Predicting diabetes diagnosis from risk factor data

DA5030 Intro to Machine Learning & Data Mining

Aditya Elayavalli elayavalli.a@northeastern.edu

Fall 2023 Signature Project

November 23, 2023

Goal of the project

The data set was created to understand the relationship between lifestyle and diabetes in the US. The goal of the project is to see what variable feature in the data set is the most common cause for diabetes and then predict if an individual has diabetes or not. The risk factors for diabetes include behaviors that may result in a greater chance of acquiring diabetes such as smoking, drinking alcohol, lack of physical activity and low physical health. Additionally high blood pressure as well as heart diseases are frequent with patients who have diabetes compared to those who dont. Smoking is another leading cause for diabetes as well. Additionally, not smoking, increased physical activity, eating healthier food such as vegetables and fruits decreases the risk for having diabetes. Because the data set does not have any missing values, we will be duplicating the data set which will have missing values and will see how that affects the prediction.

Data Preperation

The data set was created to understand the relationship between lifestyle and diabetes in the US. The goal of the project is to see what variable feature in the data set is the most common cause for diabetes and then predict if an individual has diabetes or not. The risk factors for diabetes include behaviors that may result in a greater chance of acquiring diabetes such as smoking, drinking alcohol, lack of physical activity and low physical health. Additionally high blood pressure as well as heart diseases are frequent with patients who have diabetes compared to those who dont. Smoking is another leading cause for diabetes as well. Additionally, not smoking, increased physical activity, eating healthier food such as vegetables and fruits decreases the risk for having diabetes.

The below code chunk will read in the data set. The data set was obtained from Kaggle. Below is the link

https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset

```
# Load necessary library
library(ggplot2)
library(ggplot2)
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.3

## corrplot 0.94 loaded
```

```
# Install factoextra package if not already installed
if (!requireNamespace("factoextra", quietly = TRUE)) {
    install.packages("factoextra")
}
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
# Ensure the caret package is installed and loaded
if (!requireNamespace("caret", quietly = TRUE)) {
    install.packages("caret")
}
library(caret)
```

```
## Loading required package: lattice
```

```r
library(klaR)
```

```
## Loading required package: MASS
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(gmodels)
library(gmodels)
library(C50)
library(ipred)
```

```
## Warning: package 'ipred' was built under R version 4.3.3
```

```r
url <- "https://drive.google.com/uc?export=download&id=1AP8R137JV74PBDoyfWhPq_hFeez1oZPv"
raw_data <- read.csv(url, stringsAsFactors = TRUE, header = TRUE)
```

