Chronic Kidney Disease (CKD) Prediction Project Report

Team Members:

Bhavana Parlupalli
Lavanya Pragada
Pavithra Kadri
Pooja Manikonda
Priyanka Reddy Kuta

Table of Contents:

Abstract:

Chronic Kidney Disease (CKD) represents a significant public health issue, with early detection crucial for effective management. This project utilized logistic regression to analyze a dataset from Kaggle, comprising various clinical and demographic variables from 400 patients, to identify key predictors of CKD. Significant predictors included serum creatinine, hemoglobin, age, hypertension, and diabetes mellitus. The model demonstrated high accuracy and robustness, offering insights into early CKD detection and management. The findings underscore the potential of predictive analytics in improving patient outcomes through early intervention and targeted treatment strategies.

Introduction:

Chronic Kidney Disease (CKD) IS a critical public health concern, affecting millions of people worldwide with increased mortality (Vaidya & Aeddula, 2022). CKD is gradual loss of kidney function over time, which potentially leads to end-stage renal disease (ESRD) which requires dialysis or kidney transplantation for survival. As early stages are often asymptomatic, Early detection and intervention are very crucial for CKD (Chen et al., 2019).

The goal of our project is to use predictive modeling to identify key demographic and clinical factors that influence the development and progression of CKD. By analyzing a dataset that includes various clinical parameters and comorbidities (Chronic Kidney Disease Dataset, 2017), we want to contribute to the early detection and strategic management of CKD, potentially

improving patient outcomes and reducing healthcare costs associated with advanced stages of the disease.

Methodology:

Data Collection

The foundational step in our study involved acquiring a robust dataset suitable for analyzing the progression and determinants of Chronic Kidney Disease (CKD). We sourced our data from Kaggle, which contains records for 400 patients. This dataset is rich in both numerical and categorical variables, providing a comprehensive set of features such as age, blood pressure, serum creatinine, and binary indicators for disease presence (Chronic Kidney Disease Dataset, 2017).

LINK: https://www.kaggle.com/datasets/mansoordaku/ckdisease

Data Preprocessing

- ***Handling Missing Values:***

  Ensuring data quality is crucial in predictive modeling, thus our initial focus was on cleaning. We addressed missing values by imputing the median for continuous variables and the mode for categorical ones. This approach minimizes distortion from outliers and maintains the central tendency of data.
- ***Outlier Identification:***

  We tackled outliers in our data using the Inter-Quartile Range (IQR) method. By setting upper and lower bounds based on 1.5 times the IQR, we identified and capped outliers to prevent bias in statistical analyses and predictive models. This approach maintains dataset integrity by keeping outlier influence within a reasonable range.

Data Transformation

Normalization and Scaling:

We normalized numerical data to ensure effective model integration. This prevents variables with larger ranges from overshadowing others, particularly important in logistic regression.

Encoding Categorical Variables:

We converted categorical variables into numerical codes using label encoding for logistic regression compatibility. This allowed essential predictors like hypertension, diabetes mellitus, and coronary artery disease to be effectively processed alongside continuous variables in predicting CKD.

Exploratory Data Analysis (EDA):

Statistical Tests:

T-tests compared means of continuous variables between CKD and non-CKD groups, identifying significant differences. Chi-square tests assessed categorical data distributions, indicating associations with CKD presence.

Visualization Techniques: We employed a variety of visualization techniques to explore data trends and validate assumptions of our statistical tests.

- Histograms and box plots showcased the distribution and central tendencies of variables such as age and serum creatinine levels, providing visual insights into the data skewness and presence of outliers.
- Correlation heatmaps were used to reveal potential multicollinearity between variables, guiding our model selection and feature engineering steps.

Model Building:

Given the binary nature of our outcome variable (presence vs. absence of CKD), logistic regression was a natural fit for our initial modeling due to its efficiency in handling binary classification problems, its interpretability, and its ability to provide probabilities as outputs, which are useful for medical decision-making.

It was implemented using the glm() function in R, which is designed for fitting generalized linear models. The model included a comprehensive set of variables like the demographic factors such as age and gender, clinical measurements like blood pressure, serum creatinine, hemoglobin, and others, comorbid conditions including diabetes mellitus and hypertension.

Model Validation: We rigorously validated our models using several performance metrics.

- ✓ Accuracy provided a basic measure of the overall correctness of the model.
- ✓ Precision, Recall, and F1 Score offered deeper insights into the model's capability to handle imbalanced data, which is often the case in medical datasets where one class (CKD) might be less prevalent.
- ✓ The ROC Curve and Area Under the Curve (AUC) quantified the model's ability to discriminate between the two classes under various threshold settings, essential for clinical decision-making processes where sensitivity and specificity are critical.

Results:

The logistic regression model employed in this dataset aimed to discern the impact of various clinical and demographic variables on the likelihood of developing CKD. The results of this analysis provided insightful outcomes.

❖ ***Model Performance and Statistical Significance:*** The logistic regression outputs indicated that several variables were statistically significant predictors of CKD.

- ***Serum Creatinine:*** This variable had a positive coefficient, suggesting a strong association with the likelihood of CKD. As serum creatinine levels increased, so did the probability of CKD.
- ***Hemoglobin:*** This variable showed a negative coefficient, indicating that higher hemoglobin levels were associated with a lower probability of CKD.
- ***Age:*** Age was another significant predictor, with older individuals having a higher risk of developing CKD.
- ***Hypertension and Diabetes Mellitus:*** Both conditions were positively correlated with CKD,

The model's overall accuracy and other performance metrics such as precision, recall, F1 score, and the area under the ROC curve (AUC) were commendably high. These metrics indicated excellent model performance, demonstrating the model's ability to correctly classify cases as CKD or non-CKD.

Model Diagnostics: While the model exhibited strong predictive power, initial runs highlighted convergence issues likely due to multicollinearity among predictors.

Model Diagnostics and Output: The output of the logistic regression provided several key pieces of information.

- ***Coefficients:*** The beta coefficients indicate the influence of each variable on the odds of developing CKD. Positive coefficients increase the odds of the outcome (presence of CKD), while negative coefficients decrease it.
- ***Significance Levels (p-values):*** These help determine which variables are statistically significant predictors of CKD.

While the model exhibited strong predictive power, initial runs highlighted convergence issues likely due to multicollinearity among predictors.

Discussion:

The results from the logistic regression model are highly informative in understanding the key factors influencing CKD development. The statistical significance of variables like serum creatinine and hemoglobin is particularly noteworthy, as these clinical measurements are routinely available and monitored in patients suspected of renal impairment. This underscores the potential utility of the model in clinical settings, where such a predictive model could aid in early screening and intervention.

Moreover, the significant associations observed with age, hypertension, and diabetes provide a basis for targeted interventions in these higher-risk groups. For instance, managing

blood pressure and blood sugar levels in patients could be emphasized as part of preventive strategies against CKD. One of the challenges encountered was the model's initial failure to converge, a common issue in logistic regression that often points to deeper data or model specification issues such as multicollinearity. Our approach to mitigating this through variable scaling and selection not only resolved the convergence issues but also enhanced the model's performance by focusing on the most influential predictors.

Conclusion:

The logistic regression model developed as part of this project has demonstrated a strong capacity to predict Chronic Kidney Disease using a set of clinical and demographic variables. The findings highlight significant relationships between CKD and several key predictors, with implications for early diagnosis and management of the disease.

The successful application of logistic regression, despite initial setbacks, showcases the model's robustness and the effectiveness of the corrective measures implemented. These insights not only enhance our understanding of CKD's determinants but also offer a valuable tool for healthcare professionals in predicting and managing this condition.

Future work could expand on this foundation by integrating more nuanced patient data, exploring interactions between variables, or employing more complex machine learning models that might capture non-linear relationships more effectively. Additionally, longitudinal studies could provide insights into how the risk factors impact CKD progression over time, offering a dynamic view of risk that could further refine predictive accuracy.

This project thus serves as a significant step forward in the predictive analytics of CKD, with potential implications for improving patient outcomes through earlier detection and personalized treatment strategies.

```{r}
library(dplyr)
library(tidyverse)
library(ggplot2)
library(tidyr)
```

```{r}
data <- read.csv ("C:\\Users\\Dr Lavanya\\OneDrive\\Desktop\\Lavanya\\assignment\\applied stats\\project\\kidney_disease.csv")
View(data)
data= subset(data, select = -c(X))
data
```

Description: df [400 × 27]

	S.no <int>	Age <int>	Blood.Pressure <int>	Specific.Gravity <dbl>	Albumin <int>	Sugar <int>	Red.Blood.Cells <chr>	Pus.cells <chr>	Pus.cell.clumps <chr>
1	0	48	80	1.020	1	0		normal	notpresent
2	1	7	50	1.020	4	0		normal	notpresent
3	2	62	80	1.010	2	3	normal	normal	notpresent
4	3	48	70	1.005	4	0	normal	abnormal	present
5	4	51	80	1.010	2	0	normal	normal	notpresent
6	5	60	90	1.015	3	0			notpresent
7	6	68	70	1.010	0	0		normal	notpresent
8	7	24	*NA*	1.015	2	4	normal	abnormal	notpresent
9	8	52	100	1.015	3	0	normal	abnormal	present
10	9	53	90	1.020	2	0	abnormal	abnormal	present

1-10 of 400 rows | 1-10 of 27 columns

Previous 1 2 3 4 5 6 ... 40 Next

#To examine the dimensions and the structure of the data

```{r}
#To examine the dimensions and the structure of the data
dim (data)

str (data)
```

```
'data.frame':   400 obs. of  27 variables:
 $ S.no                  : int  0 1 2 3 4 5 6 7 8 9 ...
 $ Age                   : int  48 7 62 48 51 60 68 24 52 53 ...
 $ Blood.Pressure        : int  80 50 80 70 80 90 70 NA 100 90 ...
 $ Specific.Gravity      : num  1.02 1.02 1.01 1 1.01 ...
 $ Albumin               : int  1 4 2 4 2 3 0 2 3 2 ...
 $ Sugar                 : int  0 0 3 0 0 0 0 4 0 0 ...
 $ Red.Blood.Cells       : chr  "" "" "normal" "normal" ...
 $ Pus.cells             : chr  "normal" "normal" "normal" "abnormal" ...
 $ Pus.cell.clumps       : chr  "notpresent" "notpresent" "notpresent" "present" ...
 $ Bacteria              : chr  "notpresent" "notpresent" "notpresent" "notpresent" ...
 $ Blood.glucose.random  : int  121 NA 423 117 106 74 100 410 138 70 ...
 $ Blood.urea            : num  36 18 53 56 26 25 54 31 60 107 ...
 $ Serum.creatinine      : num  1.2 0.8 1.8 3.8 1.4 1.1 24 1.1 1.9 7.2 ...
 $ Sodium                : num  NA NA NA 111 NA 142 104 NA NA 114 ...
 $ Potassium             : num  NA NA NA 2.5 NA 3.2 4 NA NA 3.7 ...
 $ Hemoglobin            : num  15.4 11.3 9.6 11.2 11.6 12.2 12.4 12.4 10.8 9.5 ...
 $ Packed.cell.volume    : int  44 38 31 32 35 39 36 44 33 29 ...
 $ White.blood.cell.count: int  7800 6000 7500 6700 7300 7800 NA 6900 9600 12100 ...
 $ Red.blood.cell.count  : num  5.2 NA NA 3.9 4.6 4.4 NA 5 4 3.7 ...
 $ hypertension          : chr  "yes" "no" "no" "yes" ...
 $ Diabetes.Mellitus     : chr  "yes" "no" "yes" "no" ...
 $ Coronary.artery.disease: chr  "no" "no" "no" "no" ...
 $ Appetite              : chr  "good" "good" "poor" "poor" ...
 $ Pedal.edema           : chr  "no" "no" "no" "yes" ...
 $ Anemia                : chr  "no" "no" "yes" "yes" ...
 $ Classification        : chr  "ckd" "ckd" "ckd" "ckd" ...
 $ X.1                   : chr  "" "" "" "" ...
```

```r
#Checking for null values in the entire data frame
if (any (is.na(data))) {
  print ("There are NA values in the data frame")
} else {
  print ("There are no NA values in the data frame")
}
```

```
[1] "There are NA values in the data frame"
```

```r
#Checking for null values
total_na <- sum (is.na (data))
total_na
```

```
[1] 775
```

```r
any (duplicated (data))
```

```
[1] FALSE
```

```r
############### FILLING "" VALUES IN CATEGORICAL COLUMNS
# Function to calculate mode
get_mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

# Identify categorical columns
cat_columns <- sapply(data, is.character)

# Loop through each categorical column
for (col in names(data)[cat_columns]) {
  # Calculate mode of the column
  mode_val <- get_mode(data[[col]])

  # Replace empty strings with mode value
  data[[col]][data[[col]] == ""] <- mode_val
}


########################### Since column "X.1" has unique values : ["#N/A", "#NAME?", ""], we should drop the column.
########################### Values doesn't convey any information.
data_updated <- subset(data, select = -"$x.1")
view(data)
```

```r
######################################################## UPDATED CODE ###########################################################################
na_count_by_column <- sapply(data, function(x) sum (is.na(x)))
na_count_by_column


#Imputing the missing values in numerical data with median
# Assuming 'data' is your dataframe
data$Blood.Pressure[is.na(data$Blood.Pressure)] <- median(data$Blood.Pressure, na.rm = TRUE)
data$Serum.creatinine[is.na(data$Serum.creatinine)] <- median(data$Serum.creatinine, na.rm = TRUE)
data$Blood.urea[is.na(data$Blood.urea)] <- median(data$Blood.urea, na.rm = TRUE)  # For numerical data
data$Albumin[is.na(data$Albumin)] <- median(data$Albumin, na.rm = TRUE)  # For numerical data
data$Sugar[is.na(data$Sugar)] <- median(data$Sugar, na.rm = TRUE)  # For numerical data
data$Specific.Gravity[is.na(data$Specific.Gravity)] <- median(data$Specific.Gravity, na.rm = TRUE) # For numerical data
data$Blood.glucose.random[is.na(data$Blood.glucose.random)] <- median(data$Blood.glucose.random, na.rm = TRUE)  # For numerical data
data$Sodium[is.na(data$Sodium)] <- median(data$Sodium, na.rm = TRUE)  # For numerical data
data$Potassium[is.na(data$Potassium)] <- median(data$Potassium, na.rm = TRUE)  # For numerical data
data$Hemoglobin[is.na(data$Hemoglobin)] <- median(data$Hemoglobin, na.rm = TRUE)  # For numerical data
data$Packed.cell.volume[is.na(data$Packed.cell.volume)] <- median(data$Packed.cell.volume, na.rm = TRUE)
data$White.blood.cell.count[is.na(data$White.blood.cell.count)] <- median(data$White.blood.cell.count, na.rm = TRUE)
data$Red.blood.cell.count[is.na(data$Red.blood.cell.count)] <- median(data$Red.blood.cell.count, na.rm = TRUE)
view(data)


data$Age[is.na(data$Age)] <- median(data$Age, na.rm = TRUE)
median(data$Age, na.rm = TRUE)


# Replaced the missing values in numerical data with Median


na_count_by_column <- sapply(data, function(x) sum (is.na(x)))
na_count_by_column
view(data)
```

```
                  S.no                    Age         Blood.Pressure        Specific.Gravity                Albumin                  Sugar 
                     0                      0                     0                       0                      0                      0 
       Red.Blood.Cells              Pus.cells        Pus.cell.clumps               Bacteria    Blood.glucose.random             Blood.urea 
                     0                      0                     0                       0                      0                      0 
       Serum.creatinine                 Sodium             Potassium             Hemoglobin      Packed.cell.volume White.blood.cell.count 
                     0                      0                     0                       0                      0                      0 
   Red.blood.cell.count           hypertension      Diabetes.Mellitus Coronary.artery.disease              Appetite            Pedal.edema 
                     0                      0                     0                       0                      0                      0 
                Anemia         Classification                   X.1 
                     0                      0                     0 
[1] 55
                  S.no                    Age         Blood.Pressure        Specific.Gravity                Albumin                  Sugar 
                     0                      0                     0                       0                      0                      0 
       Red.Blood.Cells              Pus.cells        Pus.cell.clumps               Bacteria    Blood.glucose.random             Blood.urea 
                     0                      0                     0                       0                      0                      0 
       Serum.creatinine                 Sodium             Potassium             Hemoglobin      Packed.cell.volume White.blood.cell.count 
                     0                      0                     0                       0                      0                      0 
   Red.blood.cell.count           hypertension      Diabetes.Mellitus Coronary.artery.disease              Appetite            Pedal.edema 
                     0                      0                     0                       0                      0                      0 
                Anemia         Classification                   X.1 
                     0                      0                     0 
```

```{r}
#data_updated = data[, -which(names(data)== "X.1", "X")]
data_updated = data[, !(names(data) %in% c("X.1", "X"))]
view(data_updated)
```

