
summary(backward_model)
```

```
##
## Call:
## glm(formula = Final ~ Year + Hospital + Sample + ICU + Sex +
##     Age + Chronicillness + Possibleexposure + Feveronset + Musclepainonset
+
##     SOBonset + Vomitingonset + Prostrationonset + Photophobiaonset +
##     Feverad + Nauseaad + Photophobiaad + Coughad + Headache2 +
##     Headache3 + Headache4 + Headache5 + Fever2 + Chills5 + Musclepain3 +
##     Mustender5 + Nausea5 + Vomiting2 + Vomiting4 + Vomiting5 +
##     Prostration5 + hepatictender2 + hepatictender3 + hepatictender4 +
##     hepatictender5 + Photophobia4 + Photophobia5 + Neckstiffness2 +
##     Cough5 + WBCcount + Ncount + PCV + WBC_first_day + WPqPCRDiagnosis +
##     Isolate, family = binomial(link = logit), data = train_data)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -4.073e+03  1.709e+03  -2.384 0.017126 *
## Year              5.353e+02  2.246e+02   2.384 0.017145 *
## Hospital          7.434e-01  2.037e-01   3.650 0.000262 ***
## Sample           -1.420e+00  8.650e-01  -1.641 0.100785
## ICU               6.153e-01  4.089e-01   1.505 0.132388
## Sex              -8.439e-01  4.324e-01  -1.952 0.050979 .
## Age               4.755e-01  1.139e-01   4.174 3.00e-05 ***
## Chronicillness    6.765e-01  3.267e-01   2.071 0.038396 *
## Possibleexposure -1.697e+00  3.779e-01  -4.490 7.13e-06 ***
## Feveronset        2.331e+00  8.509e-01   2.740 0.006150 **
## Musclepainonset  -1.290e+00  6.225e-01  -2.072 0.038256 *
## SOBonset          8.690e-01  5.437e-01   1.598 0.109966
## Vomitingonset    -1.452e+00  4.648e-01  -3.124 0.001783 **
## Prostrationonset  1.227e+00  4.984e-01   2.462 0.013822 *
## Photophobiaonset -1.221e+00  7.419e-01  -1.646 0.099739 .
## Feverad          -9.426e-01  6.262e-01  -1.505 0.132264
## Nauseaad          9.502e-01  5.050e-01   1.882 0.059895 .
## Photophobiaad     2.003e+00  7.136e-01   2.806 0.005012 **
## Coughad          -1.545e+00  5.567e-01  -2.775 0.005525 **
## Headache2         1.282e+00  7.618e-01   1.683 0.092404 .
## Headache3        -2.663e+00  8.180e-01  -3.256 0.001129 **
## Headache4         1.766e+00  6.403e-01   2.758 0.005813 **
## Headache5        -8.788e-01  5.949e-01  -1.477 0.139664
## Fever2           -1.322e+00  5.435e-01  -2.433 0.014983 *  
## Chills5           1.231e+00  4.547e-01   2.707 0.006788 ** 
## Musclepain3       1.214e+00  5.671e-01   2.141 0.032251 *  
## Mustender5       -9.166e-01  4.681e-01  -1.958 0.050210 .  
## Nausea5          -1.074e+00  5.458e-01  -1.968 0.049016 *  
## Vomiting2         8.020e-01  4.394e-01   1.825 0.067973 .  
## Vomiting4        -1.080e+00  5.756e-01  -1.876 0.060680 .  
## Vomiting5         9.143e-01  5.793e-01   1.578 0.114485    
## Prostration5     -7.556e-01  5.072e-01  -1.490 0.136344    
## hepatictender2   -7.633e-01  4.717e-01  -1.618 0.105606    
## hepatictender3    1.010e+00  5.715e-01   1.768 0.077019 .  
## hepatictender4   -1.609e+00  5.745e-01  -2.801 0.005087 ** 
## hepatictender5    7.851e-01  5.020e-01   1.564 0.117844    
## Photophobia4      1.447e+00  7.057e-01   2.050 0.040351 *  
## Photophobia5     -1.080e+00  7.473e-01  -1.445 0.148349    
## Neckstiffness2   -8.588e-01  3.255e-01  -2.638 0.008328 ** 
## Cough5            1.984e+00  5.075e-01   3.909 9.27e-05 ***
## WBCcount          7.050e-02  3.786e-02   1.862 0.062541 .  
## Ncount           -1.093e-01  5.295e-02  -2.064 0.039036 *  
## PCV               1.675e-01  1.191e-01   1.406 0.159584    
## WBC_first_day    -2.215e-01  1.298e-01  -1.707 0.087796 .  
## WPqPCRDiagnosis  -1.750e+00  1.642e-01 -10.657  < 2e-16 ***
## Isolate          -3.602e-01  4.874e-02  -7.390 1.46e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1434.4  on 1341  degrees of freedom
## AIC: 1526.4
## 
## Number of Fisher Scoring iterations: 5
```

```
# backward model has low AIC value 1500.
# so, best model backward model
final_model = backward_model
summary(final_model)
```

```
## 
## Call:
## glm(formula = Final ~ Year + Hospital + Sample + ICU + Sex + 
##     Age + Chronicillness + Possibleexposure + Feveronset + Musclepainonset 
+ 
##     SOBonset + Vomitingonset + Prostrationonset + Photophobiaonset + 
##     Feverad + Nauseaad + Photophobiaad + Coughad + Headache2 + 
##     Headache3 + Headache4 + Headache5 + Fever2 + Chills5 + Musclepain3 + 
##     Mustender5 + Nausea5 + Vomiting2 + Vomiting4 + Vomiting5 + 
##     Prostration5 + hepatictender2 + hepatictender3 + hepatictender4 + 
##     hepatictender5 + Photophobia4 + Photophobia5 + Neckstiffness2 + 
##     Cough5 + WBCcount + Ncount + PCV + WBC_first_day + WPqPCRDiagnosis + 
##     Isolate, family = binomial(link = logit), data = train_data)
## 
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)    
