**Load Data:**

**Code:**

install.packages("readxl")

library(readxl)

mainData <- read\_excel("E:/Coding Practice/Data Science/mid\_project/maternal\_\_mod.xlsx")

mainData

**Output:**

A screenshot of a computer screen

Description automatically generated

**Description:** Load Dataset to mainData

**Data Structure:**

**Code:**

num\_instances <- nrow(mainData)

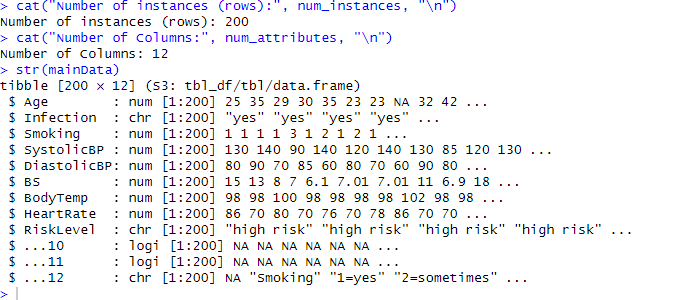
num\_attributes <- ncol(mainData)

cat("Number of instances (rows):", num\_instances, "\n")

cat("Number of Columns:", num\_attributes, "\n")

str(mainData)

**Output:**



**Description:** To see the summary of the structure of data set. All the attributes with their type and their instances are shown here. The total number of instances is 200 and columns are 12. This str() function also shows that not all the columns are attributes. We need to process them further.

**Remove unnecessary columns:**

**Code:**

actualData <- mainData[, -c((num\_attributes - 2):num\_attributes)]

actualData

num\_instances <- nrow(actualData)

num\_attributes <- ncol(actualData)

cat("Number of instances (rows):", num\_instances, "\n")

cat("Number of Attributes:", num\_attributes, "\n")

**Output:**

A screenshot of a computer

Description automatically generated

**Description:** Last three columns were unnecessary, so we needed to remove them.

Number of instances: 200

Number of attributes: 9

Categorial Attributes are Infection and RiskLevel (also class attribute as it’s our target attribute).

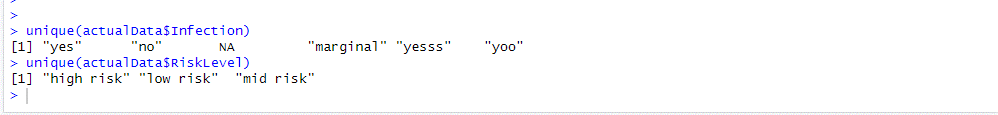
**Unique Categories:**

**Code:**

unique(actualData$Infection)

unique(actualData$RiskLevel)

**Output:**



**Description:** Using unique functions we can find unique attributes. Here in Infection category, there are: "yes" , "no" ,NA ,"marginal" ,"yesss" , "yoo". NA is missing value we will take care of them in massing values section. "yesss","yoo" are invalid values of Infection attribute.

In RiskLevel attribute, there are: "high risk", "low risk", "mid risk". All are perfect and there are no invalid or missing values.

**Recover Invalid values:**

**Code:**

actualData$Infection <- ifelse(substr(tolower(actualData$Infection), 1, 1) == "y", "yes",

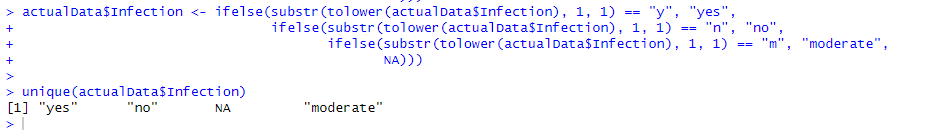
ifelse(substr(tolower(actualData$Infection), 1, 1) == "n", "no",

ifelse(substr(tolower(actualData$Infection), 1, 1) == "m", "moderate",

NA)))

unique(actualData$Infection)

**Output:**



**Description:** Previously some invalid values were found on the Infection attribute. We assumed that if the starting character was y means yes, n means no, m means moderate and ignored further characters of each instance in Infection attribute. And after recovering invalid values, there’s no extra invalid values found using unique() function.