Summary stats

```{r}
summary(data)
summary_data <- do.call(cbind, data_updated)
```

```
      S.no             Age        Blood.Pressure   Specific.Gravity    Albumin         Sugar       Red.Blood.Cells     Pus.cells         Pus.cell.clumps   
 Min.   :  0.00   Min.   : 2.00   Min.   : 50.00   Min.   :1.005    Min.   :0.0   Min.   :0.000   Length:400         Length:400         Length:400        
 1st Qu.: 99.75   1st Qu.:42.00   1st Qu.: 70.00   1st Qu.:1.015    1st Qu.:0.0   1st Qu.:0.000   Class :character   Class :character   Class :character  
 Median :199.50   Median :55.00   Median : 80.00   Median :1.020    Median :0.0   Median :0.000   Mode  :character   Mode  :character   Mode  :character  
 Mean   :199.50   Mean   :51.56   Mean   : 76.58   Mean   :1.018    Mean   :0.9   Mean   :0.395                                                           
 3rd Qu.:299.25   3rd Qu.:64.00   3rd Qu.: 80.00   3rd Qu.:1.020    3rd Qu.:2.0   3rd Qu.:0.000                                                           
 Max.   :399.00   Max.   :90.00   Max.   :180.00   Max.   :1.025    Max.   :5.0   Max.   :5.000                                                           
   Bacteria         Blood.glucose.random   Blood.urea     Serum.creatinine     Sodium         Potassium        Hemoglobin    Packed.cell.volume White.blood.cell.count
 Length:400         Min.   : 22.0        Min.   :  1.50   Min.   : 0.400   Min.   :  4.5   Min.   : 2.500   Min.   : 3.10   Min.   : 9.00      Min.   : 2200      
 Class :character   1st Qu.:101.0        1st Qu.: 27.00   1st Qu.: 0.900   1st Qu.:135.0   1st Qu.: 4.000   1st Qu.:10.88   1st Qu.:34.00      1st Qu.: 6975      
 Mode  :character   Median :121.0        Median : 42.00   Median : 1.300   Median :138.0   Median : 4.400   Median :12.65   Median :40.00      Median : 8000      
                    Mean   :145.1        Mean   : 56.69   Mean   : 2.997   Mean   :137.6   Mean   : 4.577   Mean   :12.54   Mean   :39.11      Mean   : 8303      
                    3rd Qu.:150.0        3rd Qu.: 61.75   3rd Qu.: 2.725   3rd Qu.:141.0   3rd Qu.: 4.800   3rd Qu.:14.62   3rd Qu.:44.00      3rd Qu.: 9400      
                    Max.   :490.0        Max.   :391.00   Max.   :76.000   Max.   :163.0   Max.   :47.000   Max.   :17.80   Max.   :54.00      Max.   :26400      
 Red.blood.cell.count hypertension       Diabetes.Mellitus  Coronary.artery.disease   Appetite         Pedal.edema          Anemia          Classification    
 Min.   :2.100        Length:400         Length:400         Length:400               Length:400         Length:400         Length:400         Length:400        
 1st Qu.:4.500        Class :character   Class :character   Class :character         Class :character   Class :character   Class :character   Class :character  
 Median :4.800        Mode  :character   Mode  :character   Mode  :character         Mode  :character   Mode  :character   Mode  :character   Mode  :character  
 Mean   :4.736                                                                                                                                                    
 3rd Qu.:5.100                                                                                                                                                    
 Max.   :8.000                                                                                                                                                    
     X.1           
 Length:400         
 Class :character   
 Mode  :character   
```

Calculating Range, Skewness

```{r}
# Function to calculate range
range_data_updated <- function(x) {
  if(is.numeric(x)) {
    range_val <- max(x, na.rm = TRUE) - min(x, na.rm = TRUE)
    return(range_val)
  } else {
    return(NA)  # return NA for non-numeric variables
  }
}

# Function to calculate skewness
skewness_data_updated <- function(x) {
  if(is.numeric(x)) {
    skewness_val <- (3 * (mean(x, na.rm = TRUE) - median(x, na.rm = TRUE))) / sd(x, na.rm = TRUE)
    return(skewness_val)
  } else {
    return(NA)  # return NA for non-numeric variables
  }
}

# Applying functions to each variable
result <- data.frame(
  Variable = names(data_updated),
  Range = sapply(data_updated, range_data_updated),
  Skewness = sapply(data_updated, skewness_data_updated)
)
result
```

Description: df [26 × 3]

	Variable <chr>	Range <dbl>	Skewness <dbl>
S.no	S.no	399.00	0.0000000
Age	Age	88.00	-0.6072250
Blood.Pressure	Blood.Pressure	130.00	-0.7616874
Specific.Gravity	Specific.Gravity	0.02	-1.2628752
Albumin	Albumin	5.00	2.0561557
Sugar	Sugar	5.00	1.1393814
Red.Blood.Cells	Red.Blood.Cells	*NA*	*NA*
Pus.cells	Pus.cells	*NA*	*NA*
Pus.cell.clumps	Pus.cell.clumps	*NA*	*NA*
Bacteria	Bacteria	*NA*	*NA*

1-10 of 26 rows

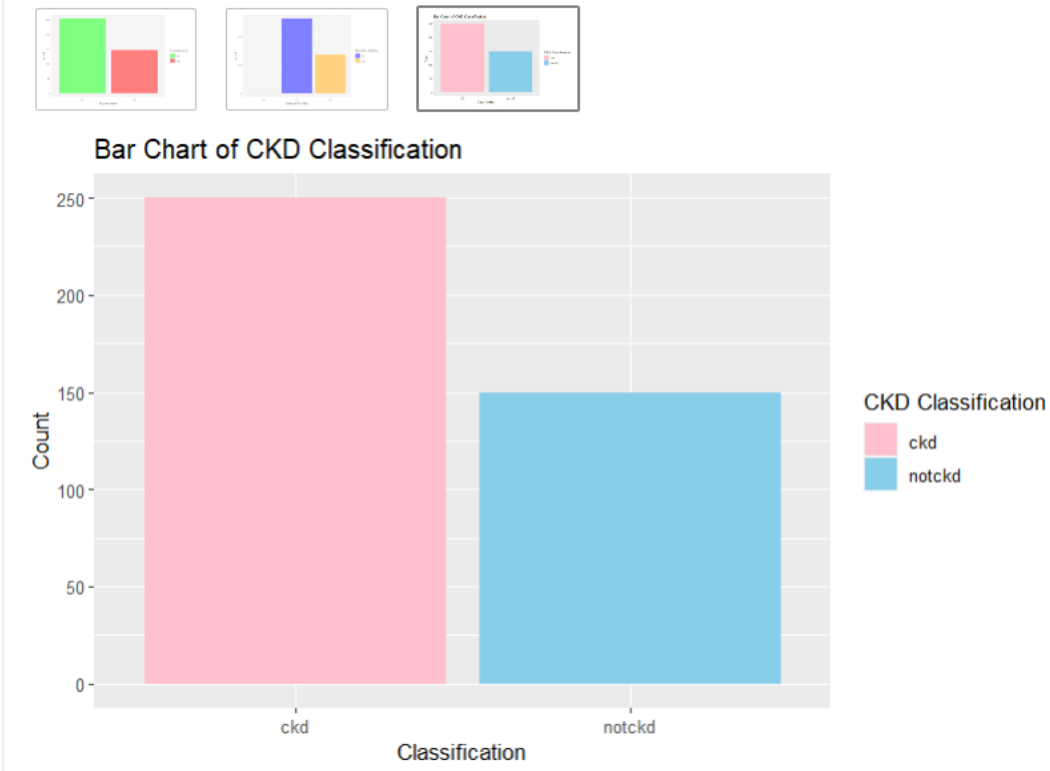
Previous 1 2 3 Next

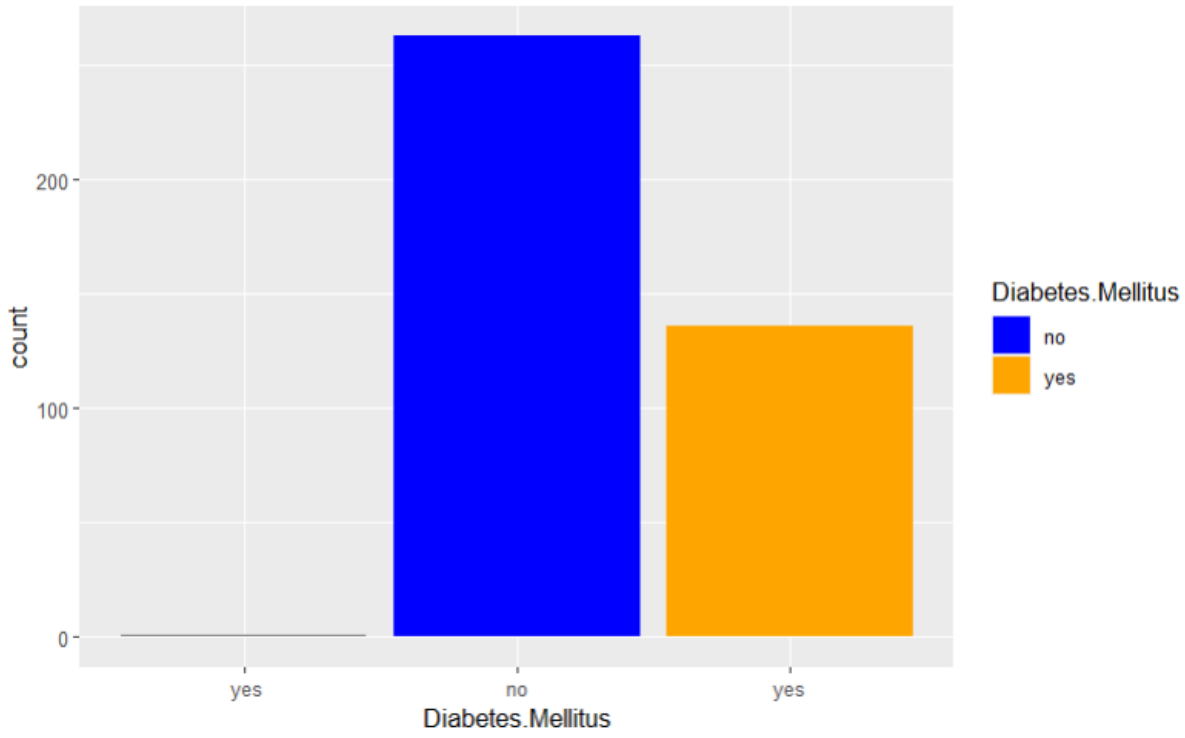
Check outliers and IQR

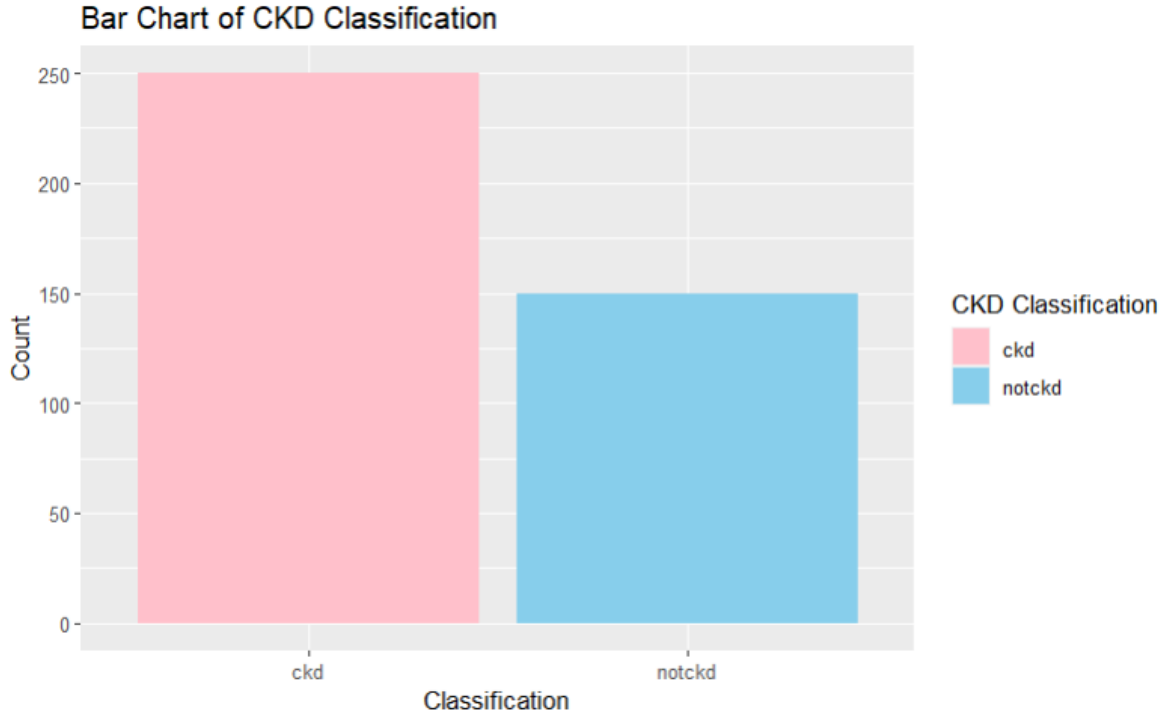
```{r}
# Checking for the presence of outliers
for (col in names(data_updated)[sapply(data_updated, is.numeric)]) {
  print(paste("Column:", col))

  # Calculate IQR
  q1 <- quantile(data_updated[[col]], 0.25, na.rm = TRUE)
  q3 <- quantile(data_updated[[col]], 0.75, na.rm = TRUE)
  iqr <- q3 - q1
  print(paste("IQR:", iqr))

  # Identify outliers
  lower_bound <- q1 - 1.5 * iqr
  upper_bound <- q3 + 1.5 * iqr
  outliers <- data_updated[[col]][data_updated[[col]] < lower_bound | data_updated[[col]] > upper_bound]
  print(paste("Outliers:", length(outliers)))
  print(outliers)

  #print("---")
}
```

Bar plots for categorical variables

```{r}
ggplot(data_updated, aes(x = hypertension, fill = hypertension)) + geom_bar() + scale_fill_manual(values = c("yes" = "red", "no" = "green"))
ggplot(data_updated, aes(x = Diabetes.Mellitus, fill = Diabetes.Mellitus)) + geom_bar() + scale_fill_manual(values = c("yes" = "orange", "no" = "blue"))
# Using scale_fill_manual to specify custom colors
ggplot(data_updated, aes(x = as.factor(Classification), fill = as.factor(Classification))) +
  geom_bar() +
  scale_fill_manual(values = c("ckd" = "pink", "notckd" = "skyblue")) +  # Adjusted color mapping
  labs(
    x = "Classification",
    y = "Count",
    fill = "CKD Classification",
    title = "Bar Chart of CKD Classification"
  )
```


Bar Chart of CKD Classification
250
200
150
100
50
0
Count
ckd
notckd
Classification
CKD Classification
ckd
notckd

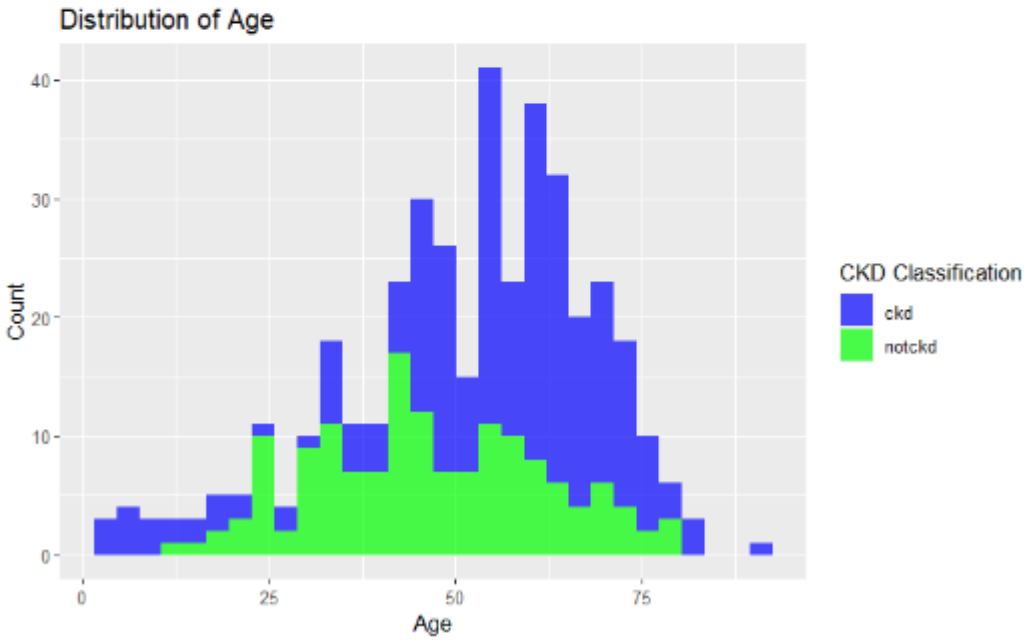

count
200
100
0
yes
no
yes
Diabetes.Mellitus
Diabetes.Mellitus
no
yes


Bar Chart of CKD Classification
Count
250
200
150
100
50
0
ckd
notckd
Classification
CKD Classification
ckd
notckd

Histogram for age vs classification of the disease

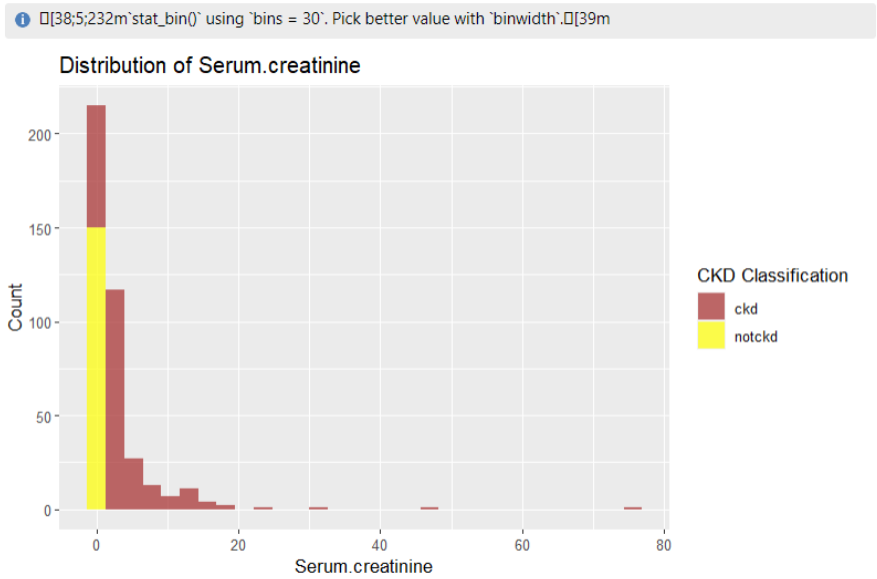
```{r}
ggplot(data_updated, aes(x = Age, fill = as.factor(Classification))) +
  geom_histogram(alpha = 0.7) +
  scale_fill_manual(values = c("ckd" = "blue", "notckd" = "green")) +
  labs(
    title = "Distribution of Age",
    x = "Age",
    y = "Count",
    fill = "CKD Classification"
  )
```

```
□[38;5;232m`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.□[39m
```