## (Intercept)      -4.073e+03  1.709e+03  -2.384 0.017126 *  
## Year              5.353e+02  2.246e+02   2.384 0.017145 *  
## Hospital          7.434e-01  2.037e-01   3.650 0.000262 ***
## Sample           -1.420e+00  8.650e-01  -1.641 0.100785    
## ICU               6.153e-01  4.089e-01   1.505 0.132388    
## Sex              -8.439e-01  4.324e-01  -1.952 0.050979 .  
## Age               4.755e-01  1.139e-01   4.174 3.00e-05 ***
## Chronicillness    6.765e-01  3.267e-01   2.071 0.038396 *  
## Possibleexposure -1.697e+00  3.779e-01  -4.490 7.13e-06 ***
## Feveronset        2.331e+00  8.509e-01   2.740 0.006150 ** 
## Musclepainonset  -1.290e+00  6.225e-01  -2.072 0.038256 *  
## SOBonset          8.690e-01  5.437e-01   1.598 0.109966    
## Vomitingonset    -1.452e+00  4.648e-01  -3.124 0.001783 ** 
## Prostrationonset  1.227e+00  4.984e-01   2.462 0.013822 *  
## Photophobiaonset -1.221e+00  7.419e-01  -1.646 0.099739 .  
## Feverad          -9.426e-01  6.262e-01  -1.505 0.132264    
## Nauseaad          9.502e-01  5.050e-01   1.882 0.059895 .  
## Photophobiaad     2.003e+00  7.136e-01   2.806 0.005012 ** 
## Coughad          -1.545e+00  5.567e-01  -2.775 0.005525 ** 
## Headache2         1.282e+00  7.618e-01   1.683 0.092404 .  
## Headache3        -2.663e+00  8.180e-01  -3.256 0.001129 ** 
## Headache4         1.766e+00  6.403e-01   2.758 0.005813 ** 
## Headache5        -8.788e-01  5.949e-01  -1.477 0.139664    
## Fever2           -1.322e+00  5.435e-01  -2.433 0.014983 *  
## Chills5           1.231e+00  4.547e-01   2.707 0.006788 ** 
## Musclepain3       1.214e+00  5.671e-01   2.141 0.032251 *  
## Mustender5       -9.166e-01  4.681e-01  -1.958 0.050210 .  
## Nausea5          -1.074e+00  5.458e-01  -1.968 0.049016 *  
## Vomiting2         8.020e-01  4.394e-01   1.825 0.067973 .  
## Vomiting4        -1.080e+00  5.756e-01  -1.876 0.060680 .  
## Vomiting5         9.143e-01  5.793e-01   1.578 0.114485    
## Prostration5     -7.556e-01  5.072e-01  -1.490 0.136344    
## hepatictender2   -7.633e-01  4.717e-01  -1.618 0.105606    
## hepatictender3    1.010e+00  5.715e-01   1.768 0.077019 .  