Now we will explore the dataset to get an idea of the data

```r
head(raw_data,10)
```

```
##    Diabetes_012 HighBP HighChol CholCheck BMI Smoker Stroke
## 1             0      1        1         1  40      1      0
## 2             0      0        0         0  25      1      0
## 3             0      1        1         1  28      0      0
## 4             0      1        0         1  27      0      0
## 5             0      1        1         1  24      0      0
## 6             0      1        1         1  25      1      0
## 7             0      1        0         1  30      1      0
## 8             0      1        1         1  25      1      0
## 9             2      1        1         1  30      1      0
## 10            0      0        0         1  24      0      0
##    HeartDiseaseorAttack PhysActivity Fruits Veggies HvyAlcoholConsump
## 1                     0            0      0       1                 0
## 2                     0            1      0       0                 0
## 3                     0            0      1       0                 0
## 4                     0            1      1       1                 0
## 5                     0            1      1       1                 0
## 6                     0            1      1       1                 0
## 7                     0            0      0       0                 0
## 8                     0            1      0       1                 0
## 9                     1            0      1       1                 0
## 10                    0            0      0       1                 0
##    AnyHealthcare NoDocbcCost GenHlth MentHlth PhysHlth DiffWalk Sex Age
## 1              1           0       5       18       15        1   0   9
## 2              0           1       3        0        0        0   0   7
## 3              1           1       5       30       30        1   0   9
## 4              1           0       2        0        0        0   0  11
## 5              1           0       2        3        0        0   0  11
## 6              1           0       2        0        2        0   1  10
## 7              1           0       3        0       14        0   0   9
## 8              1           0       3        0        0        1   0  11
## 9              1           0       5       30       30        1   0   9
## 10             1           0       2        0        0        0   1   8
##    Education Income
## 1          4      3
## 2          6      1
## 3          4      8
## 4          3      6
## 5          5      4
## 6          6      8
## 7          6      7
## 8          4      4
## 9          5      1
## 10         4      3
```

```r
str(raw_data)
```

```
## 'data.frame':    253680 obs. of  22 variables:
##  $ Diabetes_012        : num  0 0 0 0 0 0 0 0 2 0 ...
##  $ HighBP              : num  1 0 1 1 1 1 1 1 1 0 ...
##  $ HighChol            : num  1 0 1 0 1 1 0 1 1 0 ...
##  $ CholCheck           : num  1 0 1 1 1 1 1 1 1 1 ...
##  $ BMI                 : num  40 25 28 27 24 25 30 25 30 24 ...
##  $ Smoker              : num  1 1 0 0 0 1 1 1 1 0 ...
##  $ Stroke              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ HeartDiseaseorAttack: num  0 0 0 0 0 0 0 0 1 0 ...
##  $ PhysActivity        : num  0 1 0 1 1 1 0 1 0 0 ...
##  $ Fruits              : num  0 0 1 1 1 1 0 0 1 0 ...
##  $ Veggies             : num  1 0 0 1 1 1 0 1 1 1 ...
##  $ HvyAlcoholConsump   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AnyHealthcare       : num  1 0 1 1 1 1 1 1 1 1 ...
##  $ NoDocbcCost         : num  0 1 1 0 0 0 0 0 0 0 ...
##  $ GenHlth             : num  5 3 5 2 2 2 3 3 5 2 ...
##  $ MentHlth            : num  18 0 30 0 3 0 0 0 30 0 ...
##  $ PhysHlth            : num  15 0 30 0 0 2 14 0 30 0 ...
##  $ DiffWalk            : num  1 0 1 0 0 0 0 1 1 0 ...
##  $ Sex                 : num  0 0 0 0 0 1 0 0 0 1 ...
##  $ Age                 : num  9 7 9 11 11 10 9 11 9 8 ...
##  $ Education           : num  4 6 4 3 5 6 6 4 5 4 ...
##  $ Income              : num  3 1 8 6 4 8 7 4 1 3 ...
```

From the data set, we can see that all the features in the data set are either binary or integer values. This will be important later when we get to the classification model building. Additionally there are 253680rows and 22 columns.

```r
anyNA(raw_data)
```

```
## [1] FALSE
```

```r
#Based on the codejunk above there does not appear to be any missing values.
```

There does not appear to be any missing values

Now we check to see what features contain continous data

```r
summary(raw_data)
```

```
##   Diabetes_012        HighBP         HighChol        CholCheck
##  Min.   :0.0000   Min.   :0.000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:1.0000
```

```
##  Median :0.0000   Median :0.000   Median :0.0000   Median :1.0000
##  Mean   :0.2969   Mean   :0.429   Mean   :0.4241   Mean   :0.9627
##  3rd Qu.:0.0000   3rd Qu.:1.000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :2.0000   Max.   :1.000   Max.   :1.0000   Max.   :1.0000
##       BMI            Smoker           Stroke        HeartDiseaseorAttack
##  Min.   :12.00   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:24.00   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :27.00   Median :0.0000   Median :0.00000   Median :0.00000
##  Mean   :28.38   Mean   :0.4432   Mean   :0.04057   Mean   :0.09419
##  3rd Qu.:31.00   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :98.00   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000
##   PhysActivity        Fruits           Veggies        HvyAlcoholConsump
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:0.0000
##  Median :1.0000   Median :1.0000   Median :1.0000   Median :0.0000
##  Mean   :0.7565   Mean   :0.6343   Mean   :0.8114   Mean   :0.0562
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  AnyHealthcare     NoDocbcCost          GenHlth          MentHlth
##  Min.   :0.0000   Min.   :0.00000   Min.   :1.000   Min.   : 0.000
##  1st Qu.:1.0000   1st Qu.:0.00000   1st Qu.:2.000   1st Qu.: 0.000
##  Median :1.0000   Median :0.00000   Median :2.000   Median : 0.000
##  Mean   :0.9511   Mean   :0.08418   Mean   :2.511   Mean   : 3.185
##  3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:3.000   3rd Qu.: 2.000
##  Max.   :1.0000   Max.   :1.00000   Max.   :5.000   Max.   :30.000
##     PhysHlth         DiffWalk          Sex              Age
##  Min.   : 0.000   Min.   :0.0000   Min.   :0.0000   Min.   : 1.000
##  1st Qu.: 0.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 6.000
##  Median : 0.000   Median :0.0000   Median :0.0000   Median : 8.000
##  Mean   : 4.242   Mean   :0.1682   Mean   :0.4403   Mean   : 8.032
##  3rd Qu.: 3.000   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:10.000
##  Max.   :30.000   Max.   :1.0000   Max.   :1.0000   Max.   :13.000
##    Education        Income
##  Min.   :1.00   Min.   :1.000
##  1st Qu.:4.00   1st Qu.:5.000
##  Median :5.00   Median :7.000
##  Mean   :5.05   Mean   :6.054
##  3rd Qu.:6.00   3rd Qu.:8.000
##  Max.   :6.00   Max.   :8.000
```