Histogram for serum creatinine vs classification of the disease

```{r}
ggplot(data_updated, aes(x = Serum.creatinine, fill = as.factor(Classification))) +
  geom_histogram(alpha = 0.7) +
  scale_fill_manual(values = c("ckd" = "brown", "notckd" = "yellow")) +
  labs(
    title = "Distribution of Serum.creatinine",
    x = "Serum.creatinine",
    y = "Count",
    fill = "CKD Classification"
  )
```

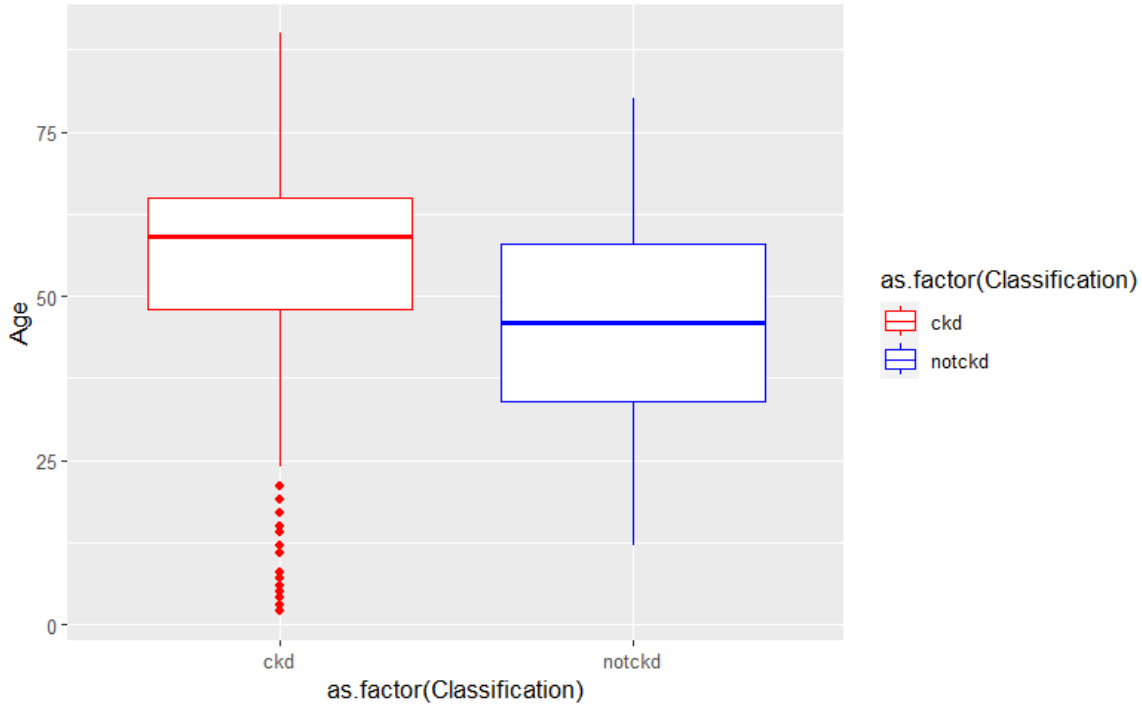


Age vs ckd/notckd - scatter plot

```{r}
ggplot(data_updated, aes(x = Age, y = as.factor(Classification), color = as.factor(Classification))) +
  geom_point() + geom_jitter(width = 0.1, height = 0.1) +
  scale_color_manual(values = c("ckd" = "red", "notckd" = "blue"))
```

Box plots for age vs classification of ckd

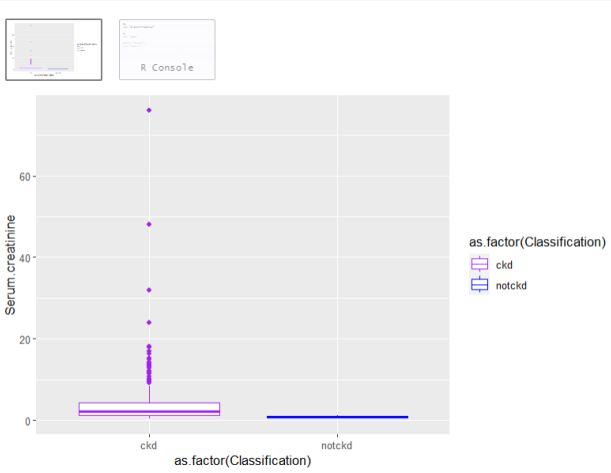
```{r}
ggplot(data_updated, aes(x = as.factor(Classification), y = Age, color = as.factor(Classification))) +
  geom_boxplot() + scale_color_manual(values = c("ckd" = "red", "notckd" = "blue")) +
  labs(x = "Classification", y = "Age")
```



```
$x
[1] "Classification"

$y
[1] "Age"

attr(,"class")
[1] "labels"
```

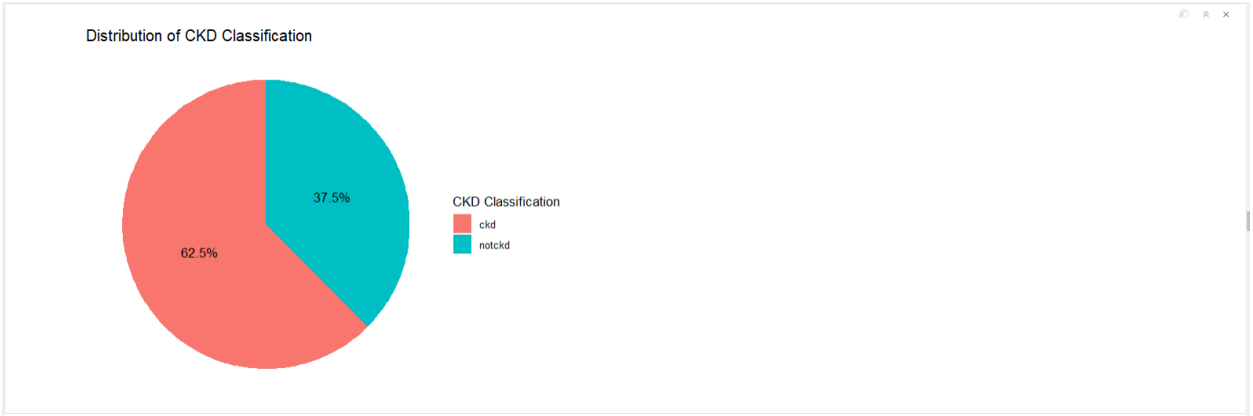
Box plots for Serum.creatinine vs classification of ckd

```r
ggplot(data_updated, aes(x = as.factor(Classification), y = Serum.creatinine, color = as.factor(Classification))) +
  geom_boxplot() + scale_color_manual(values = c("ckd" = "purple", "notckd" = "blue")) +
  labs(x = "Classification", y = "Age")
```



Pie chart showing the distribution of CKD classfication in the data set

```r
data_updated$classification <- as.factor(data_updated$Classification)
ckd_counts <- table(data_updated$Classification)

# Convert the table to a data frame for ggplot and calculate percentages
ckd_counts_df <- as.data.frame(ckd_counts) %>%
  mutate(Percentage = Freq / sum(Freq) * 100)

# Rename columns for clarity
names(ckd_counts_df) <- c("Classification", "Freq", "Percentage")

# Create the pie chart with percentage labels
ggplot(ckd_counts_df, aes(x = "", y = Freq, fill = Classification)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +  # This converts the bar chart to a pie chart
  geom_text(aes(label = sprintf("%.1f%%", Percentage)), position = position_stack(vjust = 0.5)) +
  labs(
    title = "Distribution of CKD Classification",
    x = NULL,
    y = NULL,
    fill = "CKD Classification"
  ) +
  theme_void() +  # This removes the background, gridlines, and axes
  theme(legend.position = "right")
```



Shapiro-Wilk test to check normality

Shapiro-Wilk test to check normality

```{r}
# Convert categorical variables to factor
data_updated <- data_updated %>%
  mutate_if(is.character, as.factor)

# Convert factor variables to numeric
data_updated <- data_updated %>%
  mutate_if(is.factor, as.numeric)

# Perform Shapiro-Wilk normality test on numeric variables
for (col in names(data_updated)[sapply(data_updated, is.numeric)]) {
  print(paste("Normality test for:", col))
  print(shapiro.test(data_updated[[col]]))
  print("---")
}
```

```
[1] "Normality test for: S.no"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.95469, p-value = 9.578e-10

[1] "---"
[1] "Normality test for: Age"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.96323, p-value = 1.818e-08

[1] "---"
[1] "Normality test for: Blood.Pressure"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.8673, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Specific.Gravity"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.8776, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Albumin"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.70599, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Sugar"
```

```
        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.4339, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Red.Blood.Cells"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.3744, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Pus.cells"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.4783, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Pus.cell.clumps"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.35165, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Bacteria"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.238, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Blood.glucose.random"

        Shapiro-Wilk normality test
```

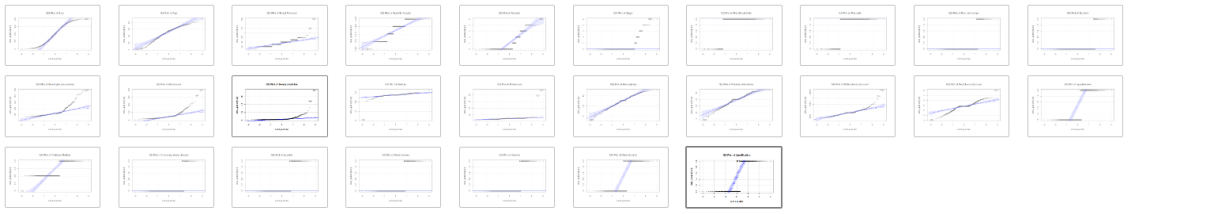
```
data:  data_updated[[col]]
W = 0.74109, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Blood.urea"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.71225, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Serum.creatinine"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.38989, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Sodium"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.56688, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Potassium"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.18237, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Hemoglobin"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.98377, p-value = 0.0001838
```

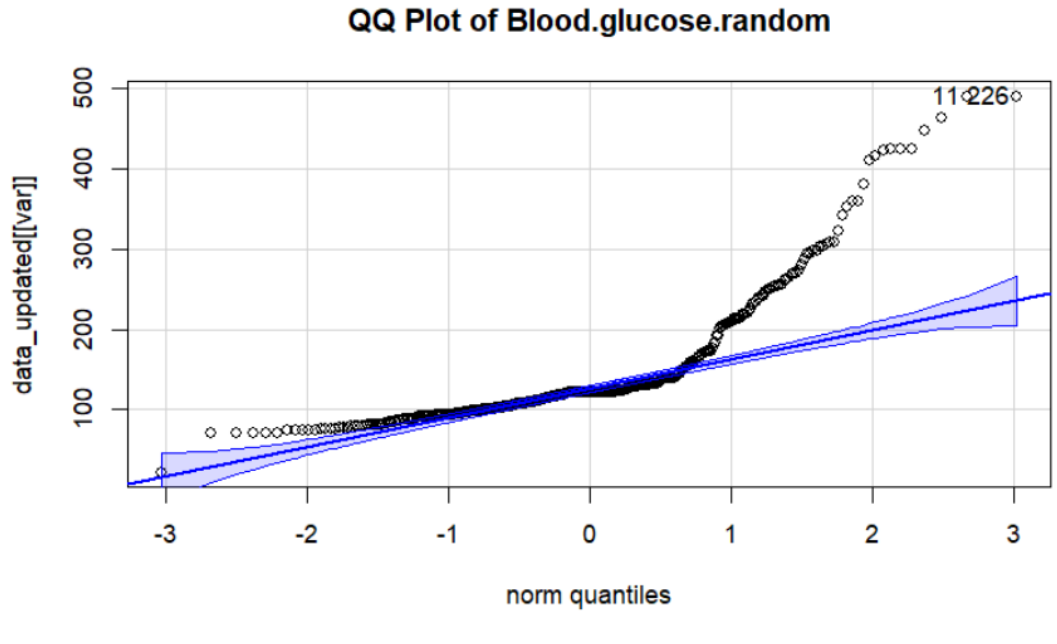
```
[1] "---"
[1] "Normality test for: Packed.cell.volume"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.96593, p-value = 5.005e-08

[1] "---"
[1] "Normality test for: White.blood.cell.count"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.85983, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Red.blood.cell.count"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.93832, p-value = 7.993e-12

[1] "---"
[1] "Normality test for: hypertension"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.61046, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Diabetes.Mellitus"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.61294, p-value < 2.2e-16

[1] "---"
```

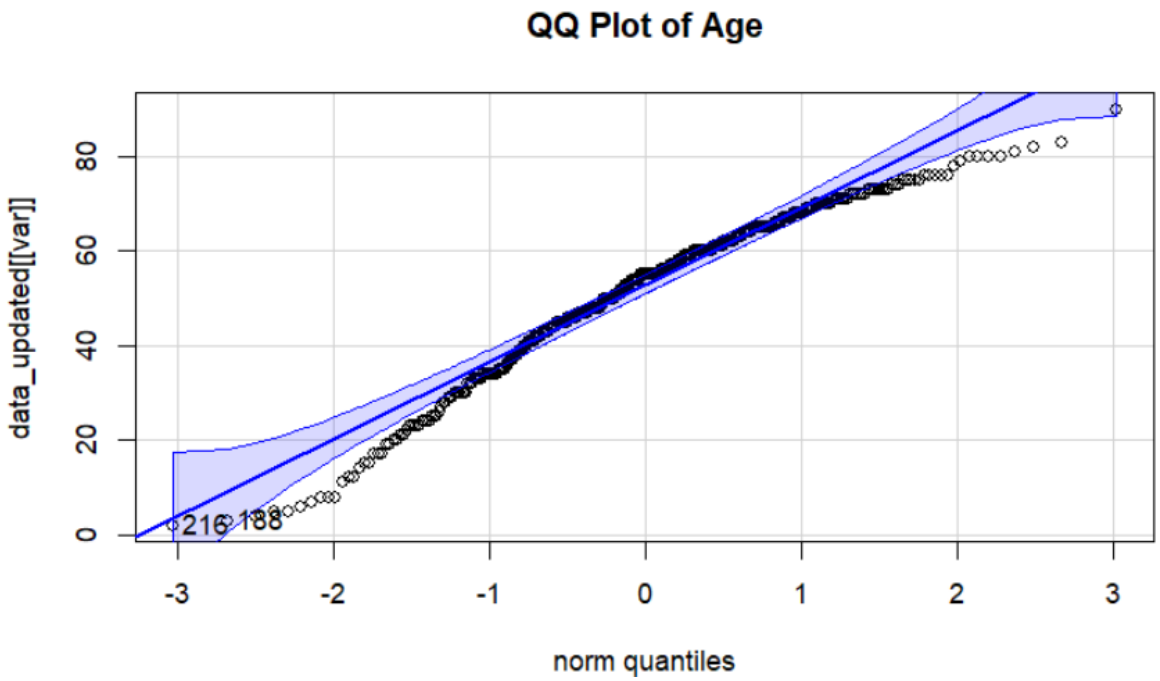
```
[1] "Normality test for: Coronary.artery.disease"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.31116, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Appetite"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.49515, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Pedal.edema"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.4783, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Anemia"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.42618, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: Classification"

        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.61338, p-value < 2.2e-16

[1] "---"
[1] "Normality test for: classification"
```

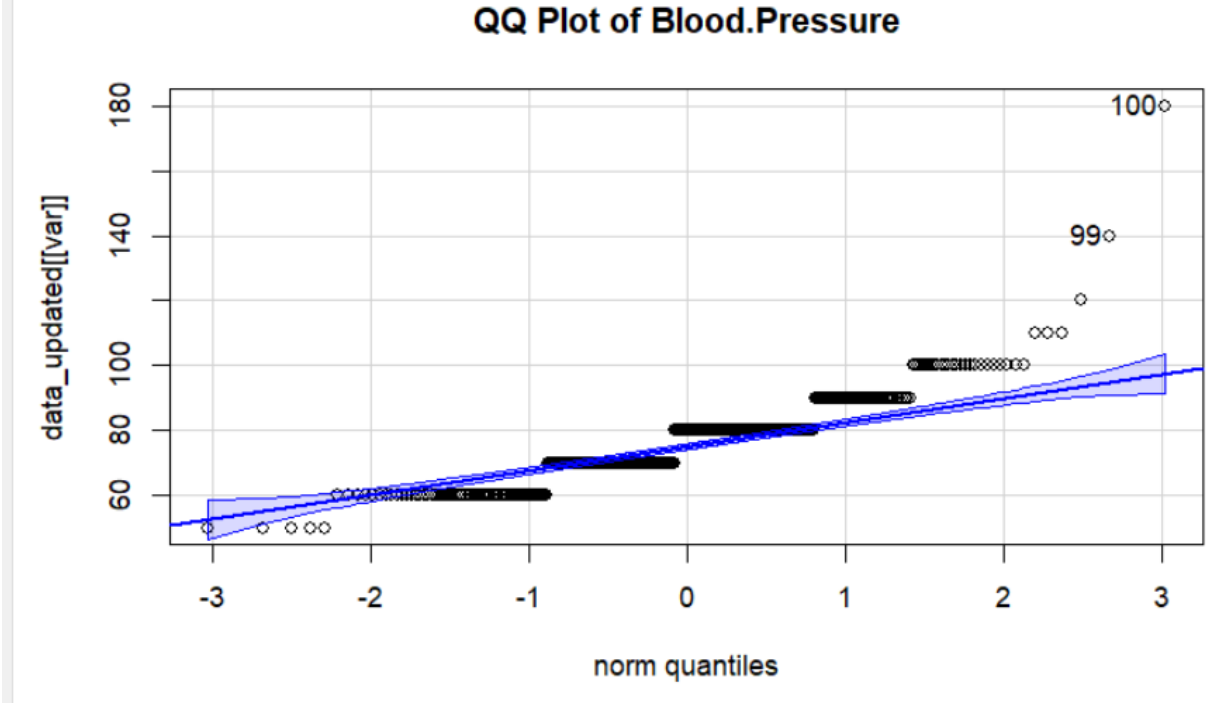
```
        Shapiro-Wilk normality test

data:  data_updated[[col]]
W = 0.61338, p-value < 2.2e-16

[1] "---"
```