## hepatictender4   -1.609e+00  5.745e-01  -2.801 0.005087 ** 
## hepatictender5    7.851e-01  5.020e-01   1.564 0.117844    
## Photophobia4      1.447e+00  7.057e-01   2.050 0.040351 *  
## Photophobia5     -1.080e+00  7.473e-01  -1.445 0.148349    
## Neckstiffness2   -8.588e-01  3.255e-01  -2.638 0.008328 ** 
## Cough5            1.984e+00  5.075e-01   3.909 9.27e-05 ***
## WBCcount          7.050e-02  3.786e-02   1.862 0.062541 .  
## Ncount           -1.093e-01  5.295e-02  -2.064 0.039036 *  
## PCV               1.675e-01  1.191e-01   1.406 0.159584    
## WBC_first_day    -2.215e-01  1.298e-01  -1.707 0.087796 .  
## WPqPCRDiagnosis  -1.750e+00  1.642e-01 -10.657  < 2e-16 ***
## Isolate          -3.602e-01  4.874e-02  -7.390 1.46e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1434.4  on 1341  degrees of freedom
## AIC: 1526.4
##
## Number of Fisher Scoring iterations: 5
```

```
# build full logistic model
model_nc = glm(Final ~ ., data = train_non_clinical,
               family=binomial(link=logit))
summary(model_nc)
```

```
##
## Call:
## glm(formula = Final ~ ., family = binomial(link = logit), data = train_non
_clinical)
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -6.919e+03  1.332e+03  -5.195 2.05e-07 ***
## Year               9.090e+02  1.750e+02   5.193 2.07e-07 ***
## Month              9.575e-02  1.160e-01   0.825 0.409305
## Hospital           7.986e-01  1.815e-01   4.401 1.08e-05 ***
## Sample            -9.775e-01  8.151e-01  -1.199 0.230485
## ICU               -1.506e+00  5.961e-01  -2.527 0.011501 *
## OPD                8.578e-01  6.108e-01   1.405 0.160159
## Sex               -6.512e-01  4.113e-01  -1.583 0.113328
## Age                3.214e-01  1.087e-01   2.955 0.003124 **
## Ethnicity         -5.401e-01  4.160e-01  -1.298 0.194241
## Income             1.899e-02  1.870e-02   1.016 0.309766
## Education         -1.998e-01  1.242e-01  -1.609 0.107603
## TertiaryEducation  5.764e-01  2.638e-01   2.185 0.028876 *
## Prophylactics     -1.795e-01  3.543e-01  -0.507 0.612406
## Pasttreatments     2.089e-02  3.633e-01   0.057 0.954148
## Pastantibiotics    1.215e-01  2.782e-01   0.437 0.662254
## Chronicillness     6.884e-01  3.231e-01   2.131 0.033106 *
## Possibleexposure  -1.311e+00  3.594e-01  -3.647 0.000265 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1671.4  on 1369  degrees of freedom
## AIC: 1707.4
##
## Number of Fisher Scoring iterations: 4
```