**Annotating Datasets:**

**Code:**

actualData$Infection <- factor(actualData$Infection,

levels = c("yes", "moderate", "no"),

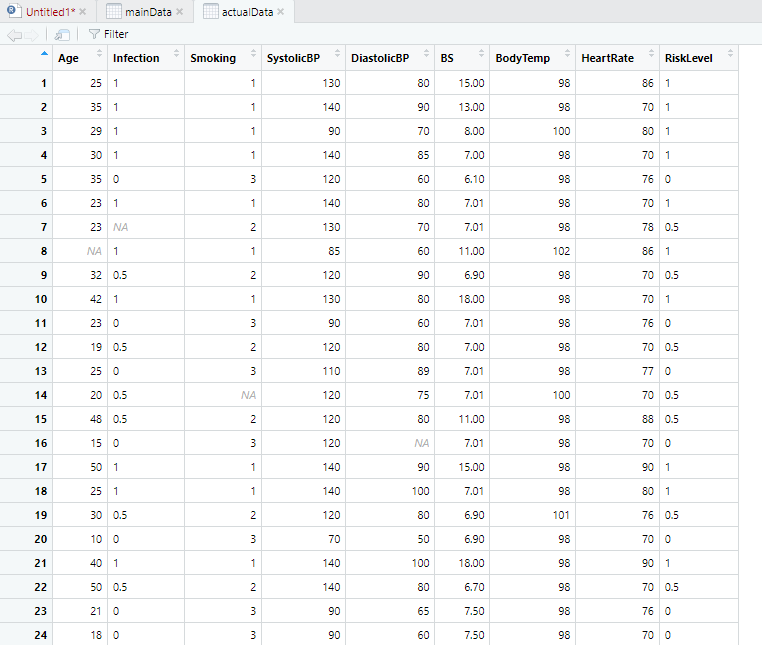
labels = c(1, 0.5, 0))

actualData$RiskLevel <- factor(actualData$RiskLevel,

levels = c("high risk", "low risk", "mid risk"),

labels = c(1, 0, 0.5))

actualData

**Output:**

**Description:** The value of these categorical attributes has been converted to numerical values using factor() function.

For Infection attribute: "yes"→1, "moderate"→0.5, "no"→0

For RiskLevel attribute: "high risk"→1, "mid risk"→0.5, "low risk"→0

**Data Statistics:**

**Code:**

summary(actualData)

**Output:**

A close-up of a number

Description automatically generated

**Description:** Descriptive Statistics has shown using summary() function. This summary will help on our following works.

**Find Missing Values:**

**Code:**

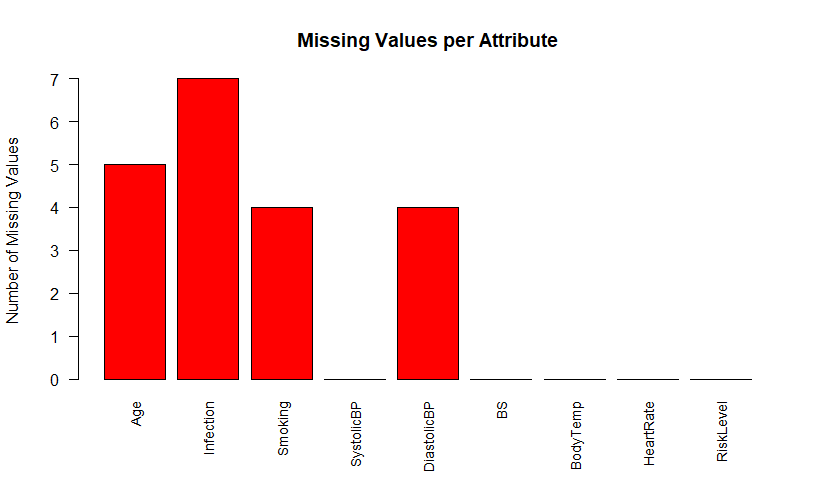
na\_counts <- colSums(is.na(actualData))

print(na\_counts)