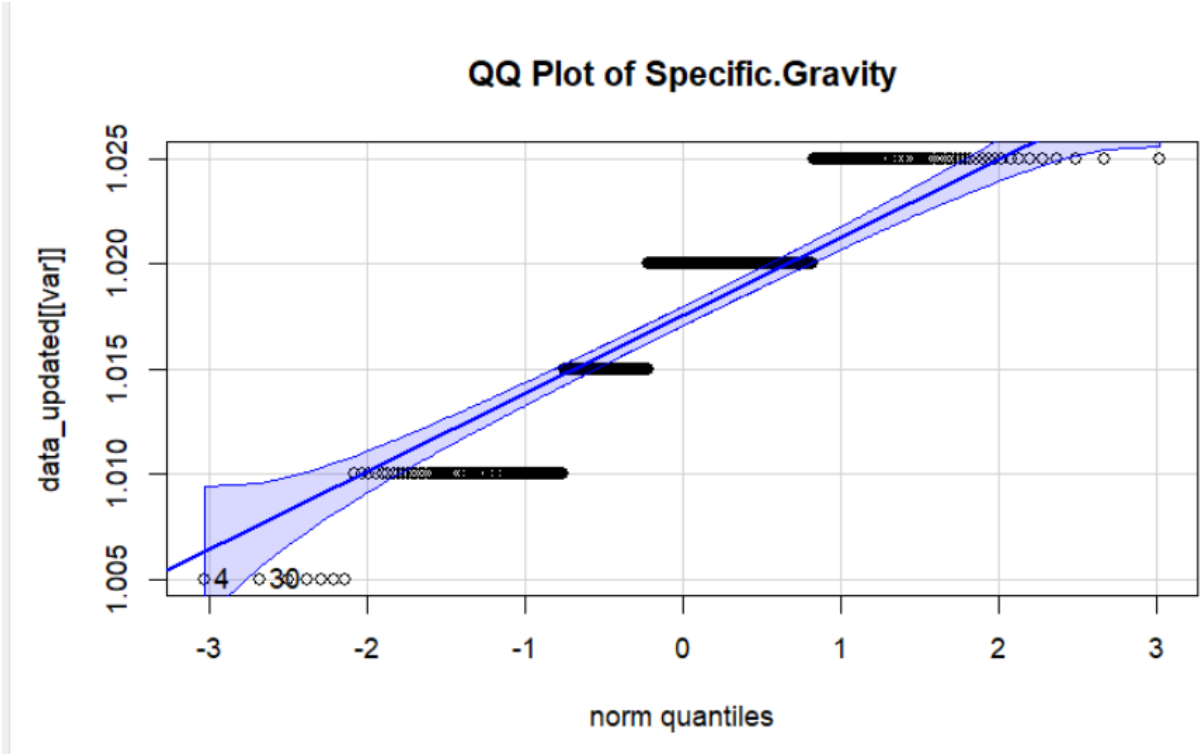
```{r}
library(car)

# Loop through each numeric column in the dataset
par(mfrow = c(2, 2)) # Adjust the layout depending on the number of numeric variables
for (var in names(data_updated)) {
  if (is.numeric(data_updated[[var]])) {
    qqPlot(data_updated[[var]], main = paste("QQ Plot of", var))
  }
}
```

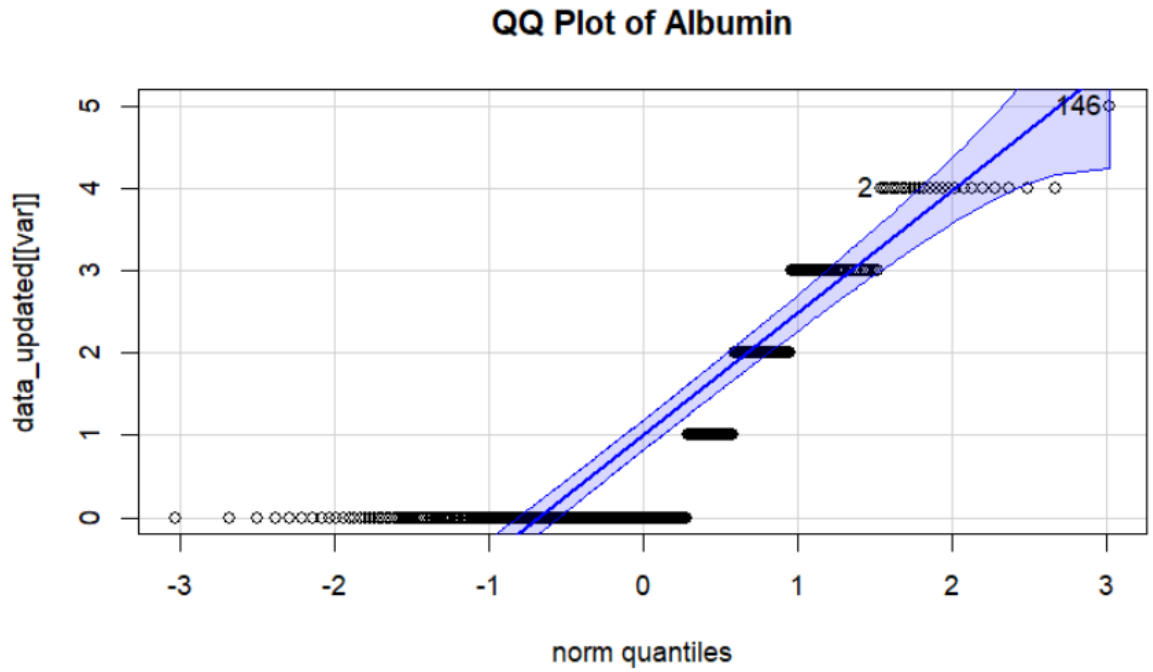


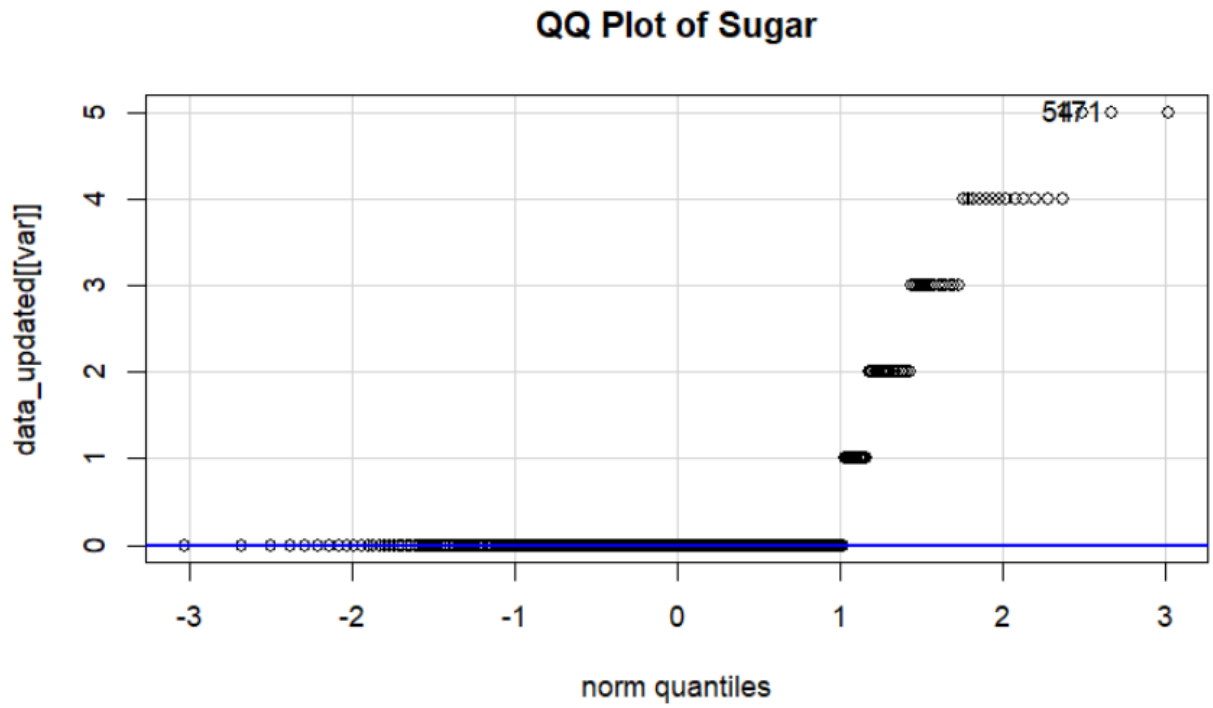


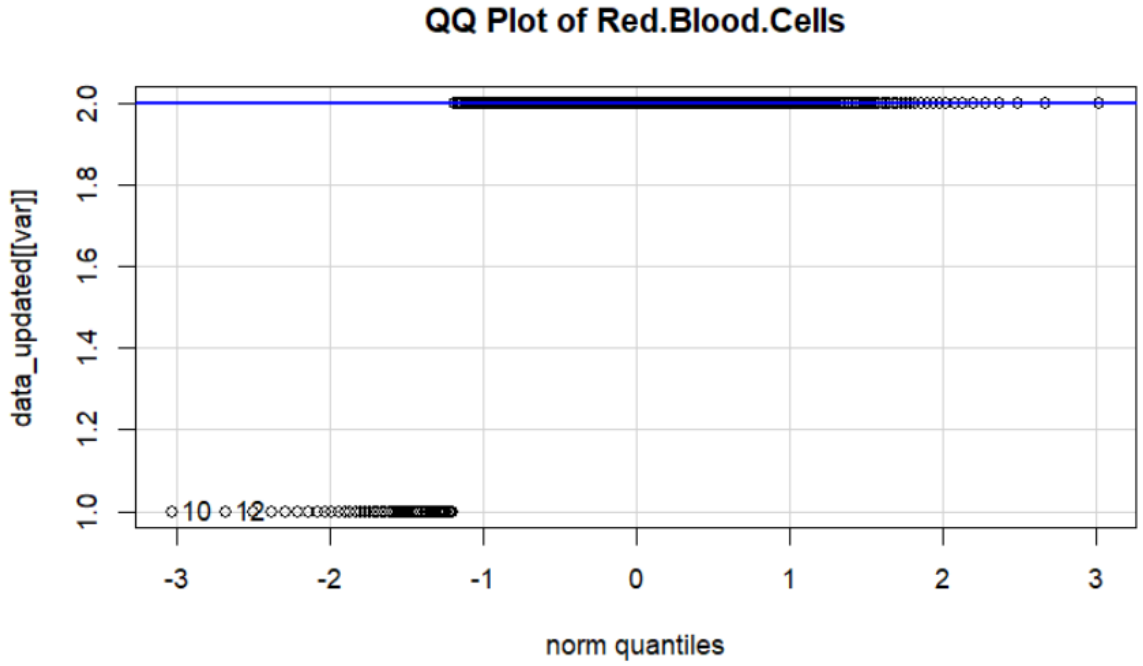

QQ Plot of Age
data_updated[[var]]
0
20
40
60
80
216
188
-3
-2
-1
0
1
2
3
norm quantiles

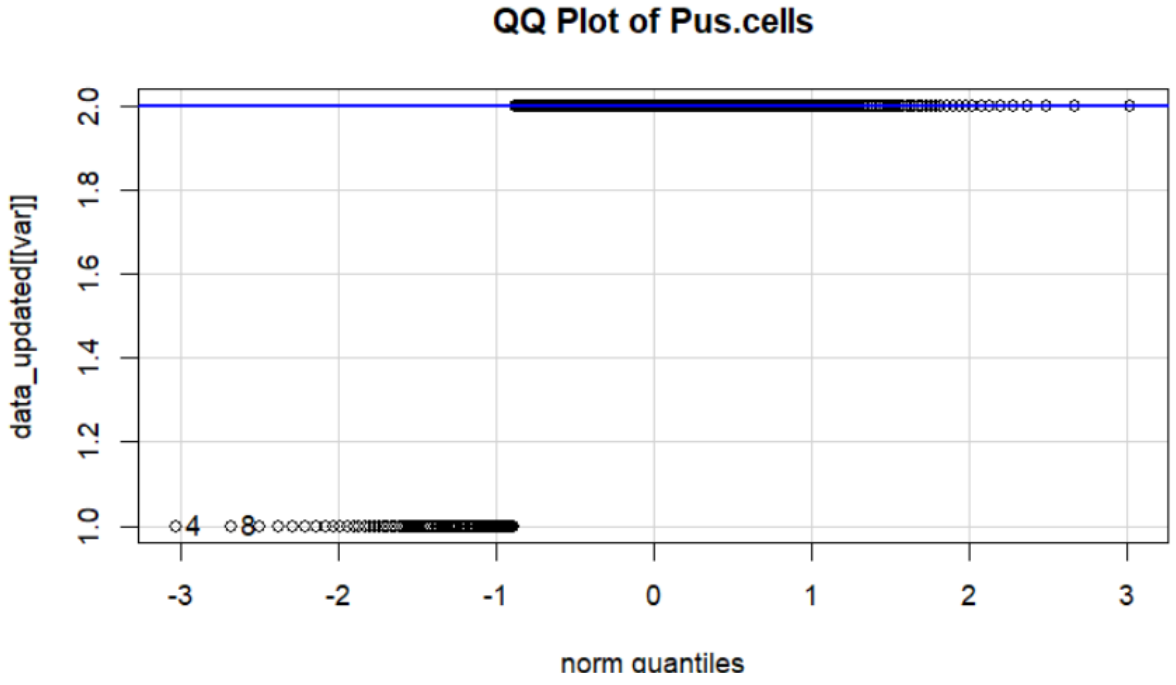

QQ Plot of Blood.Pressure
data_updated[[var]]
60
80
100
140
180
100
99
-3
-2
-1
0
1
2
3
norm quantiles

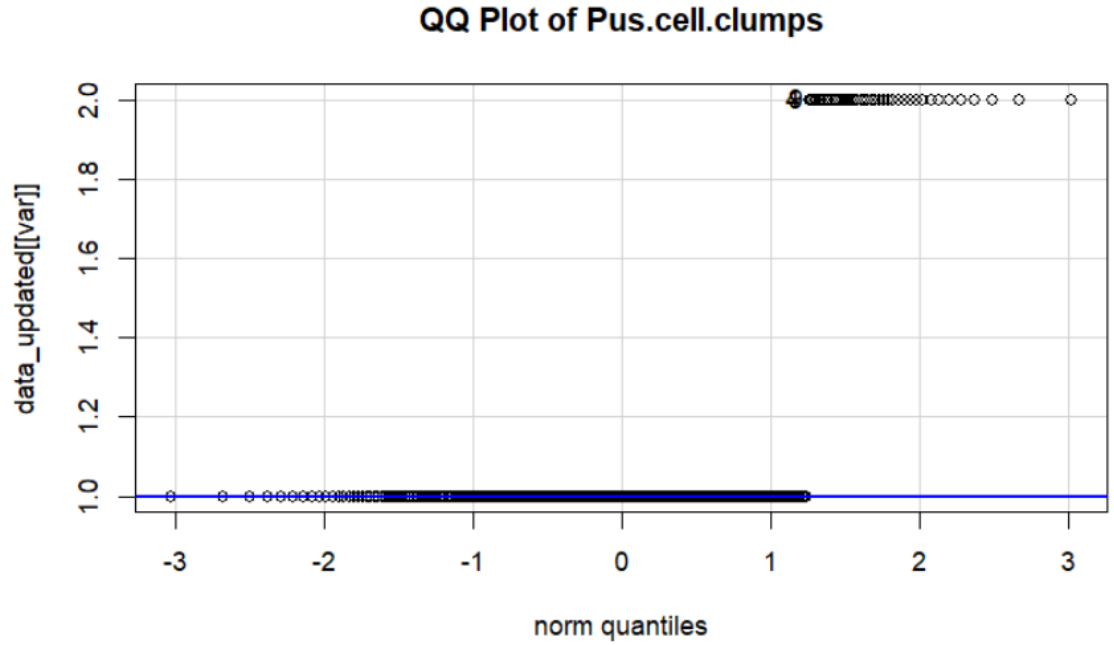

QQ Plot of Specific.Gravity
data_updated[[var]]
1.005
1.010
1.015
1.020
1.025
4
30
-3
-2
-1
0
1
2
3
norm quantiles