```
# Forward selection
forward_model_nc = step(model_nc, direction = "forward", trace = 0)
summary(forward_model_nc)
```

```
##
## Call:
## glm(formula = Final ~ Year + Month + Hospital + Sample + ICU +
##     OPD + Sex + Age + Ethnicity + Income + Education + TertiaryEducation +
##     Prophylactics + Pasttreatments + Pastantibiotics + Chronicillness +
##     Possibleexposure, family = binomial(link = logit), data = train_non_cl
inical)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -6.919e+03  1.332e+03  -5.195 2.05e-07 ***
## Year               9.090e+02  1.750e+02   5.193 2.07e-07 ***
## Month              9.575e-02  1.160e-01   0.825 0.409305
## Hospital           7.986e-01  1.815e-01   4.401 1.08e-05 ***
## Sample            -9.775e-01  8.151e-01  -1.199 0.230485
## ICU               -1.506e+00  5.961e-01  -2.527 0.011501 *
## OPD                8.578e-01  6.108e-01   1.405 0.160159
## Sex               -6.512e-01  4.113e-01  -1.583 0.113328
## Age                3.214e-01  1.087e-01   2.955 0.003124 **
## Ethnicity         -5.401e-01  4.160e-01  -1.298 0.194241
## Income             1.899e-02  1.870e-02   1.016 0.309766
## Education         -1.998e-01  1.242e-01  -1.609 0.107603
## TertiaryEducation  5.764e-01  2.638e-01   2.185 0.028876 *
## Prophylactics     -1.795e-01  3.543e-01  -0.507 0.612406
## Pasttreatments     2.089e-02  3.633e-01   0.057 0.954148
## Pastantibiotics    1.215e-01  2.782e-01   0.437 0.662254
## Chronicillness     6.884e-01  3.231e-01   2.131 0.033106 *
## Possibleexposure  -1.311e+00  3.594e-01  -3.647 0.000265 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1671.4  on 1369  degrees of freedom
## AIC: 1707.4
##
## Number of Fisher Scoring iterations: 4
```

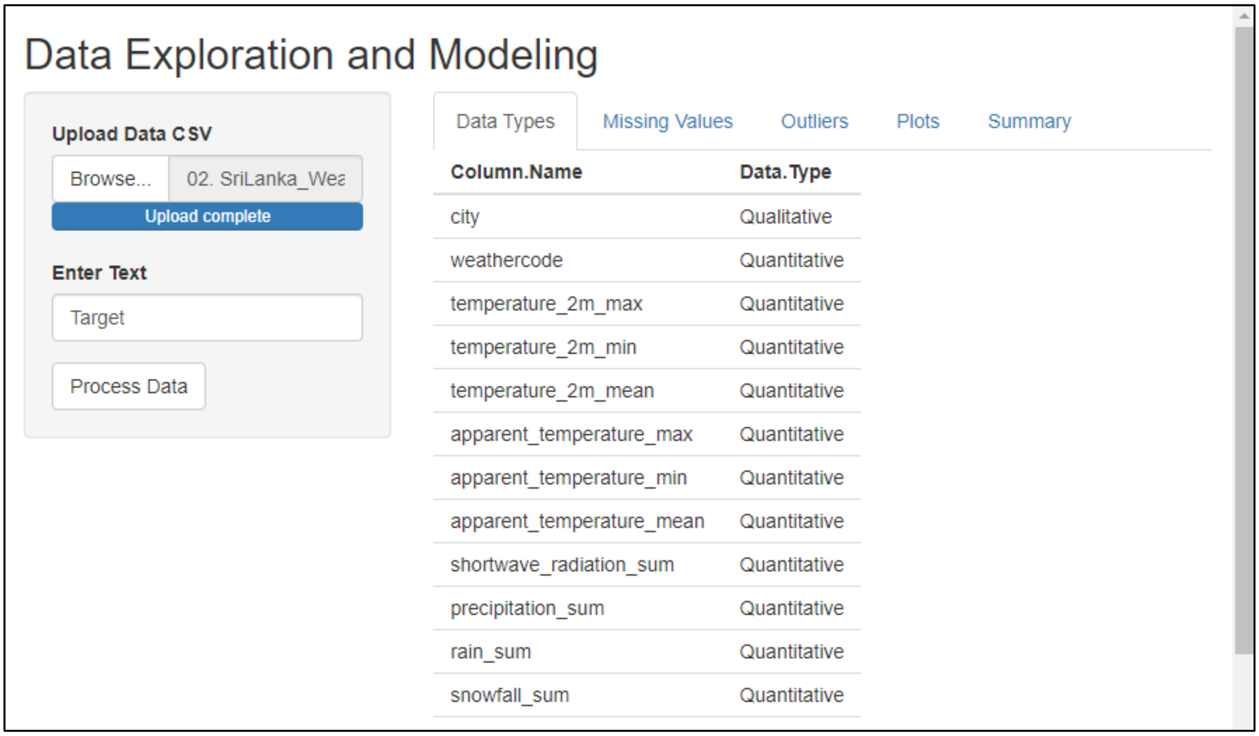
```
# Backward selection
backward_model_nc = step(model_nc, direction = "backward", trace = 0)
summary(backward_model_nc)
```

```
##
## Call:
## glm(formula = Final ~ Year + Hospital + Sample + ICU + Sex +
##     Age + TertiaryEducation + Chronicillness + Possibleexposure,
##     family = binomial(link = logit), data = train_non_clinical)
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -6.969e+03  1.244e+03  -5.604 2.10e-08 ***