**Output:**



**Visualize:**

 barplot(na\_counts, names.arg = names(na\_counts),

ylab = "Number of Missing Values", col = "red",

main = "Missing Values per Attribute", las =2)

**Description:** Using colSums() and is.na() we found the number of missing values for each attribute. The following missing values were found:

Age: 5

Infection: 7

Smoking: 4

SystolicBP: 0

DiastolicBP: 4

BS: 0

BodyTemp: 0

HeartRate: 0

RiskLevel: 0

# **Handle Missing Values:**

## **1. By Discard Instances:**

**Code:**

cleanData1 <- na.omit(actualData)

na\_counts <- colSums(is.na(cleanData1))

print(na\_counts)

barplot(na\_counts, names.arg = names(na\_counts),

ylab = "Number of Missing Values", col = "red",cex.names = 0.9,

main = "Missing Values per Attribute", las =2)

**Output:**

A white paper with black text

Description automatically generatedA black text on a white background

Description automatically generated

**Description:** All the instances having null values have been removed using na.omit() function.

## **2. Replace by Most Frequent/Average Value**

**Code: (for categorical attribute)**

cleanData2 <- actualData

mode\_Infection <- names(sort(table(cleanData2$Infection), decreasing = TRUE))[1]

cleanData2$Infection[is.na(cleanData2$Infection)] <- mode\_Infection

mode\_RiskLevel <- names(sort(table(cleanData2$RiskLevel), decreasing = TRUE))[1]

cleanData2$RiskLevel[is.na(cleanData2$RiskLevel)] <- mode\_RiskLevel

cleanData2

na\_counts <- colSums(is.na(cleanData2))

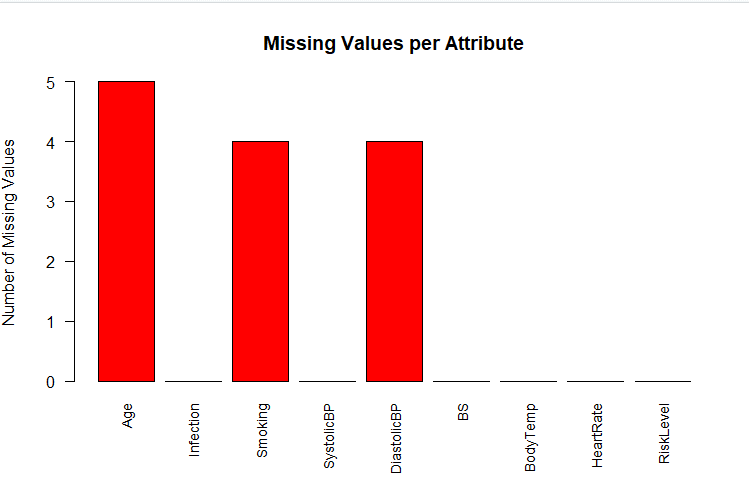
print(na\_counts)

barplot(na\_counts, names.arg = names(na\_counts),

ylab = "Number of Missing Values", col = "red",cex.names = 0.9,

main = "Missing Values per Attribute", las =2)

**Output:**



**Description:** After sorting the values of a column, the most frequent value has been found for each column, replaced with missing values of that column.

**Code: (for numerical attribute)**

for(col\_name in names(cleanData2)) {

if(is.numeric(cleanData2[[col\_name]])) {

column\_mean <- mean(cleanData2[[col\_name]], na.rm = TRUE)

cleanData2[[col\_name]][is.na(cleanData2[[col\_name]])] <- column\_mean

cleanData2[[col\_name]] <- round(cleanData2[[col\_name]], digits = 0)

}

}

na\_counts <- colSums(is.na(cleanData2))