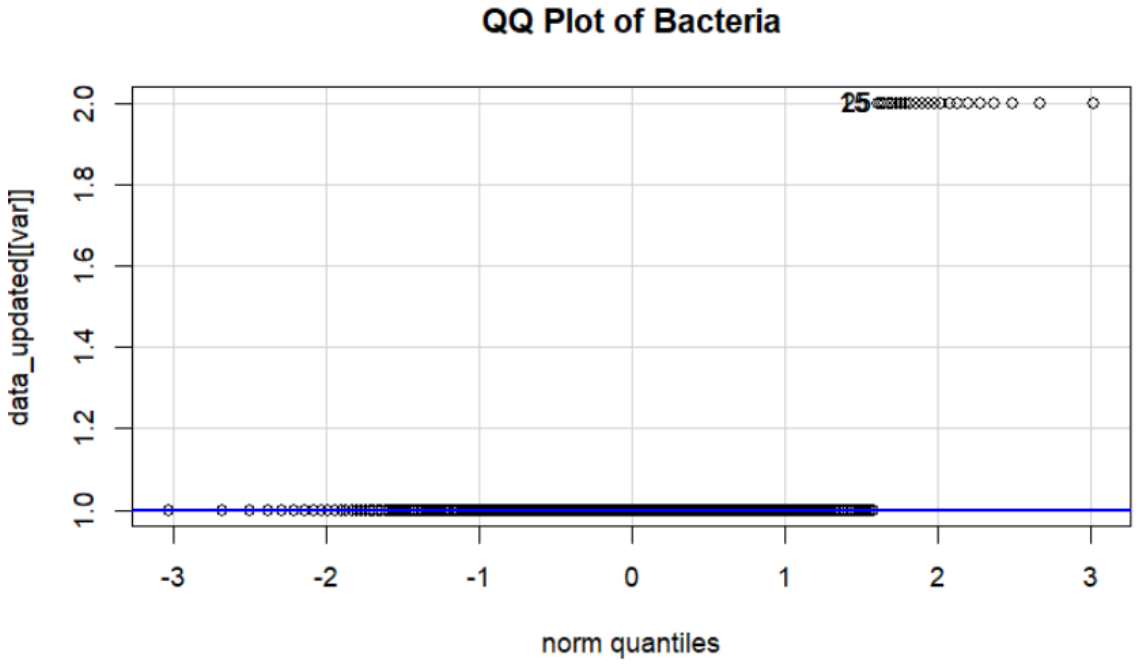

QQ Plot of Albumin
data_updated[[var]]
0
1
2
3
4
5
146
2
-3
-2
-1
0
1
2
3
norm quantiles

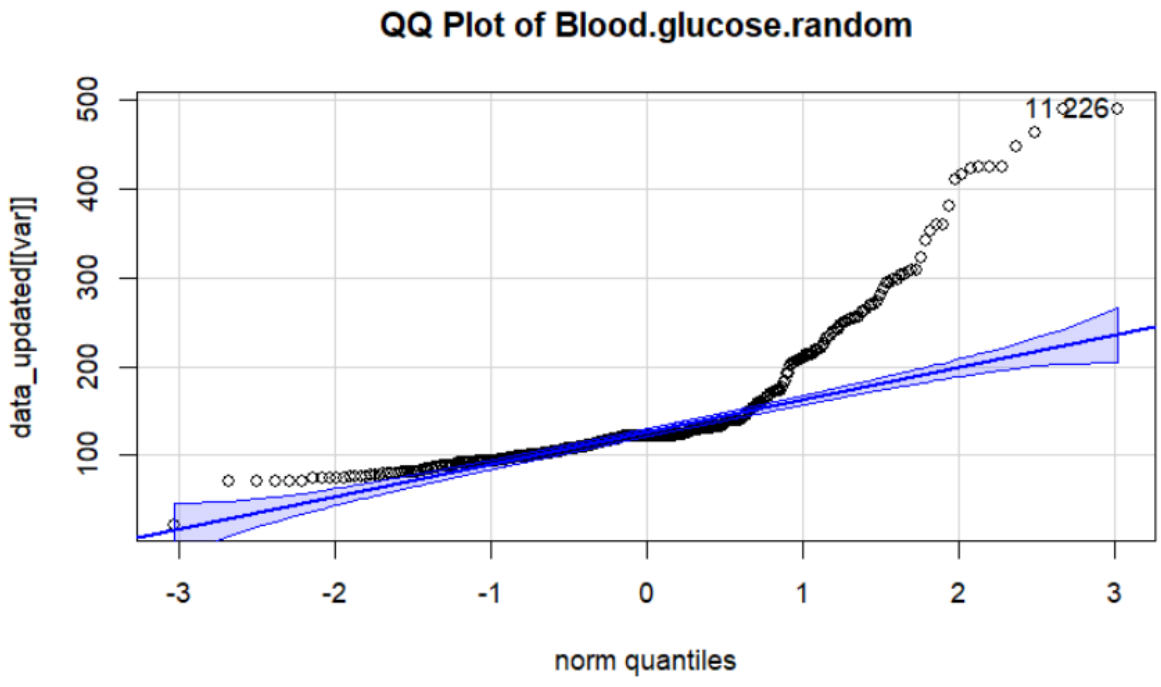

QQ Plot of Sugar
data_updated[[var]]
norm quantiles
0
1
2
3
4
5
-3
-2
-1
0
1
2
3

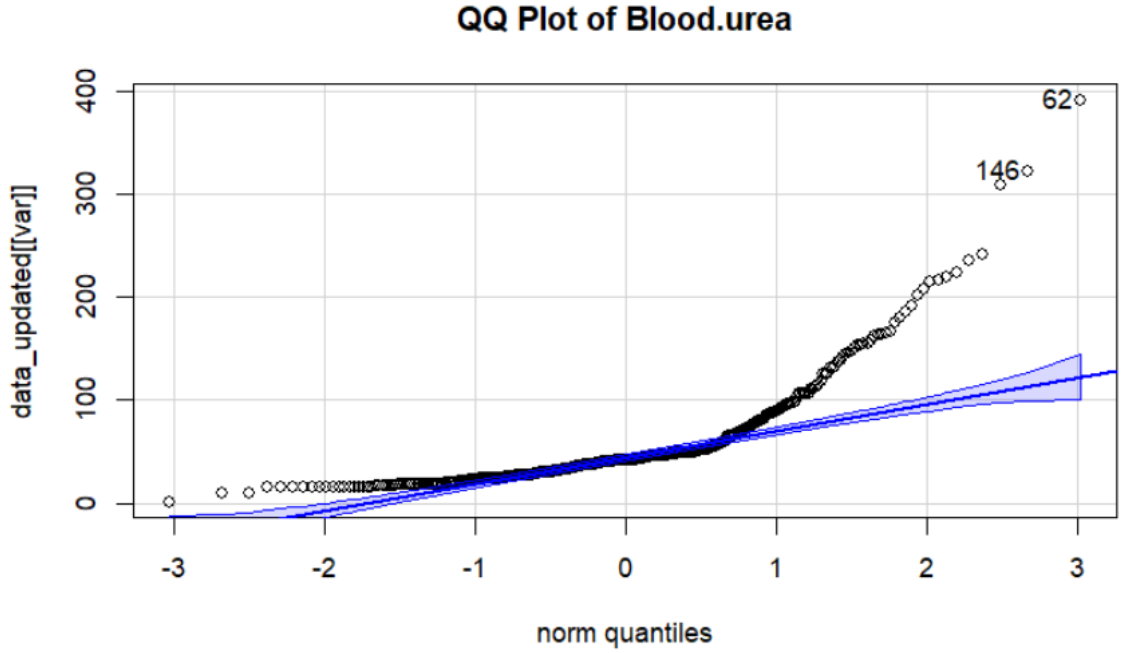

QQ Plot of Red.Blood.Cells
data_updated[[var]]
norm quantiles
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3

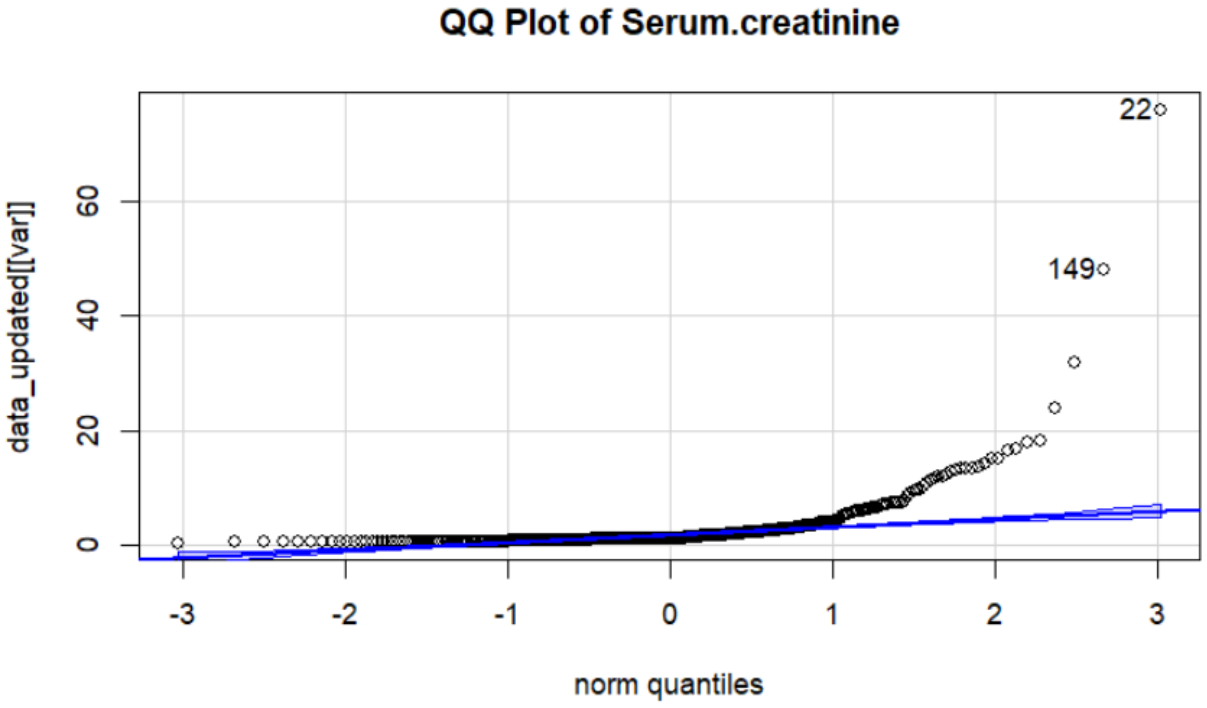

QQ Plot of Pus.cells
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
4
8
-3
-2
-1
0
1
2
3
norm quantiles

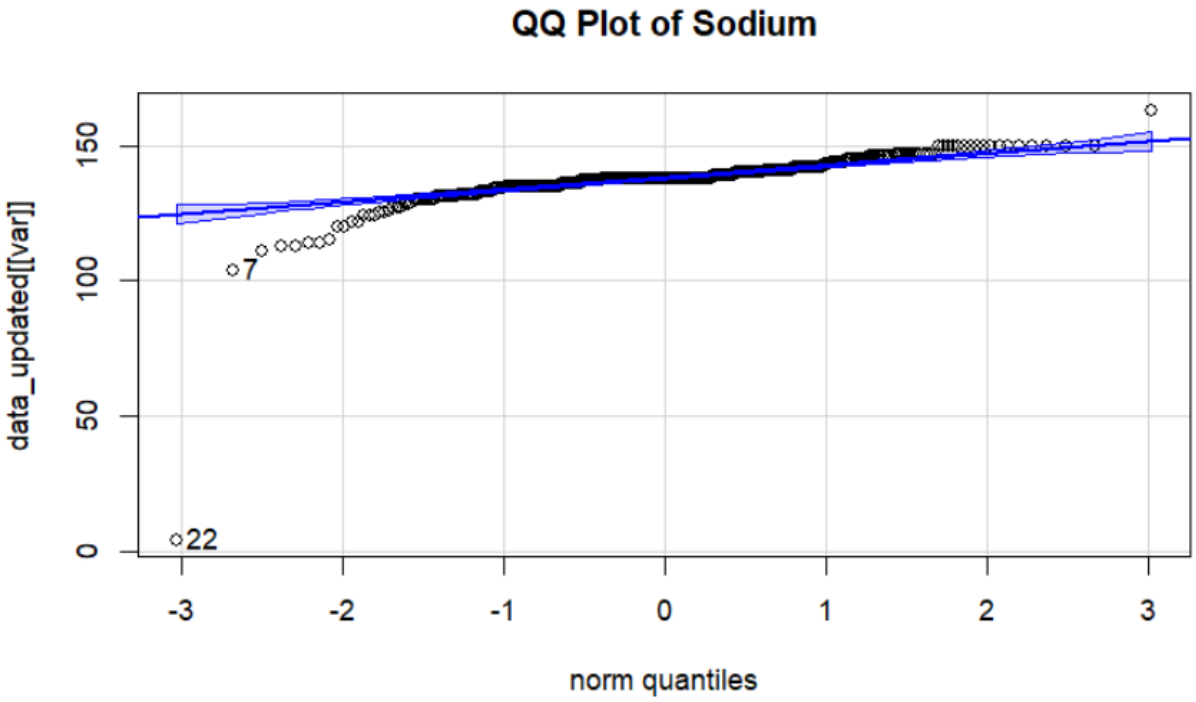

QQ Plot of Pus.cell.clumps
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
norm quantiles

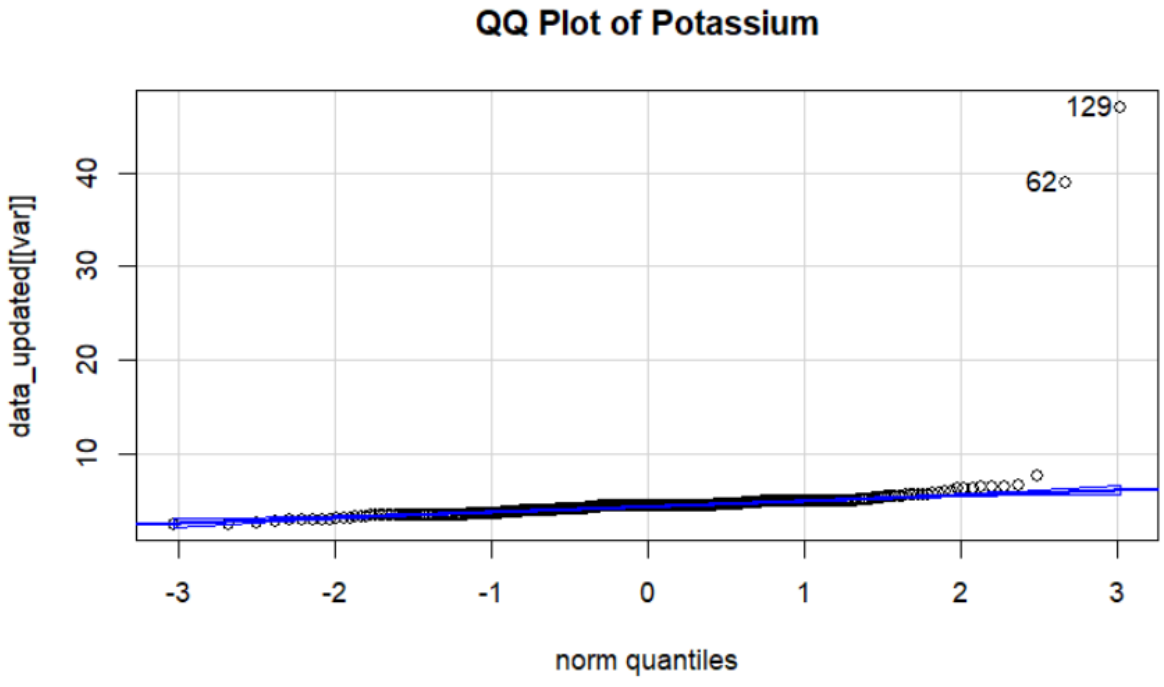

QQ Plot of Bacteria
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
25
-3
-2
-1
0
1
2
3
norm quantiles

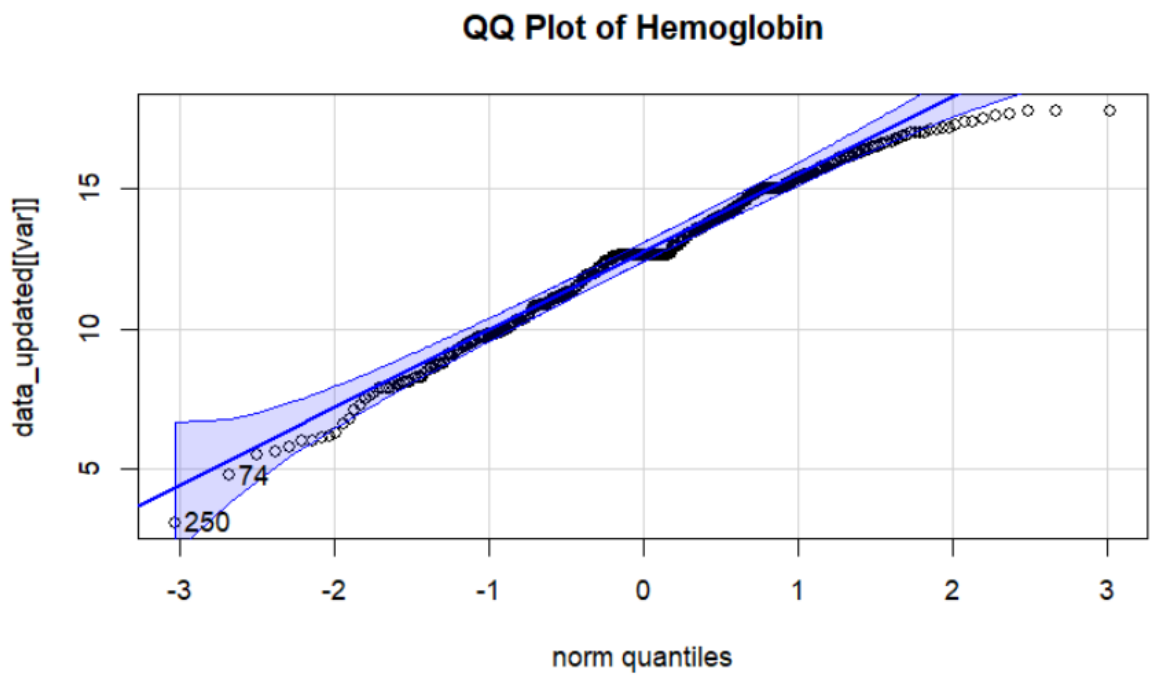

QQ Plot of Blood.glucose.random
data_updated[[var]]
100
200
300
400
500
11
226
-3
-2
-1
0
1
2
3
norm quantiles