Data Exploration

Feature engineering

Prediabetes is a serious health condition where blood sugar levels are higher than normal, but not high enough yet to be diagnosed as type 2 diabetes. However for the sake of this data set, we are going to combine both prediabetic and diabetic patients as one group. We can use the ifelse funciton to do so. Additionally we want to convert all the binary data into categorical data for this section of the project. Additionally, features like veggies and fruits, AnyHealthcare and NoDocbcCost can be combined into one column.

```r
raw_data$Diabetes_012 <- ifelse(raw_data$Diabetes_012 == 1 | raw_data$Diabetes_012 == 2, 1,
raw_data$FruitsOrVeggies <- as.numeric((raw_data$Fruits == 1) | (raw_data$Veggies == 1))
raw_data$AnyHealthcareorNoDocbcCost <-  as.numeric((raw_data$AnyHealthcare == 1) | (raw_dat
raw_data$Fruits <- NULL
raw_data$Veggies <- NULL
raw_data$AnyHealthcare <- NULL
raw_data$NoDocbcCost <- NULL


#print the structure of the dataframe to verify the changes
str(raw_data)
```

```
## 'data.frame':    253680 obs. of  20 variables:
##  $ Diabetes_012              : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ HighBP                    : num  1 0 1 1 1 1 1 1 1 0 ...
##  $ HighChol                  : num  1 0 1 0 1 1 0 1 1 0 ...
##  $ CholCheck                 : num  1 0 1 1 1 1 1 1 1 1 ...
##  $ BMI                       : num  40 25 28 27 24 25 30 25 30 24 ...
##  $ Smoker                    : num  1 1 0 0 0 1 1 1 1 0 ...
##  $ Stroke                    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ HeartDiseaseorAttack      : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ PhysActivity              : num  0 1 0 1 1 1 0 1 0 0 ...
##  $ HvyAlcoholConsump         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ GenHlth                   : num  5 3 5 2 2 2 3 3 5 2 ...
##  $ MentHlth                  : num  18 0 30 0 3 0 0 0 30 0 ...
##  $ PhysHlth                  : num  15 0 30 0 0 2 14 0 30 0 ...
##  $ DiffWalk                  : num  1 0 1 0 0 0 0 1 1 0 ...
##  $ Sex                       : num  0 0 0 0 0 1 0 0 0 1 ...
##  $ Age                       : num  9 7 9 11 11 10 9 11 9 8 ...
##  $ Education                 : num  4 6 4 3 5 6 6 4 5 4 ...
##  $ Income                    : num  3 1 8 6 4 8 7 4 1 3 ...
##  $ FruitsOrVeggies           : num  1 0 1 1 1 1 0 1 1 1 ...
##  $ AnyHealthcareorNoDocbcCost: num  1 1 1 1 1 1 1 1 1 1 ...
```

```r
head(raw_data,10)
```

```
##    Diabetes_012 HighBP HighChol CholCheck BMI Smoker Stroke
## 1             0      1        1         1  40      1      0
## 2             0      0        0         0  25      1      0
## 3             0      1        1         1  28      0      0
## 4             0      1        0         1  27      0      0
```

```
## 5             0      1        1          1  24       0      0
## 6             0      1        1          1  25       1      0
## 7             0      1        0          1  30       1      0
## 8             0      1        1          1  25       1      0
## 9             1      1        1          1  30       1      0
## 10            0      0        0          1  24       0      0
##    HeartDiseaseorAttack PhysActivity HvyAlcoholConsump GenHlth MentHlth
## 1                     0            0                 0       5       18
## 2                     0            1                 0       3        0
## 3                     0            0                 0       5       30
## 4                     0            1                 0       2        0
## 5                     0            1                 0       2        3
## 6                     0            1                 0       2        0
## 7                     0            0                 0       3        0
## 8                     0            1                 0       3        0
## 9                     1            0                 0       5       30
## 10                    