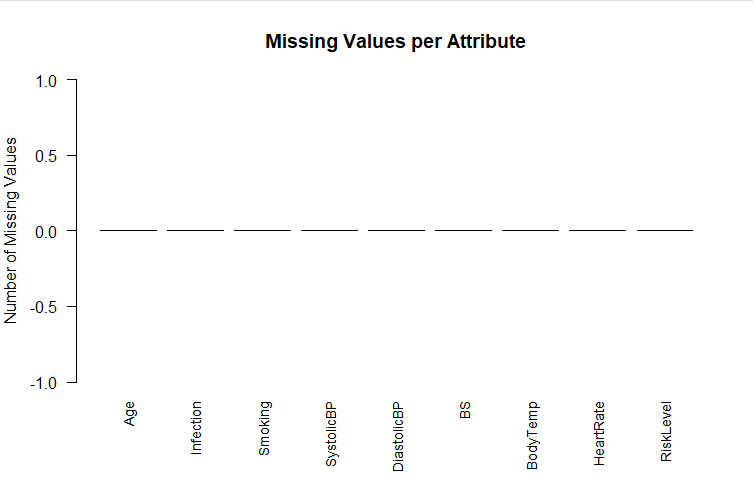
print(na\_counts)

barplot(na\_counts, names.arg = names(na\_counts),

ylab = "Number of Missing Values", col = "red",cex.names = 0.9,

main = "Missing Values per Attribute", las =2)

**Output:**



**Description:** After replacing categorical attributes, the estimated values for each row were founded using the average value. Then after rounded the estimated values, the missing values replaced with them.

# **Mean-Median-Mode Graph:**

**Code:**

getMode <- function(v) {

tabulated <- table(v)

mode\_value <- names(sort(tabulated, decreasing = TRUE))[1]

return(as.numeric(mode\_value))

}

means <- sapply(cleanData2, function(x) if(is.numeric(x)) mean(x, na.rm = TRUE) else NA)

medians <- sapply(cleanData2, function(x) if(is.numeric(x)) median(x, na.rm = TRUE) else NA)

modes <- sapply(cleanData2, function(x) if(is.numeric(x) || is.factor(x) || is.character(x)) getMode(x) else NA)

stat\_values <- rbind(means, medians, modes)

row\_names <- c("Mean", "Median", "Mode")

rownames(stat\_values) <- row\_names

barplot(stat\_values, beside = TRUE,

col = c("green", "orange", "brown"),

legend.text = row\_names, args.legend = list(x = "topright", cex = 0.7),

cex.names = 0.7)

A graph of different colored bars

Description automatically generated**Output:**

**Description:** Initially mean, mode and were found using sapply() function, then the values were combined in stat\_values matrix using rblind(). A Barplot has been drawn to visualize.

# **Find Outliers:**

**Code:**

boxplot(cleanData2$Age, main = "Age")

boxplot(cleanData2$Smoking, main = "Smoking" )

boxplot(cleanData2$SystolicBP, main = "SystolicBP" )

boxplot(cleanData2$DiastolicBP, main = "DiastolicBP" )

boxplot(cleanData2$BS, main = "BS")

boxplot(cleanData2$BodyTemp, main = "BodyTemp" )

boxplot(cleanData2$HeartRate, main = "HeartRate" )

**A diagram of a graph

Description automatically generatedOutput:**

A graph with a bar chart

Description automatically generated with medium confidence

A diagram of a graph

Description automatically generated with medium confidenceA diagram of a graph

Description automatically generated

A square with a black line

Description automatically generated

**A diagram of a graph

Description automatically generated**

**A white rectangular object with black text

Description automatically generated**

**Description:** Using boxplot, we found outliers on Age, BS, BodyTemp

# **Removing Outliers:**

**Code:**

age\_mean <- round(mean(cleanData2$Age, na.rm = TRUE))

age\_outliers <- boxplot.stats(cleanData2$Age)$out

age\_outliers

cleanData2$Age[cleanData2$Age %in% age\_outliers] <- age\_mean

boxplot(cleanData2$Age, main = "Age")

age\_outliers

bs\_mean <- round(mean(cleanData2$BS, na.rm = TRUE))