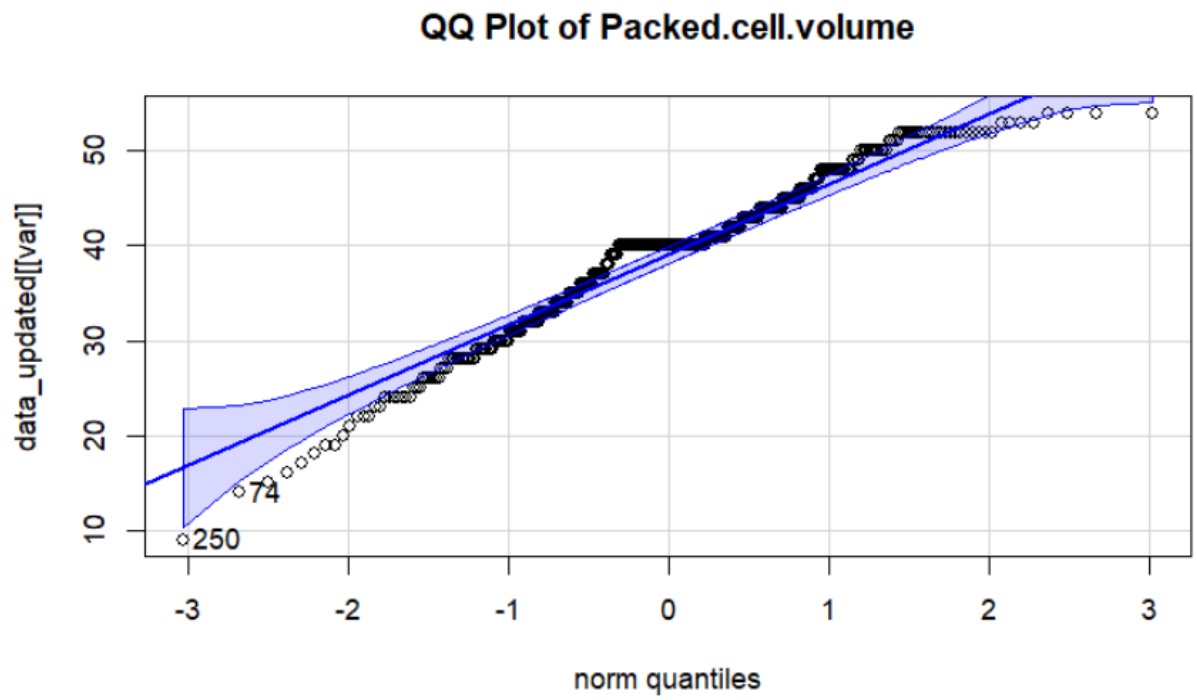

QQ Plot of Blood.urea
data_updated[[var]]
0
100
200
300
400
62
146
-3
-2
-1
0
1
2
3
norm quantiles

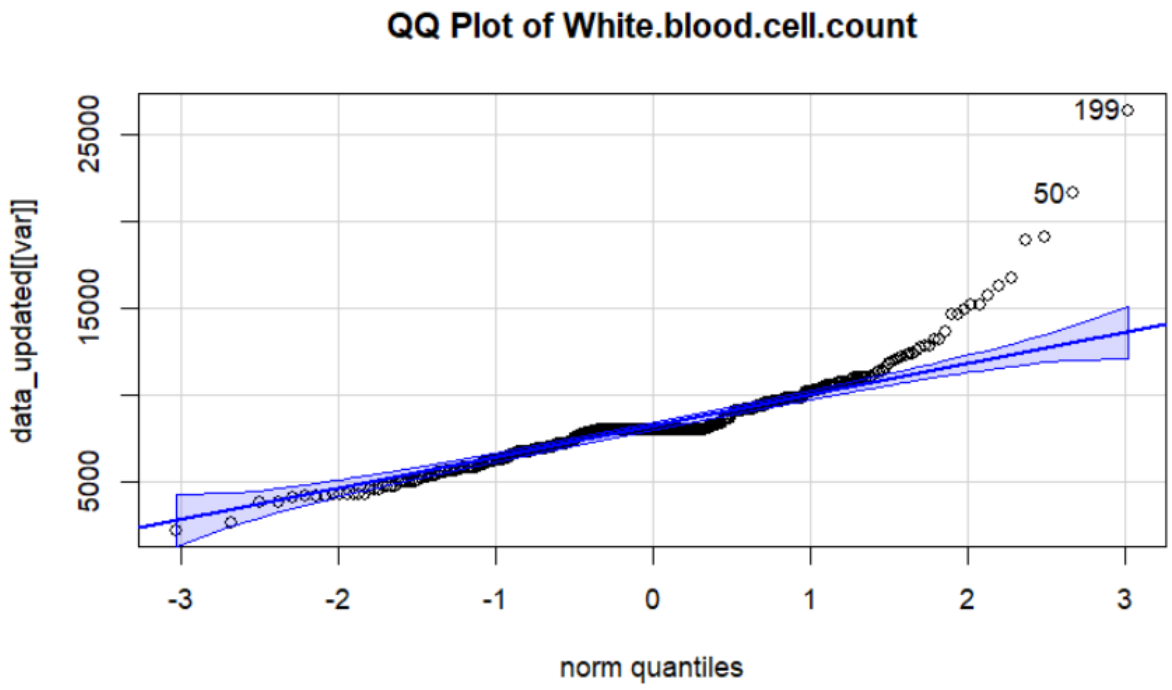

QQ Plot of Serum.creatinine
data_updated[[var]]
0
20
40
60
22
149
-3
-2
-1
0
1
2
3
norm quantiles

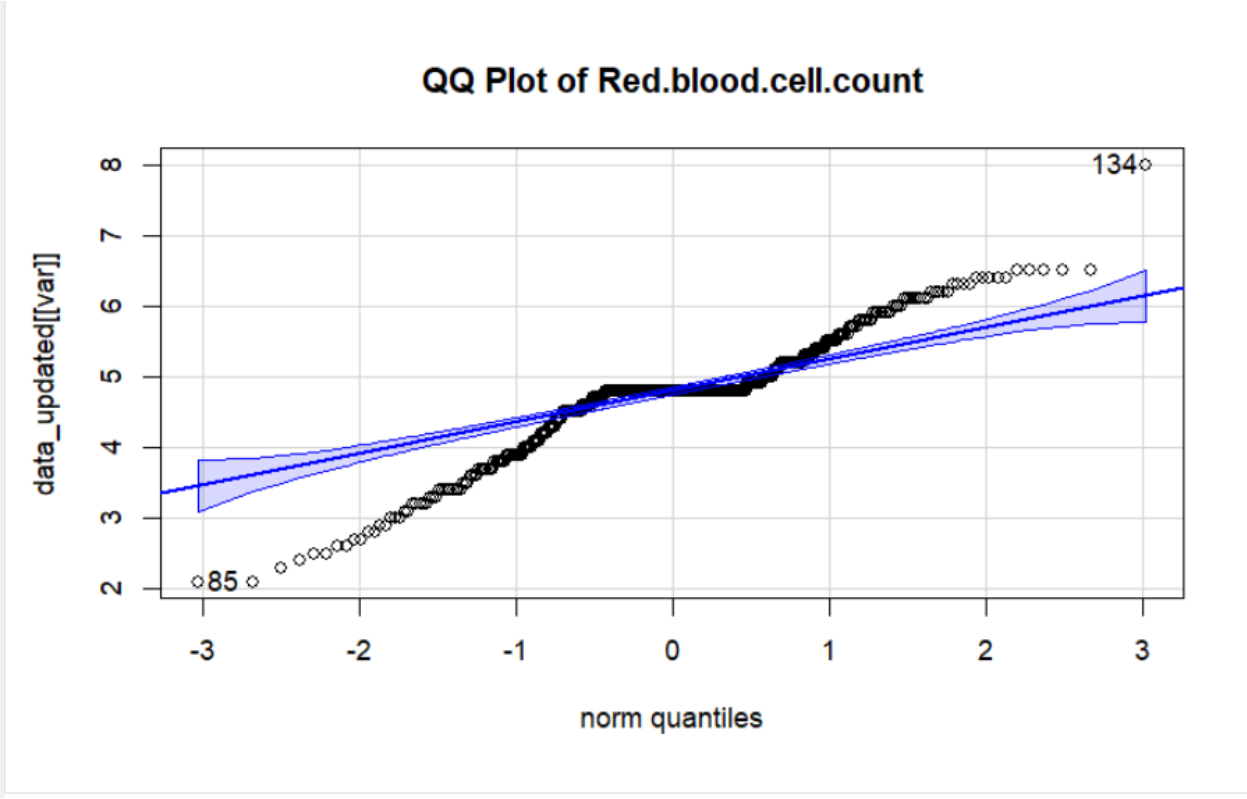

QQ Plot of Sodium
data_updated[[var]]
0
50
100
150
22
7
-3
-2
-1
0
1
2
3
norm quantiles

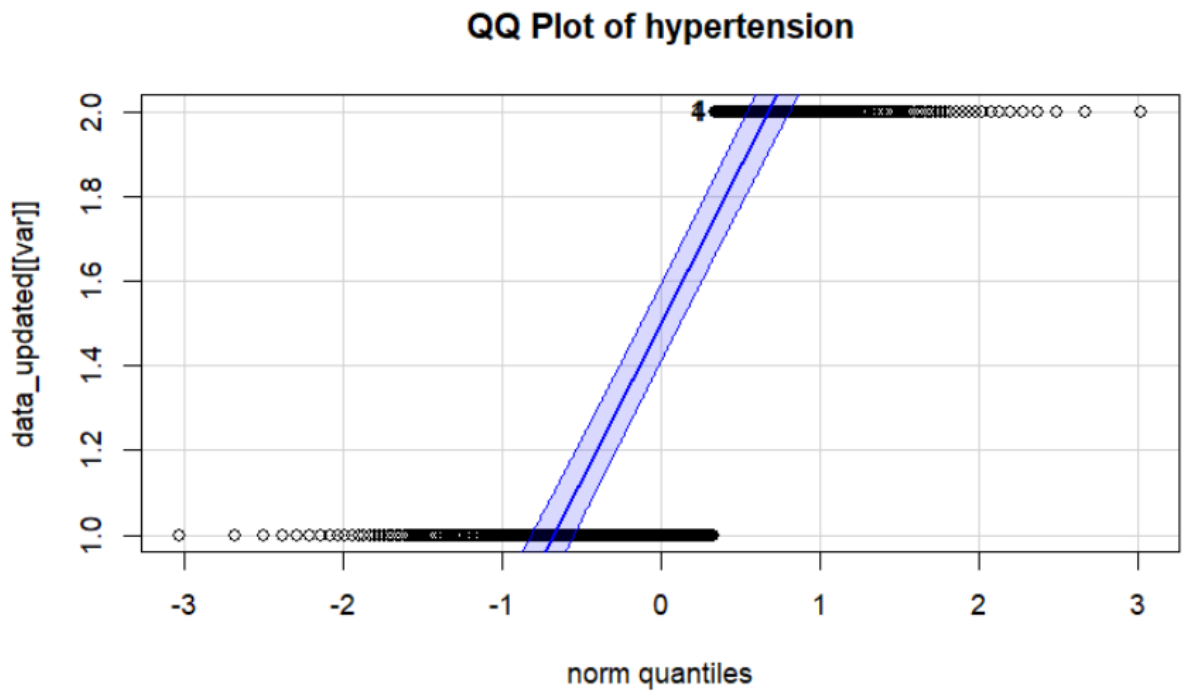

QQ Plot of Potassium
data_updated[[var]]
10
20
30
40
129
62
-3
-2
-1
0
1
2
3
norm quantiles

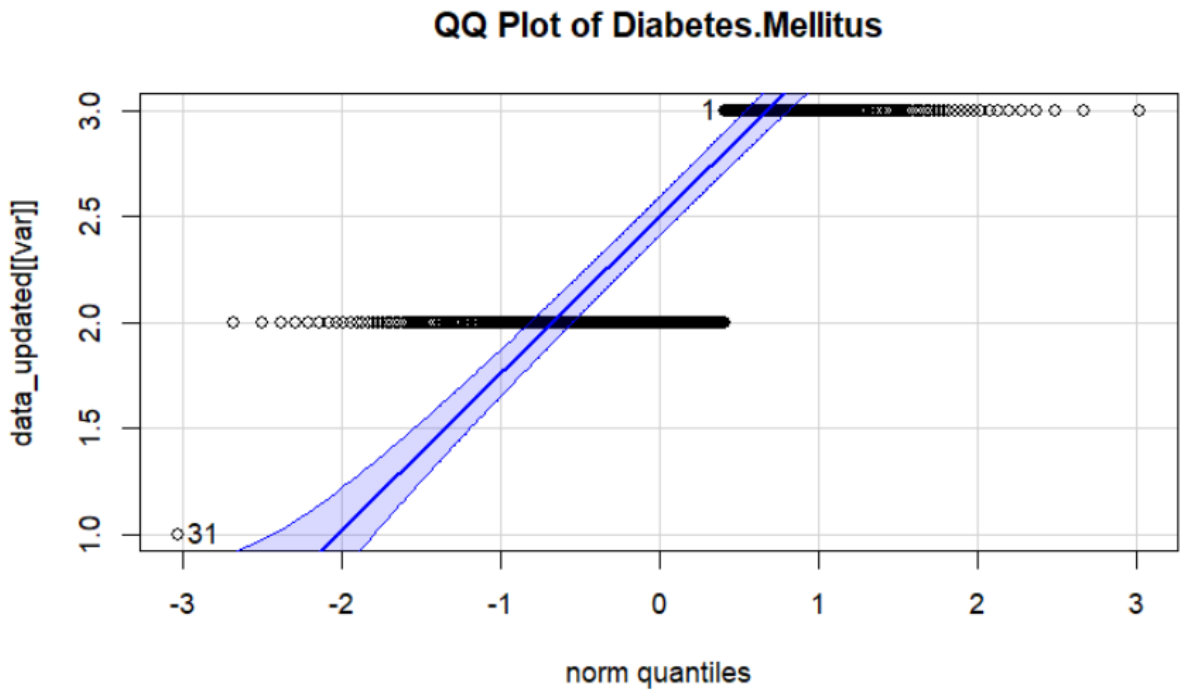

QQ Plot of Hemoglobin
data_updated[[var]]
5
10
15
74
250
-3
-2
-1
0
1
2
3
norm quantiles

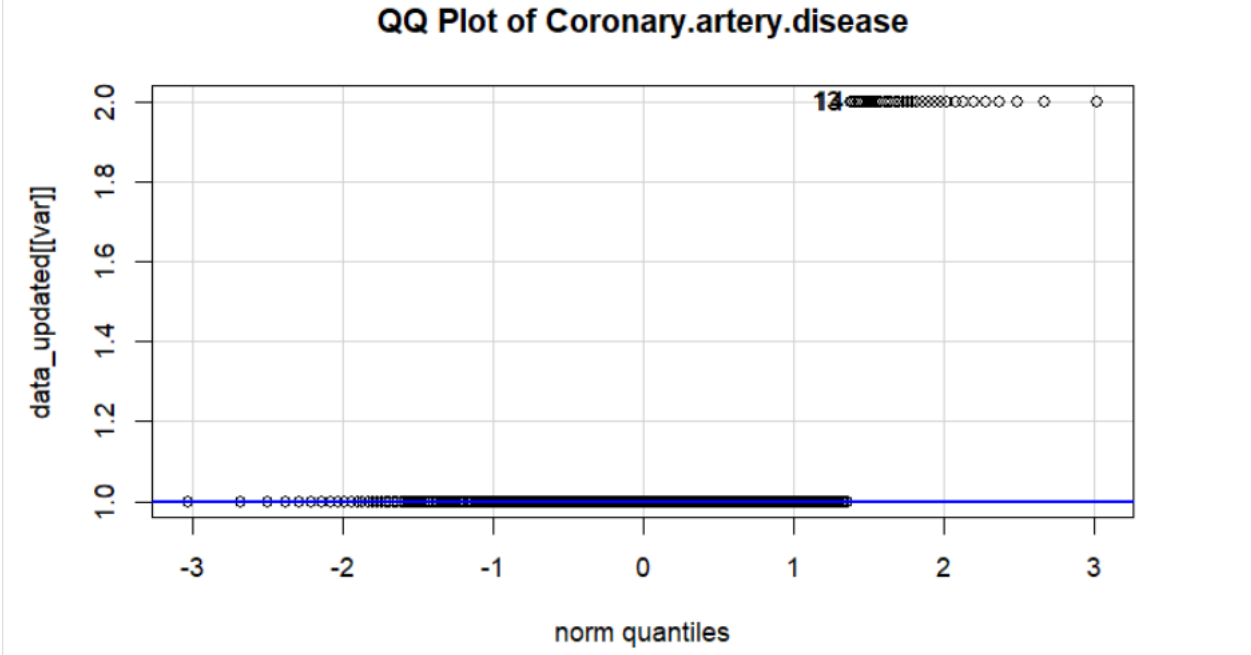

QQ Plot of Packed.cell.volume
data_updated[[var]]
10
20
30
40
50
74
250
-3
-2
-1
0
1
2
3
norm quantiles