0            0                 0       2        0
##    PhysHlth DiffWalk Sex Age Education Income FruitsOrVeggies
## 1        15        1   0   9         4      3               1
## 2         0        0   0   7         6      1               0
## 3        30        1   0   9         4      8               1
## 4         0        0   0  11         3      6               1
## 5         0        0   0  11         5      4               1
## 6         2        0   1  10         6      8               1
## 7        14        0   0   9         6      7               0
## 8         0        1   0  11         4      4               1
## 9        30        1   0   9         5      1               1
## 10        0        0   1   8         4      3               1
##    AnyHealthcareorNoDocbcCost
## 1                          1
## 2                          1
## 3                          1
## 4                          1
## 5                          1
## 6                          1
## 7                          1
## 8                          1
## 9                          1
## 10                         1
```

```r
raw_data1 <- raw_data
raw_data2 <- raw_data
```

```
## 'data.frame':    253680 obs. of  20 variables:
##  $ Diabetes_012             : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ HighBP                   : num  1 0 1 1 1 1 1 1 1 0 ...
##  $ HighChol                 : num  1 0 1 0 1 1 0 1 1 0 ...
```

```
##  $ CholCheck                : num  1 0 1 1 1 1 1 1 1 1 ...
##  $ BMI                      : num  40 25 28 27 24 25 30 25 30 24 ...
##  $ Smoker                   : num  1 1 0 0 0 1 1 1 1 0 ...
##  $ Stroke                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ HeartDiseaseorAttack     : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ PhysActivity             : num  0 1 0 1 1 1 0 1 0 0 ...
##  $ HvyAlcoholConsump        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ GenHlth                  : num  5 3 5 2 2 2 3 3 5 2 ...
##  $ MentHlth                 : num  18 0 30 0 3 0 0 0 30 0 ...
##  $ PhysHlth                 : num  15 0 30 0 0 2 14 0 30 0 ...
##  $ DiffWalk                 : num  1 0 1 0 0 0 0 1 1 0 ...
##  $ Sex                      : num  0 0 0 0 0 1 0 0 0 1 ...
##  $ Age                      : num  9 7 9 11 11 10 9 11 9 8 ...
##  $ Education                : num  4 6 4 3 5 6 6 4 5 4 ...
##  $ Income                   : num  3 1 8 6 4 8 7 4 1 3 ...
##  $ FruitsOrVeggies          : num  1 0 1 1 1 1 0 1 1 1 ...
##  $ AnyHealthcareorNoDocbcCost: num  1 1 1 1 1 1 1 1 1 1 ...

## 'data.frame':    253680 obs. of  20 variables:
##  $ Diabetes_012             : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ HighBP                   : num  1 0 1 1 1 1 1 1 1 0 ...
##  $ HighChol                 : num  1 0 1 0 1 1 0 1 1 0 ...
##  $ CholCheck                : num  1 0 1 1 1 1 1 1 1 1 ...
##  $ BMI                      : num  40 25 28 27 24 25 30 25 30 24 ...
##  $ Smoker                   : num  1 1 0 0 0 1 1 1 1 0 ...
##  $ Stroke                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ HeartDiseaseorAttack     : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ PhysActivity             : num  0 1 0 1 1 1 0 1 0 0 ...
##  $ HvyAlcoholConsump        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ GenHlth                  : num  5 3 5 2 2 2 3 3 5 2 ...
##  $ MentHlth                 : num  18 0 30 0 3 0 0 0 30 0 ...
##  $ PhysHlth                 : num  15 0 30 0 0 2 14 0 30 0 ...
##  $ DiffWalk                 : num  1 0 1 0 0 0 0 1 1 0 ...
##  $ Sex                      : num  0 0 0 0 0 1 0 0 0 1 ...
##  $ Age                      : num  9 7 9 11 11 10 9 11 9 8 ...
##  $ Education                : num  4 6 4 3 5 6 6 4 5 4 ...
##  $ Income                   : num  3 1 8 6 4 8 7 4 1 3 ...
##  $ FruitsOrVeggies          : num  1 0 1 1 1 1 0 1 1 1 ...
##  $ AnyHealthcareorNoDocbcCost: num  1 1 1 1 1 1 1 1 1 1 ...
```