## Year               9.157e+02  1.635e+02   5.602 2.12e-08 ***
## Hospital           8.045e-01  1.798e-01   4.475 7.62e-06 ***
## Sample            -1.369e+00  7.284e-01  -1.880 0.060140 .
## ICU               -9.735e-01  3.514e-01  -2.770 0.005598 **
## Sex               -8.884e-01  3.773e-01  -2.355 0.018545 *
## Age                3.575e-01  9.718e-02   3.679 0.000234 ***
## TertiaryEducation  3.271e-01  1.905e-01   1.717 0.085911 .
## Chronicillness     5.970e-01  2.813e-01   2.122 0.033799 *
## Possibleexposure  -1.419e+00  3.382e-01  -4.196 2.71e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1678.5  on 1377  degrees of freedom
## AIC: 1698.5
##
## Number of Fisher Scoring iterations: 4
```

```
# backward model has low AIC value 1687.8
# so, best model backward model
final_model_nc = backward_model_nc
summary(final_model_nc)
```

```
##
## Call:
## glm(formula = Final ~ Year + Hospital + Sample + ICU + Sex +
##     Age + TertiaryEducation + Chronicillness + Possibleexposure,
##     family = binomial(link = logit), data = train_non_clinical)
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -6.969e+03  1.244e+03  -5.604 2.10e-08 ***
## Year               9.157e+02  1.635e+02   5.602 2.12e-08 ***
## Hospital           8.045e-01  1.798e-01   4.475 7.62e-06 ***
## Sample            -1.369e+00  7.284e-01  -1.880 0.060140 .
## ICU               -9.735e-01  3.514e-01  -2.770 0.005598 **
## Sex               -8.884e-01  3.773e-01  -2.355 0.018545 *
## Age                3.575e-01  9.718e-02   3.679 0.000234 ***
## TertiaryEducation  3.271e-01  1.905e-01   1.717 0.085911 .
## Chronicillness     5.970e-01  2.813e-01   2.122 0.033799 *  
## Possibleexposure  -1.419e+00  3.382e-01  -4.196 2.71e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1787.9  on 1386  degrees of freedom
## Residual deviance: 1678.5  on 1377  degrees of freedom
## AIC: 1698.5
##
## Number of Fisher Scoring iterations: 4
```

Appendix of R Shiny.