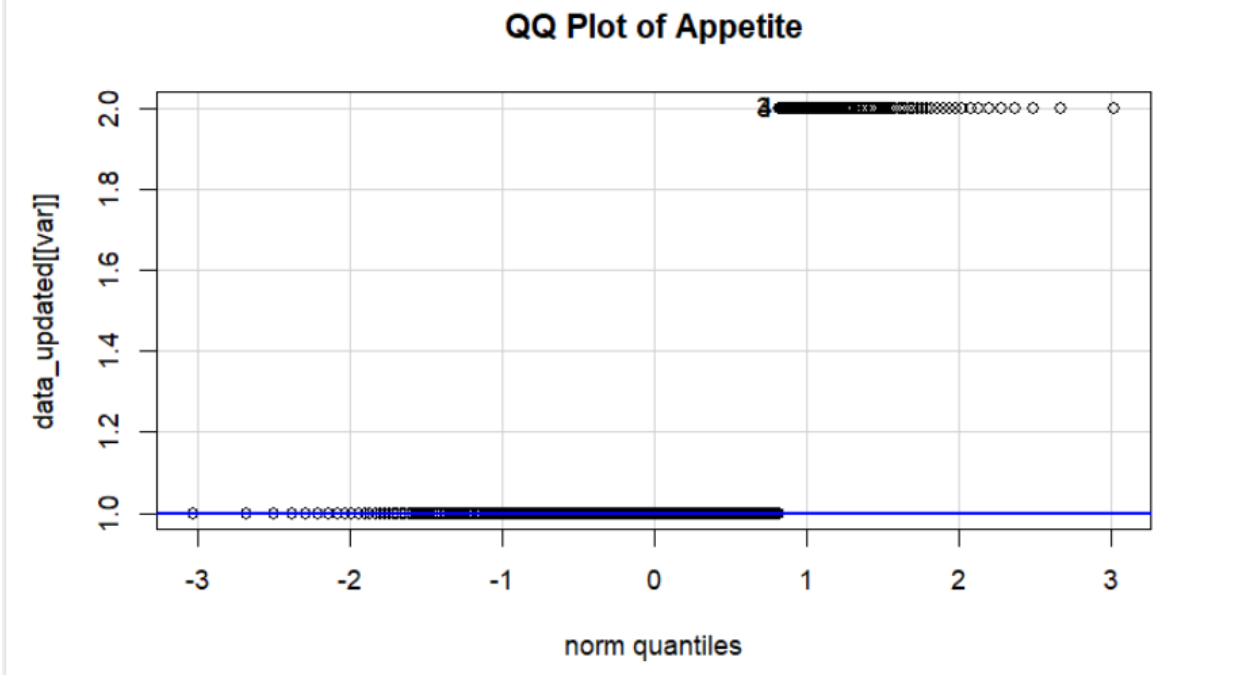

QQ Plot of White.blood.cell.count
data_updated[[var]]
25000
15000
5000
199
50
-3
-2
-1
0
1
2
3
norm quantiles

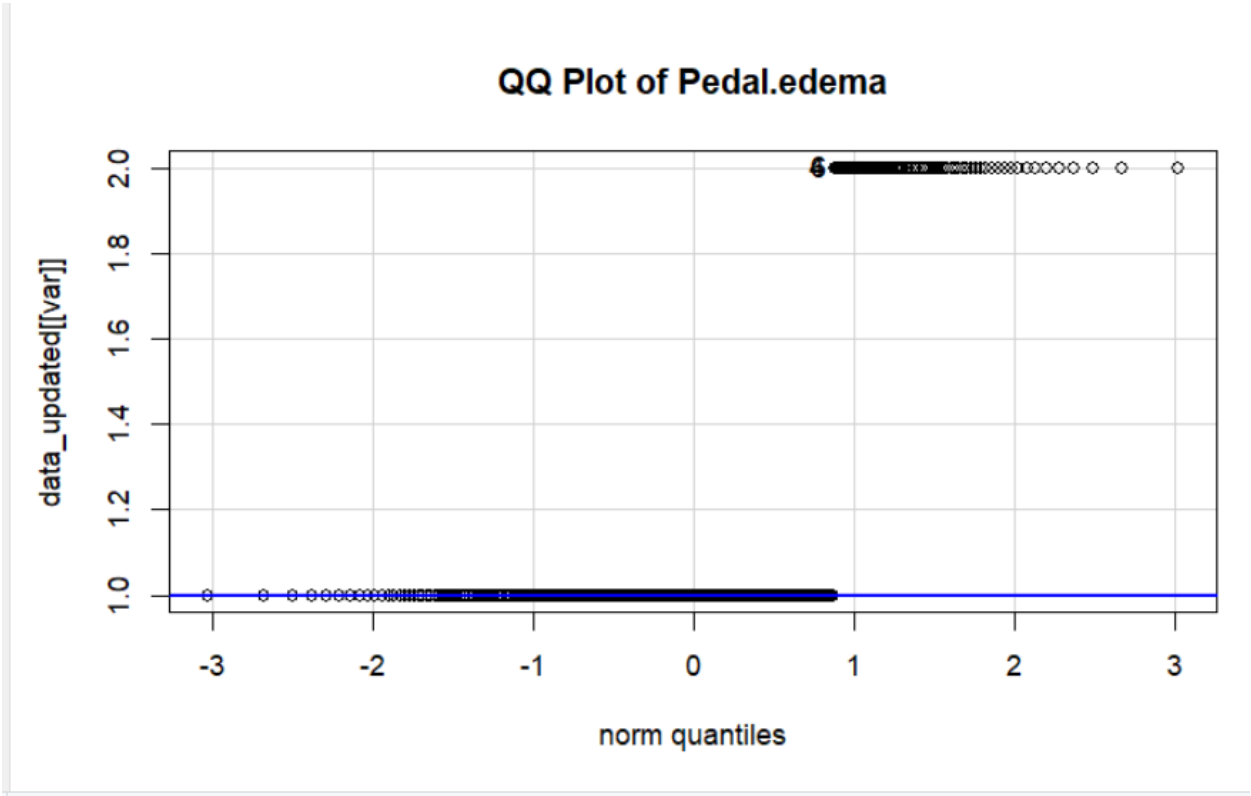

QQ Plot of Red.blood.cell.count
data_updated[[var]]
8
7
6
5
4
3
2
134
85
-3
-2
-1
0
1
2
3
norm quantiles

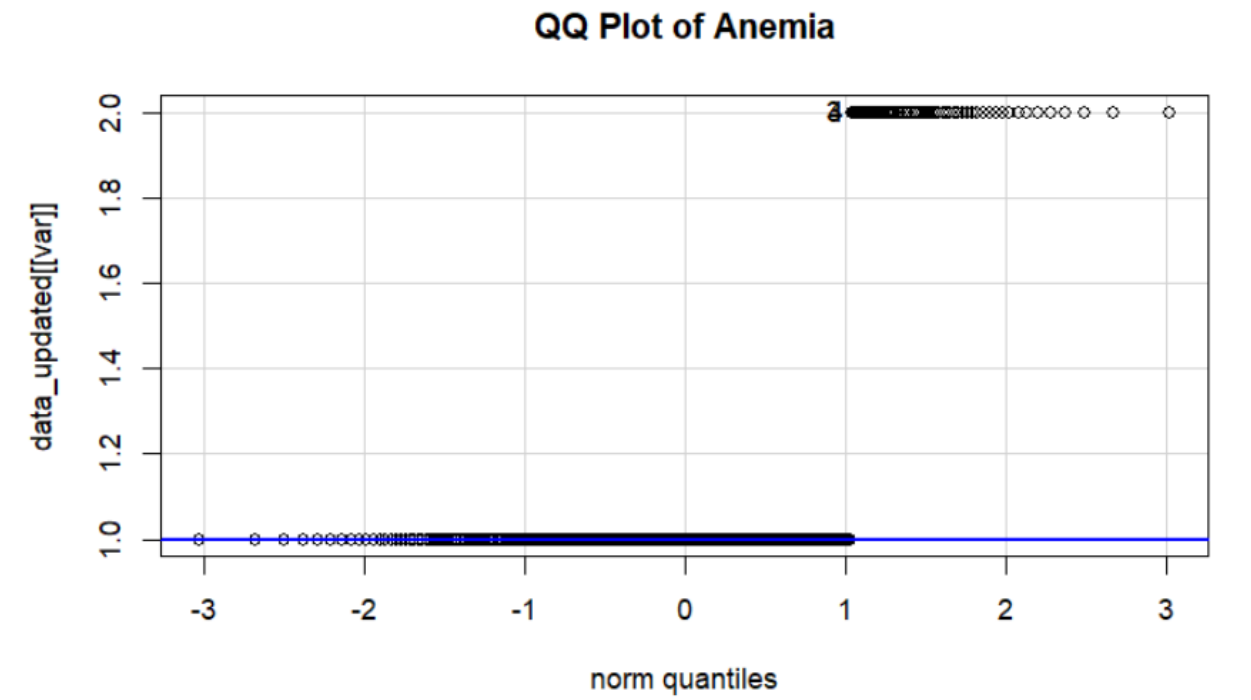

QQ Plot of hypertension
data_updated[[var]]
norm quantiles
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
4

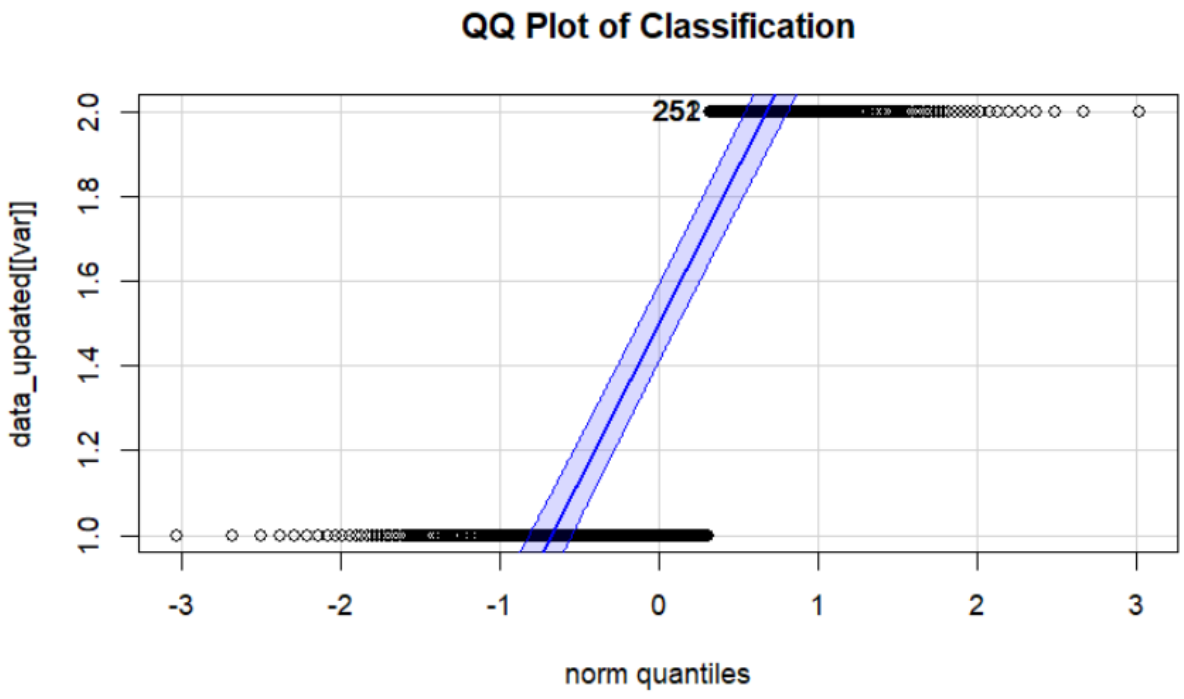

QQ Plot of Diabetes.Mellitus
data_updated[[var]]
norm quantiles
1.0
1.5
2.0
2.5
3.0
-3
-2
-1
0
1
2
3
1
31

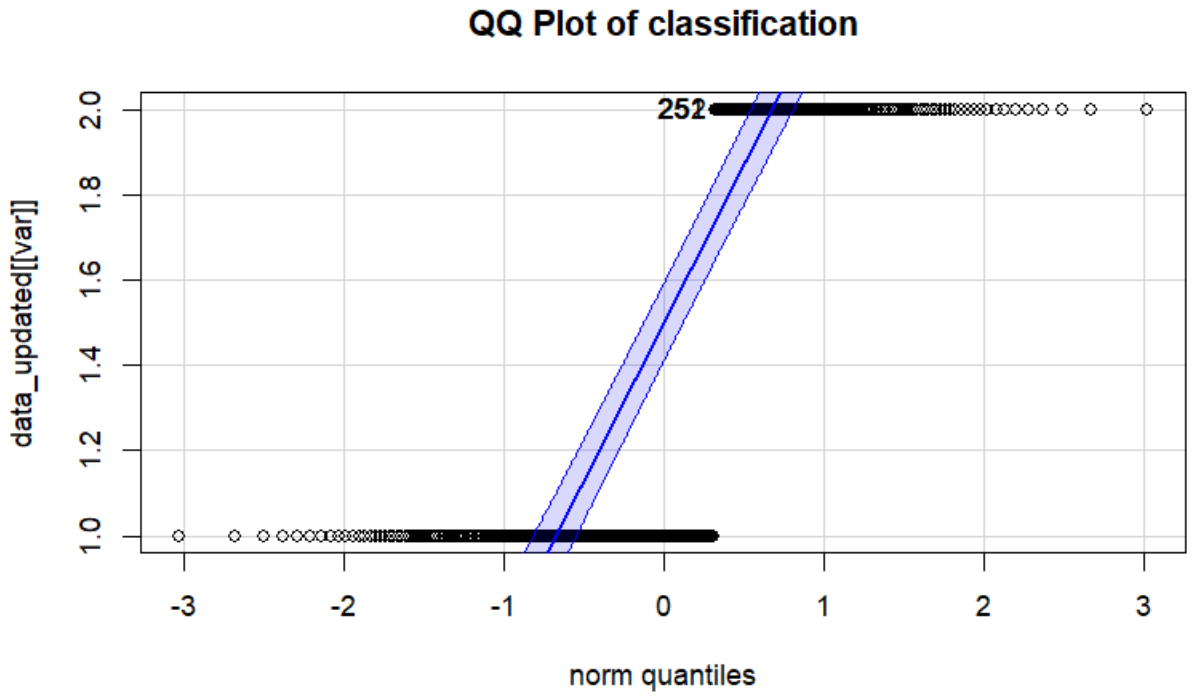

QQ Plot of Coronary.artery.disease
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
norm quantiles

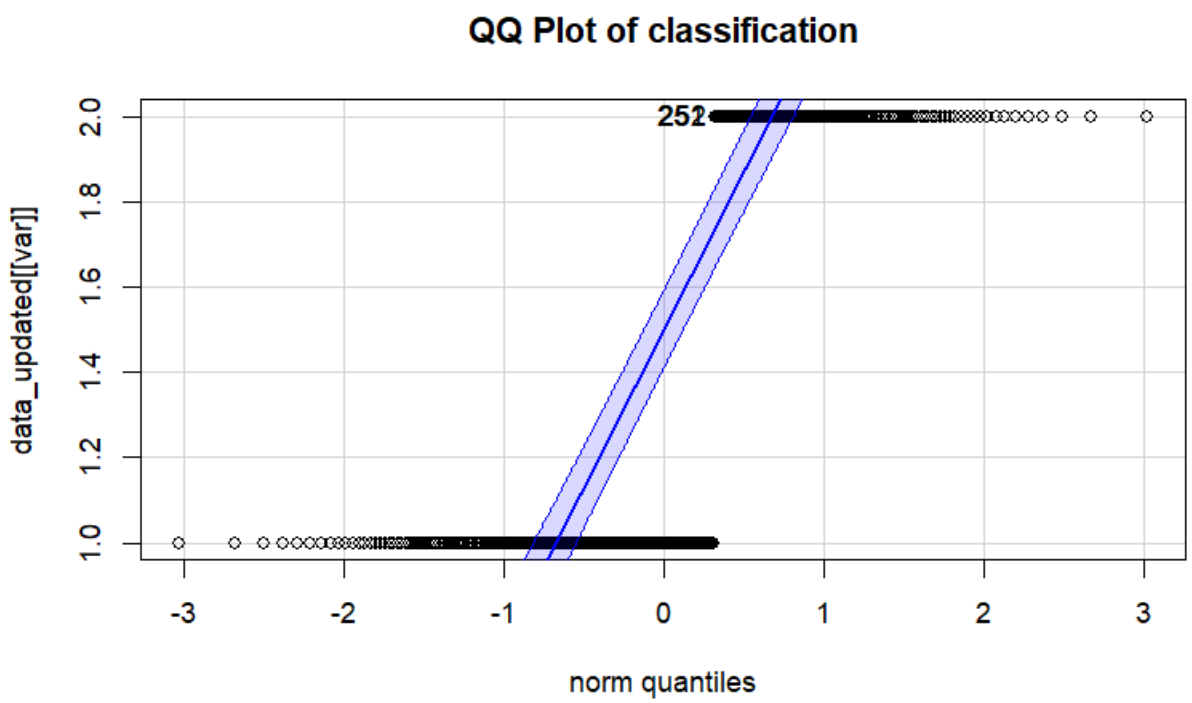

QQ Plot of Appetite
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
norm quantiles


QQ Plot of Pedal.edema
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
norm quantiles


QQ Plot of Anemia
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
norm quantiles


QQ Plot of Classification
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
252
-3
-2
-1
0
1
2
3
norm quantiles


QQ Plot of classification
252
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
norm quantiles


QQ Plot of classification
252
data_updated[[var]]
1.0
1.2
1.4
1.6
1.8
2.0
-3
-2
-1
0
1
2
3
norm quantiles