Missing values

Before we proceed, I would like to mention we will be creating raw_data 1 and raw_data2 from raw_data. This is because of the project criteria requiring imputing of missing values into the data frame. We will be comparing both the data sets to test the fitness of our algorithm and see if there is any difference in the prediction if we were to contain any missing values.

introducing missing values into the data frame raw_data2

```r
set.seed(123)  # For reproducibility

# Loop through each column
for(column_name in names(raw_data2)) {
  # Skip the Diabetes_012 column and ensure the column is numeric before introducing NAs
  if(column_name != "Diabetes_012" && is.numeric(raw_data2[[column_name]])) {
    # Randomly select indices to set as NA
    na_indices <- sample(1:nrow(raw_data2), size = 0.35 * nrow(raw_data2))

    # Introduce NA values in the column
    raw_data2[na_indices, column_name] <- NA
  }
}

head(raw_data2)
```

```
##   Diabetes_012 HighBP HighChol CholCheck BMI Smoker Stroke HeartDiseaseorAttack
## 1            0      1       NA         1  40      1      0                   NA
## 2            0      0        0         0  25      1     NA                   NA
## 3            0      1       NA         1  NA     NA     NA                    0
## 4            0      1        0        NA  NA      0      0                    0
## 5            0      1        1        NA  24      0      0                    0
## 6            0      1        1         1  NA      1     NA                    0
##   PhysActivity HvyAlcoholConsump GenHlth MentHlth PhysHlth DiffWalk Sex Age
## 1            0                NA       5       18       15        1  NA   9
## 2            1                NA      NA       NA       NA        0  NA   7
## 3            0                 0       5       NA       NA       NA   0   9
## 4            1                NA       2       NA        0        0  NA  11
## 5            1                 0       2        3       NA       NA   0  11
## 6           NA                NA       2        0       NA        0  NA  10
##   Education Income FruitsOrVeggies AnyHealthcareorNoDocbcCost
## 1        NA      3               1                          1
## 2         6     NA               0                          1
## 3         4     NA              NA                          1
## 4        NA      6               1                          1
## 5        NA      4               1                          1
## 6         6     NA               1                         NA
```

```r
anyNA(raw_data2)
```

```
## [1] TRUE
```

To deal with missing values, we can impute by using the median

```r
# Loop through each column in raw_data2
for(column_name in names(raw_data2)) {
  # Check if the column is numeric
  if(is.numeric(raw_data2[[column_name]])) {
    # Calculate the median of the column, excluding NA values
    column_median <- median(raw_data2[[column_name]], na.rm = TRUE)

    # Replace NA values with the column median
    raw_data2[[column_name]][is.na(raw_data2[[column_name]])] <- column_median
  }
}

# Optionally, print the first few rows of the dataframe to verify the changes
head(raw_data2)
```

```
##   Diabetes_012 HighBP HighChol CholCheck BMI Smoker Stroke HeartDiseaseorAttack
## 1            0      1        0         1  40      1      0                    0
## 2            0      0        0         0  25      1      0                    0
## 3            0      1        0         1  27      0      0                    0
## 4            0      1        0         1  27      0      0                    0
## 5            0      1        1         1  24      0      0                    0
## 6            0      1        1         1  27      1      0                    0
##   PhysActivity HvyAlcoholConsump GenHlth MentHlth PhysHlth DiffWalk Sex Age
## 1            0                 0       5       18       15        1   0   9
## 2            1                 0       2        0        0        0   0   7
## 3            0                 0       5        0        0        0   0   9
## 4            1                 0       2        0        0        0   0  11
## 5            1                 0       2        3        0        0   0  11
## 6            1                 0       2        0        0        0   0  10
##   Education Income FruitsOrVeggies AnyHealthcareorNoDocbcCost
## 1         5      3               1                          1
## 2         6      7               0                          1
## 3         4      7               1                          1
## 4         5      6               1                          1
## 5         5      4               1                          1
## 6         6      7               1                          1
```

```r
anyNA(raw_data2)
```

```
## [1] FALSE
```