Data Exploration and Modeling

Upload Data CSV

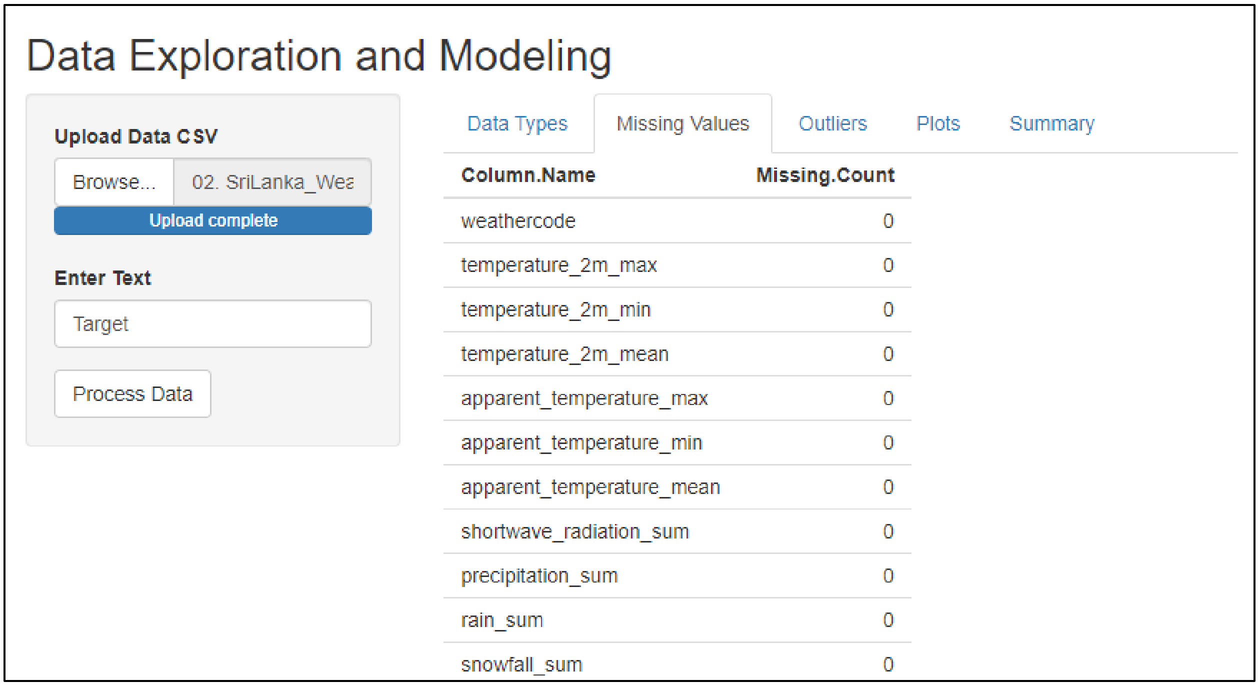
02. SriLanka_Wea

Upload complete

Enter Text

Target

Column.Name	Data.Type
city	Qualitative
weathercode	Quantitative
temperature_2m_max	Quantitative
temperature_2m_min	Quantitative
temperature_2m_mean	Quantitative
apparent_temperature_max	Quantitative
apparent_temperature_min	Quantitative
apparent_temperature_mean	Quantitative
shortwave_radiation_sum	Quantitative
precipitation_sum	Quantitative
rain_sum	Quantitative
snowfall_sum	Quantitative

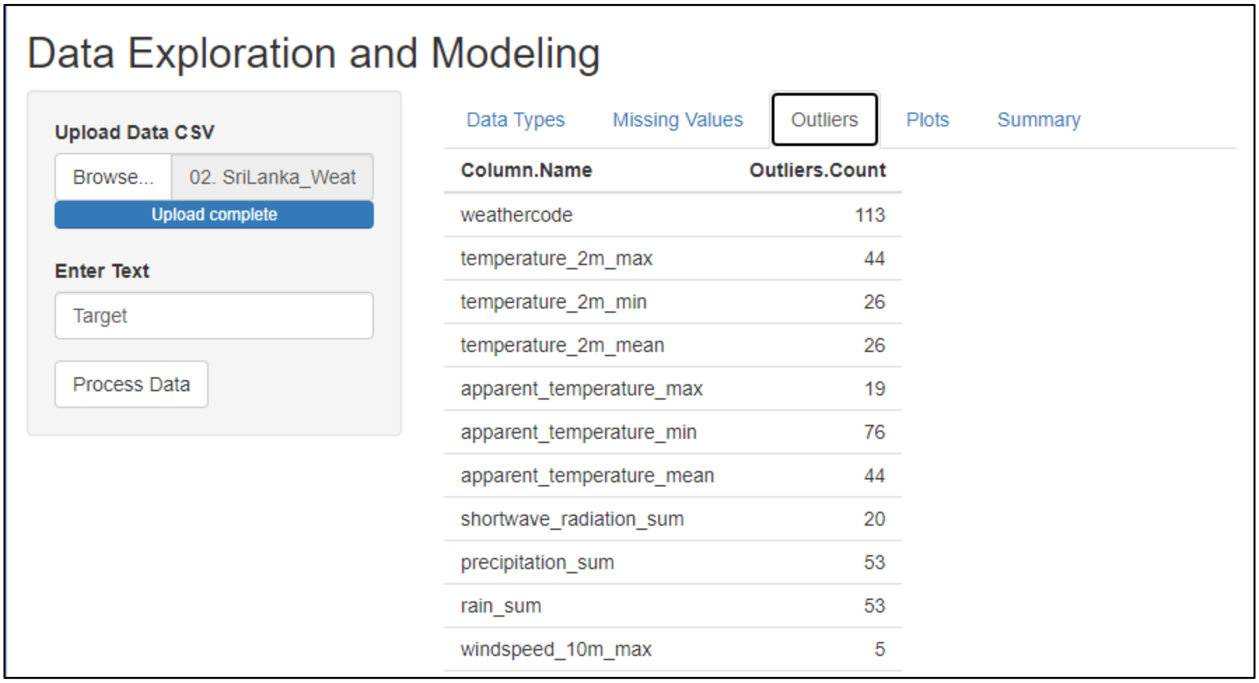


Data Exploration and Modeling