bs\_outliers <- boxplot.stats(cleanData2$BS)$out

bs\_outliers

cleanData2$BS[cleanData2$BS %in% bs\_outliers] <- bs\_mean

boxplot(cleanData2$BS, main = "BS")

bodytemp\_mean <- round(mean(cleanData2$BodyTemp, na.rm = TRUE))

bodytemp\_outliers <- boxplot.stats(cleanData2$BodyTemp)$out

bodytemp\_outliers

table(cleanData2$BodyTemp)

bodytemp\_outliers <- bodytemp\_outliers[bodytemp\_outliers < 0]

bodytemp\_outliers

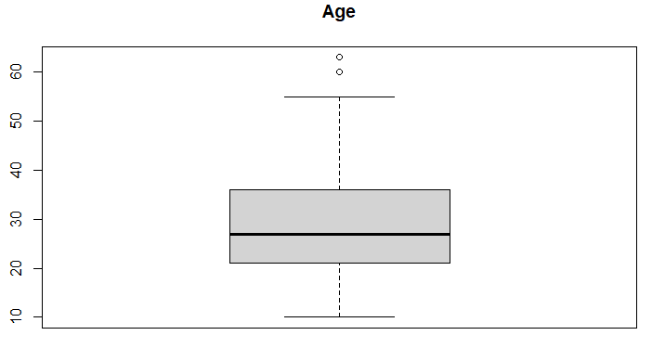
cleanData2$BodyTemp[cleanData2$BodyTemp %in% bodytemp\_outliers] <- bodytemp\_mean

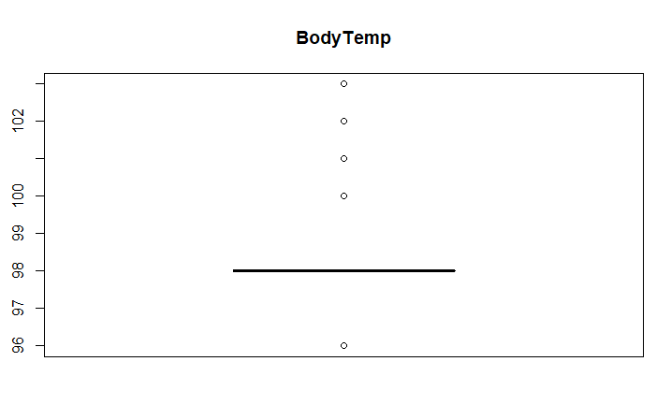
boxplot(cleanData2$BodyTemp, main = "BodyTemp" )

table(cleanData2$BodyTemp)

**A number on a white background

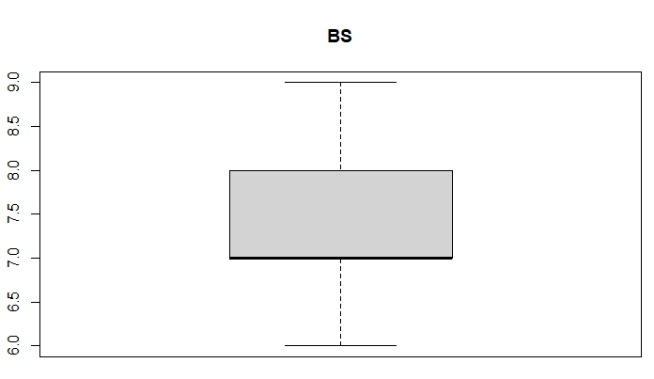
Description automatically generatedOutput:**



****

**A white background with black text

Description automatically generated**



**Description:** The outliers of Age attribute were removed from Age, but while plotting box plot, we found that some new outliers appeared. We ignored them because the values are so near to our data. After converting outliers of BS using mean value, there no outlier remains.

In BodyTemp attribute, we removed negative outliers first, then after considering value counts of the rest values of bodytemp\_outlier , all other values are valid. So, we shouldn’t remove them.