Mann Whitney U test to check the significance for numerical variables

```{r}
wilcox.test(Age ~ Classification, data = data_updated)
wilcox.test(Blood.Pressure ~ Classification, data = data_updated)
wilcox.test(Albumin ~ Classification, data = data_updated)
wilcox.test(Serum.creatinine ~ Classification, data = data_updated)
wilcox.test(Hemoglobin ~ Classification, data = data_updated)
wilcox.test(Specific.Gravity ~ Classification, data = data_updated)
wilcox.test( Sugar~ Classification, data = data_updated)
wilcox.test(Red.Blood.Cells ~ Classification, data = data_updated)
wilcox.test(Pus.cells ~ Classification, data = data_updated)
wilcox.test(Pus.cell.clumps ~ Classification, data = data_updated)
wilcox.test(Bacteria ~ Classification, data = data_updated)
wilcox.test(Blood.glucose.random ~ Classification, data = data_updated)
wilcox.test(Blood.urea ~ Classification, data = data_updated)
wilcox.test(Sodium ~ Classification, data = data_updated)
wilcox.test(Potassium ~ Classification, data = data_updated)
wilcox.test(Packed.cell.volume ~ Classification, data = data_updated)
wilcox.test(White.blood.cell.count ~ Classification, data = data_updated)
wilcox.test(Red.blood.cell.count ~ Classification, data = data_updated)
```

```
        Wilcoxon rank sum test with continuity correction

data:  Age by Classification
W = 24941, p-value = 3.17e-08
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Blood.Pressure by Classification
W = 25028, p-value = 6.94e-09
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Albumin by Classification
W = 30375, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Serum.creatinine by Classification
W = 34557, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Hemoglobin by Classification
W = 1235.5, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Specific.Gravity by Classification
W = 4185, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0
```

```
        Wilcoxon rank sum test with continuity correction

data:  Sugar by Classification
W = 23325, p-value = 6.38e-11
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Red.Blood.Cells by Classification
W = 15225, p-value = 1.652e-08
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Pus.cells by Classification
W = 13050, p-value = 6.729e-14
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Pus.cell.clumps by Classification
W = 21900, p-value = 1.166e-07
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Bacteria by Classification
W = 20400, p-value = 0.0001903
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Blood.glucose.random by Classification
W = 27619, p-value = 2.198e-15
alternative hypothesis: true location shift is not equal to 0
```

```
        Wilcoxon rank sum test with continuity correction

data:  Blood.urea by Classification
W = 27911, p-value = 2.728e-16
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Sodium by Classification
W = 8291, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Potassium by Classification
W = 18421, p-value = 0.7671
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Packed.cell.volume by Classification
W = 1911.5, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  White.blood.cell.count by Classification
W = 22363, p-value = 0.001116
alternative hypothesis: true location shift is not equal to 0


        Wilcoxon rank sum test with continuity correction

data:  Red.blood.cell.count by Classification
W = 5357, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0
```

Spearman Rank correlation test for assessing the association between 2 numerical variables

```{r}
spearman_result <- cor.test(data_updated$Age, data_updated$Hemoglobin, method = "spearman", use = "complete.obs")
spearman_result

```

```
        Spearman's rank correlation rho

data:  data_updated$Age and data_updated$Hemoglobin
S = 12984029, p-value = 1.165e-05
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho
-0.2172603
```

```{r}
spearman_result <- cor.test(data_updated$Age, data_updated$hypertension, method = "spearman", use = "complete.obs")
spearman_result
```

```
Warning: Cannot compute exact p-value with ties
        Spearman's rank correlation rho

data:  data_updated$Age and data_updated$hypertension
S = 6432073, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.3969894
```

```{r}
spearman_result <- cor.test(data_updated$Age, data_updated$Albumin, method = "spearman", use = "complete.obs")
spearman_result
```

```
Warning: Cannot compute exact p-value with ties
        Spearman's rank correlation rho

data:  data_updated$Age and data_updated$Albumin
S = 8880294, p-value = 0.000772
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.1674673
```

```{r}
install.packages("reshape2")
library(reshape2)
```

```
package 'reshape2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Dr Lavanya\AppData\Local\Temp\RtmpeqvAvG\downloaded_packages
```

Correlation Heat map to check the association between 2 numerical variables

```{r}
# Install and load necessary packages
if (!require(reshape2)) install.packages("reshape2")
if (!require(ggplot2)) install.packages("ggplot2")
library(reshape2)
library(ggplot2)

cor_matrix <- cor(data_updated[, sapply(data_updated, is.numeric)], use = "pairwise.complete.obs")

# Melt the correlation matrix for ggplot2
melted_cor_matrix <- melt(cor_matrix)

# Create a heatmap with ggplot2 and add text annotations
ggplot(melted_cor_matrix, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +  # Tiles with white borders
  geom_text(aes(label = sprintf("%.2f", value)), color = "black", size = 0.8) +  # Add text labels
  scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(x = "", y = "", fill = "Correlation") +
  theme(legend.key.height = unit(1, "in"))
```

Chi square test for categorical variables

```{r}
library(stats)

# Convert categorical variables to factors
data_updated <- data_updated %>%
  mutate_if(is.character, as.factor)

# Perform Chi-Square tests
for (col in c("hypertension", "Diabetes.Mellitus", "Coronary.artery.disease", "Appetite", "Pedal.edema", "Anemia")) {
  print(paste("Chi-Square test for:", col))
  print(chisq.test(data_updated[[col]], data_updated$classification))
  print("---")
}
```

```
[1] "Chi-Square test for: hypertension"

        Pearson's Chi-squared test with Yates' continuity correction

data:  data_updated[[col]] and data_updated$classification
X-squared = 136.93, df = 1, p-value < 2.2e-16

[1] "---"
[1] "Chi-Square test for: Diabetes.Mellitus"
Warning: Chi-squared approximation may be incorrect
        Pearson's Chi-squared test

data:  data_updated[[col]] and data_updated$classification
X-squared = 125.02, df = 2, p-value < 2.2e-16

[1] "---"
[1] "Chi-Square test for: Coronary.artery.disease"

        Pearson's Chi-squared test with Yates' continuity correction

data:  data_updated[[col]] and data_updated$classification
X-squared = 20.581, df = 1, p-value = 5.717e-06

[1] "---"
[1] "Chi-Square test for: Appetite"

        Pearson's Chi-squared test with Yates' continuity correction

data:  data_updated[[col]] and data_updated$classification
X-squared = 59.891, df = 1, p-value = 1.003e-14

[1] "---"
[1] "Chi-Square test for: Pedal.edema"

        Pearson's Chi-squared test with Yates' continuity correction

data:  data_updated[[col]] and data_updated$classification
X-squared = 54.338, df = 1, p-value = 1.688e-13

[1] "---"
[1] "Chi-Square test for: Anemia"

        Pearson's Chi-squared test with Yates' continuity correction

data:  data_updated[[col]] and data_updated$classification
X-squared = 40.492, df = 1, p-value = 1.975e-10

[1] "---"
```

Logistic regression model for all the parameters

```{r}
library(readr)
#data <- data %>%
  #mutate_if(is.numeric, ~if_else(is.na(.), median(., na.rm = TRUE), .)) %>%
  #mutate_if(is.character, ~if_else(is.na(.), Mode(.), .)) # Define Mode function if not defined

# Convert categorical variables to factors
categorical_vars <- c("Red.Blood.Cells", "Pus.cells", "Pus.cell.clumps", "Bacteria", "hypertension",
                      "Diabetes.Mellitus", "Coronary.artery.disease", "Appetite", "Pedal.edema", "Anemia")
data[categorical_vars] <- lapply(data[categorical_vars], as.factor)

# Ensure the target variable is a factor
data$Classification <- factor(data$Classification, levels = c("notckd", "ckd"))

# Fit the logistic regression model
# Including significant continuous and categorical variables identified from Wilcoxon and Chi-square tests
model <- glm(Classification ~ Age + `Blood.Pressure` + Albumin + `Serum.creatinine` + Hemoglobin +
             `Specific.Gravity` + Sugar + `Blood.glucose.random` + `Blood.urea` + Sodium +
             `Packed.cell.volume` + `White.blood.cell.count` + `Red.blood.cell.count` +
             `Red.Blood.Cells` + `Pus.cells` + `Pus.cell.clumps` + Bacteria + hypertension +
             `Diabetes.Mellitus` + `Coronary.artery.disease` + Appetite + `Pedal.edema` + Anemia,
             data = data, family = binomial(link = "logit"))

# Summary of the model
summary(model)
```

```
Call:
glm(formula = Classification ~ Age + Blood.Pressure + Albumin +
    Serum.creatinine + Hemoglobin + Specific.Gravity + Sugar +
    Blood.glucose.random + Blood.urea + Sodium + Packed.cell.volume +
    White.blood.cell.count + Red.blood.cell.count + Red.Blood.Cells +
    Pus.cells + Pus.cell.clumps + Bacteria + hypertension + Diabetes.Mellitus +
    Coronary.artery.disease + Appetite + Pedal.edema + Anemia,
    family = binomial(link = "logit"), data = data)

Coefficients:
                             Estimate Std. Error z value Pr(>|z|)
(Intercept)                 1.028e+04  6.320e+06   0.002    0.999
Age                        -3.820e-01  9.476e+02   0.000    1.000
Blood.Pressure              6.139e-01  1.875e+03   0.000    1.000
Albumin                     7.333e+00  6.559e+04   0.000    1.000
Serum.creatinine            1.379e+00  6.690e+03   0.000    1.000
Hemoglobin                 -1.336e+01  2.168e+04  -0.001    1.000
Specific.Gravity           -9.798e+03  5.901e+06  -0.002    0.999
Sugar                       1.184e+01  2.196e+05   0.000    1.000
Blood.glucose.random        3.529e-01  3.555e+02   0.001    0.999
Blood.urea                  4.869e-01  6.317e+02   0.001    0.999
Sodium                     -9.191e-02  1.738e+03   0.000    1.000
Packed.cell.volume         -3.740e+00  6.028e+03  -0.001    1.000
White.blood.cell.count     -5.008e-03  1.315e+01   0.000    1.000
Red.blood.cell.count       -5.833e-01  2.352e+04   0.000    1.000
Red.Blood.Cellsnormal      -3.845e+01  1.083e+05   0.000    1.000
Pus.cellsnormal            -2.435e+01  8.814e+04   0.000    1.000
Pus.cell.clumpspresent      2.070e+01  2.514e+05   0.000    1.000
Bacteriapresent            -5.142e+01  6.902e+05   0.000    1.000
hypertensionyes             3.414e+01  1.043e+05   0.000    1.000
Diabetes.Mellitusno         4.624e+01  3.814e+05   0.000    1.000
Diabetes.Mellitusyes        8.570e+01  3.682e+05   0.000    1.000
Coronary.artery.diseaseyes -7.651e+01  7.477e+05   0.000    1.000
Appetitepoor                4.456e+01  4.450e+04   0.001    0.999
Pedal.edemayes              4.306e+01  5.613e+04   0.001    0.999
Anemiayes                  -4.422e+01  8.149e+04  -0.001    1.000

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 5.2925e+02  on 399  degrees of freedom
Residual deviance: 7.6890e-08  on 375  degrees of freedom
AIC: 50

Number of Fisher Scoring iterations: 25

Error: attempt to use zero-length variable name
```

VIF to check the multicollinearity of the fitted model

```{r}
# Load necessary libraries
library(car)
library(glmnet)

# Assuming 'data' is your dataset and 'Classification' is the binary outcome variable
# Fit a logistic regression model with all predictors
full_model <- glm(Classification ~ Age + Blood.Pressure + Albumin + Serum.creatinine + Hemoglobin +
                  Specific.Gravity + Sugar + Blood.glucose.random + Blood.urea +
                  Sodium + Packed.cell.volume + White.blood.cell.count +
                  Red.blood.cell.count + hypertension + Diabetes.Mellitus +
                  Coronary.artery.disease + Appetite + Pedal.edema + Anemia,
                  data = data, family = binomial())

# Calculate VIF for the fitted model
vif_model <- vif(full_model)
print(vif_model[vif_model > 5])  # Print VIF values greater than 5
```

```
  [1] 40.986523 31.517504  8.113222 40.321387 24.158248 27.351635 27.041900 11.836509  7.812591  9.712860 35.743718 22.946366  6.280132 58.587265 70.884321 15.160787
 [17]  5.252627  7.171851 13.708853  6.402072  5.614045  6.349912  5.229879  5.200183  5.978605  7.654232
Error: attempt to use zero-length variable name
```

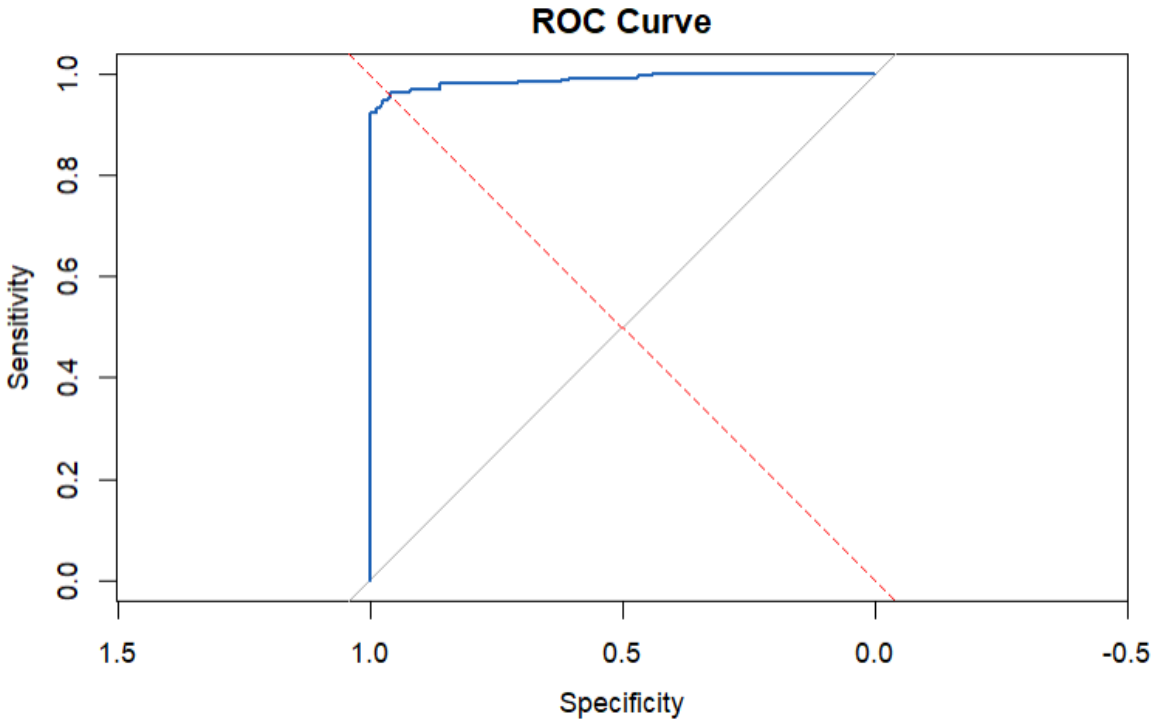
Logistic regression with few parameters

```{r}
# Fit the logistic regression model with selected variables
# Assuming the significant variables identified are 'Hemoglobin', 'Serum.creatinine', 'hypertension'
final_model <- glm(Classification ~ Hemoglobin + Serum.creatinine + Specific.Gravity,
                   data = data, family = binomial())

# Display the summary of the final model
summary(final_model)
```

```
Call:
glm(formula = Classification ~ Hemoglobin + Serum.creatinine +
    Specific.Gravity, family = binomial(), data = data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      717.0023   147.2556   4.869 1.12e-06 ***
Hemoglobin        -1.6232     0.3048  -5.326 1.01e-07 ***
Serum.creatinine   5.3631     1.3472   3.981 6.87e-05 ***
Specific.Gravity -687.3246   143.2408  -4.798 1.60e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 529.25  on 399  degrees of freedom
Residual deviance:  62.01  on 396  degrees of freedom
AIC: 70.01

Number of Fisher Scoring iterations: 11
```

ROC curve and AUC values to evaluate model performance

```r

# Load packages
library(caret)
library(pROC)
library(e1071)

model <- glm(Classification ~ Hemoglobin + Serum.creatinine + hypertension,
             family = binomial(), data = data)
# Predict probabilities
probabilities <- predict(model, type = "response")

# Convert probabilities to binary outcomes based on a threshold of 0.5
predicted_classes <- ifelse(probabilities > 0.5, 1, 0)

# Actual classes
actual_classes <- as.numeric(data$Classification) - 1  # Adjust if necessary based on how your factor levels are coded

# Calculate confusion matrix
conf_matrix <- confusionMatrix(as.factor(predicted_classes), as.factor(actual_classes))
print(conf_matrix)

# Compute ROC curve and AUC
roc_result <- roc(actual_classes, probabilities)
auc_value <- auc(roc_result)
print(auc_value)

# Plot ROC curve
plot(roc_result, main="ROC Curve", col="#1c61b6")
abline(a=0, b=1, col="red", lty=2)  # Diagonal line for reference
```



```
Prediction   0   1
         0 142   9
         1   8 241

               Accuracy : 0.9575
                 95% CI : (0.9328, 0.9751)
    No Information Rate : 0.625
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9095

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9467
            Specificity : 0.9640
         Pos Pred Value : 0.9404
         Neg Pred Value : 0.9679
             Prevalence : 0.3750
         Detection Rate : 0.3550
   Detection Prevalence : 0.3775
      Balanced Accuracy : 0.9553

       'Positive' Class : 0

Setting levels: control = 0, case = 1
Setting direction: controls < cases
Area under the curve: 0.9884
```

Precision, Recall, F1 Score, and Accuracy

```{r}
# Precision, Recall, F1 Score, and Accuracy
precision <- conf_matrix$byClass["Pos Pred Value"]  # Precision (Positive Predictive Value)
recall <- conf_matrix$byClass["Sensitivity"]       # Recall (True Positive Rate)
f1_score <- 2 * (precision * recall) / (precision + recall)  # F1 Score
accuracy <- conf_matrix$overall["Accuracy"]         # Accuracy

# Print the metrics
print(paste("Precision:", precision))
print(paste("Recall:", recall))
print(paste("F1 Score:", f1_score))
print(paste("Accuracy:", accuracy))
```

```
[1] "Precision: 0.940397350993377"
[1] "Recall: 0.946666666666667"
[1] "F1 Score: 0.943521594684385"
[1] "Accuracy: 0.9575"
```