Upload Data CSV: 02. SriLanka_Wea

Enter Text: Target

Column.Name	Missing.Count
weathercode	0
temperature_2m_max	0
temperature_2m_min	0
temperature_2m_mean	0
apparent_temperature_max	0
apparent_temperature_min	0
apparent_temperature_mean	0
shortwave_radiation_sum	0
precipitation_sum	0
rain_sum	0
snowfall_sum	0

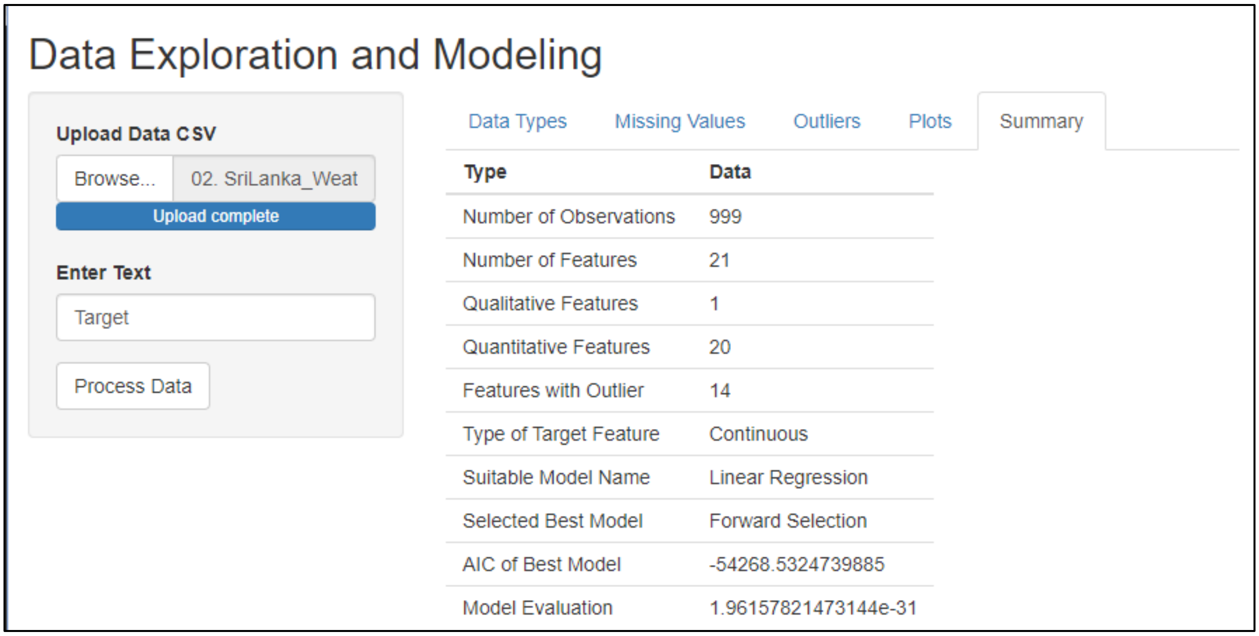


Data Exploration and Modeling

Upload Data CSV: 02. SriLanka_Weat

Enter Text: Target

Column.Name	Outliers.Count
weathercode	113
temperature_2m_max	44
temperature_2m_min	26
temperature_2m_mean	26
apparent_temperature_max	19
apparent_temperature_min	76
apparent_temperature_mean	44