# **Normalize SystolicBP:**

**cleanData2 table:**

A screenshot of a computer

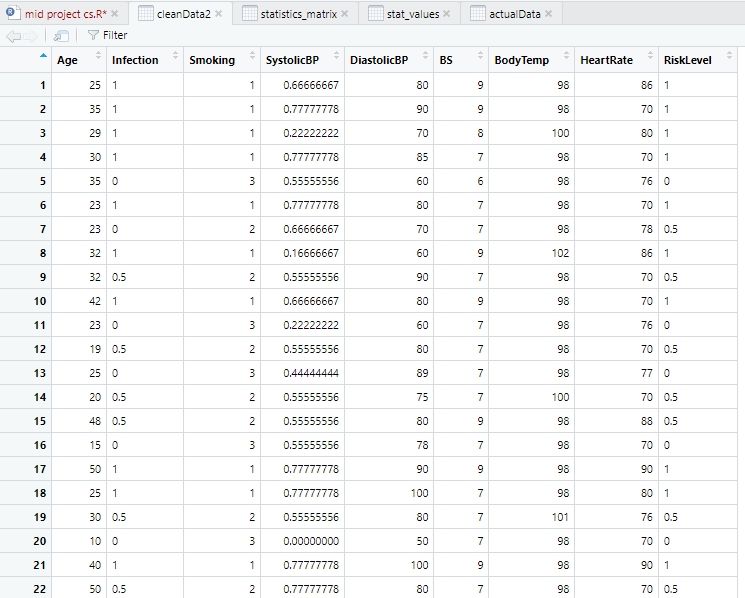
Description automatically generated

**Code:**

min\_SystolicBP <- min(cleanData2$SystolicBP, na.rm = TRUE)

max\_SystolicBP <- max(cleanData2$SystolicBP, na.rm = TRUE)

cleanData2$SystolicBP <- (cleanData2$SystolicBP - min\_SystolicBP) / (max\_SystolicBP - min\_SystolicBP)

**Output:**

**Description:** Normalized using Xnew=(X-Xmin)/(Xmax-Xmin)

# **Convert Numerical Attributes to Categorical:**

**Code:**

outputData<- cleanData2

outputData$Infection <- factor(outputData$Infection,

levels = c(1, 0.5, 0),

labels = c("yes", "moderate", "no"))

outputData$RiskLevel <- factor(outputData$RiskLevel,

labels = c("high risk", "low risk", "mid risk"),

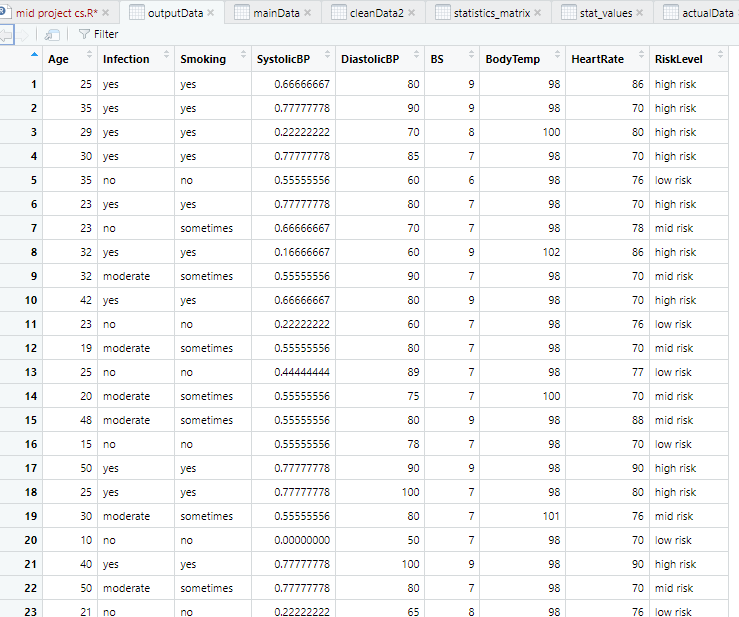
levels = c(1, 0, 0.5))

outputData$Smoking <- factor(outputData$Smoking,

labels = c("yes", "sometimes", "no"),

levels = c(1, 2, 3))

outputData

**Output:**

**Description:** Three categorical attributes shown using factor() function.