shortwave_radiation_sum	20
precipitation_sum	53
rain_sum	53
windspeed_10m_max	5

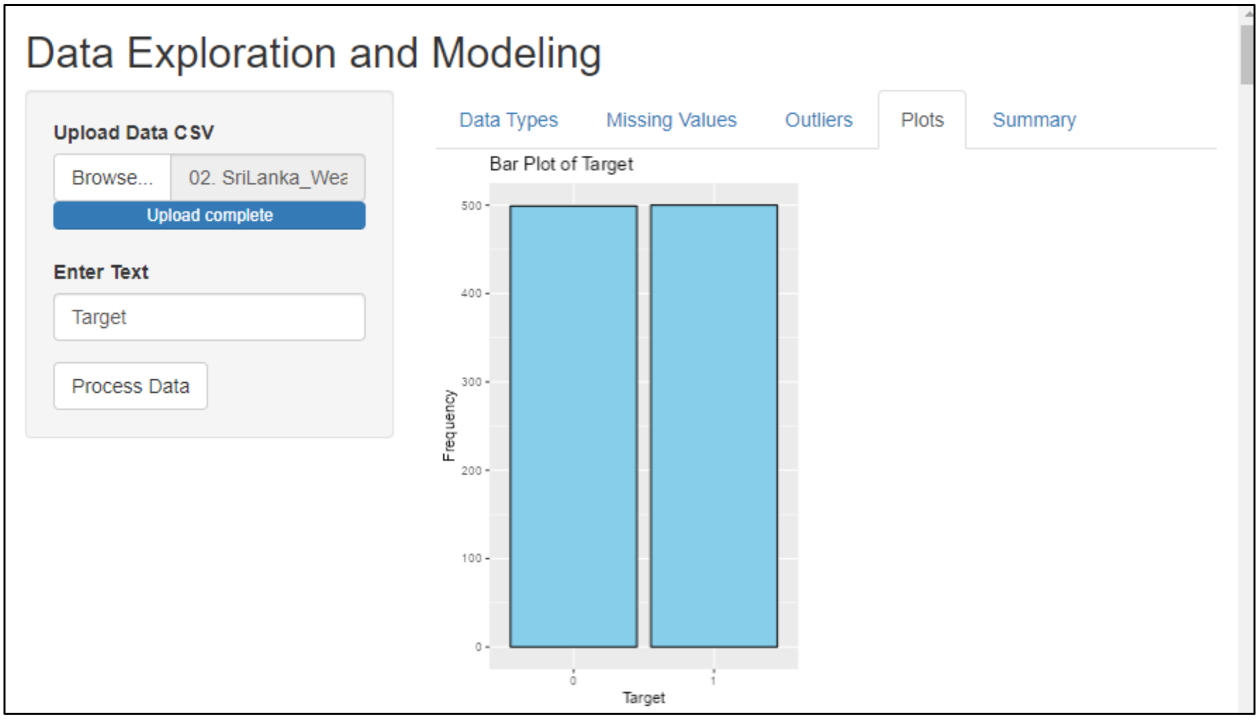


Data Exploration and Modeling

Upload Data CSV: 02. SriLanka_Weat

Enter Text: Target

Type	Data
Number of Observations	999
Number of Features	21
Qualitative Features	1
Quantitative Features	20
Features with Outlier	14
Type of Target Feature	Continuous
Suitable Model Name	Linear Regression
Selected Best Model	Forward Selection
AIC of Best Model	-54268.5324739885
Model Evaluation	1.96157821473144e-31



Data Exploration and Modeling

Upload Data CSV: 02. SriLanka_Wea

Enter Text: Target