We can see that the missing values are no longer present in the data frame

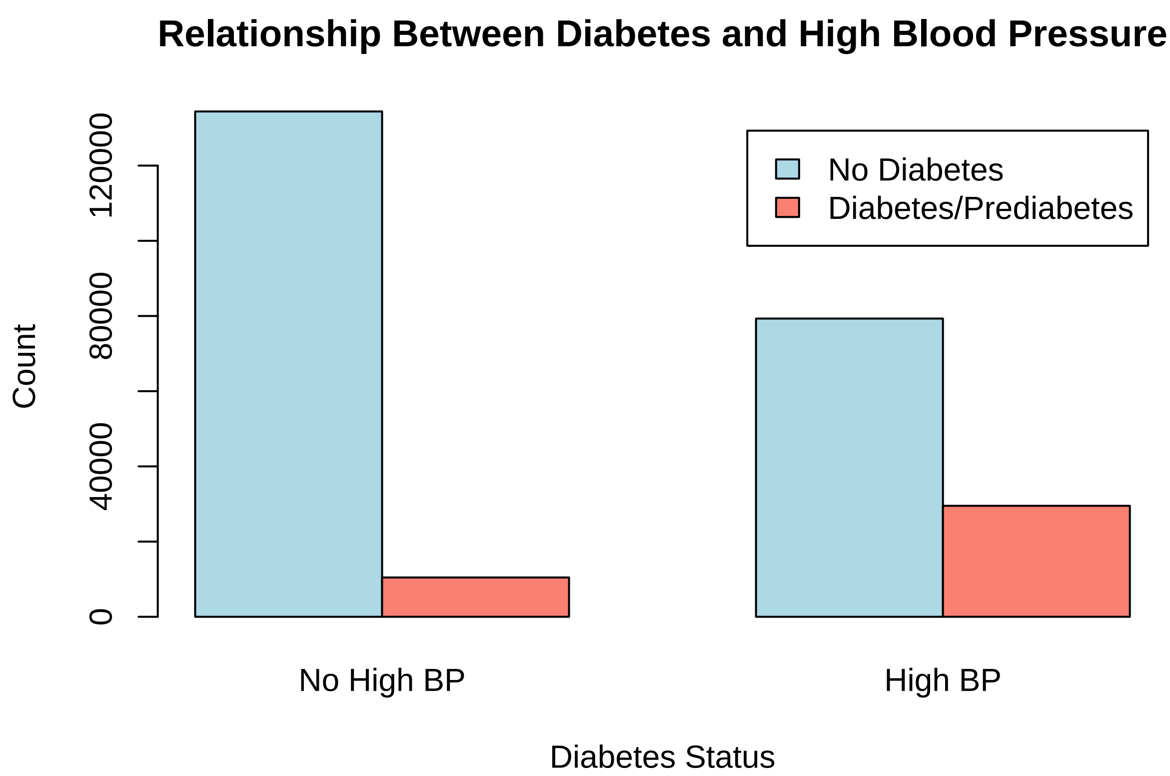
```
## 'data.frame':    253680 obs. of  20 variables:
##  $ Diabetes_012              : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 2 1 ...
##  $ HighBP                    : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 2 2 2 1 ...
```

```
##  $ HighChol                   : Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 1 2 2 1 ...
##  $ CholCheck                  : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 2 2 2 2 ...
##  $ BMI                        : num  40 25 28 27 24 25 30 25 30 24 ...
##  $ Smoker                     : Factor w/ 2 levels "No","Yes": 2 2 1 1 1 2 2 2 2 1 ...
##  $ Stroke                     : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ HeartDiseaseorAttack       : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 2 1 ...
##  $ PhysActivity               : Factor w/ 2 levels "No","Yes": 1 2 1 2 2 2 1 2 1 1 ...
##  $ HvyAlcoholConsump          : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ GenHlth                    : num  5 3 5 2 2 2 3 3 5 2 ...
##  $ MentHlth                   : num  18 0 30 0 3 0 0 0 30 0 ...
##  $ PhysHlth                   : num  15 0 30 0 0 2 14 0 30 0 ...
##  $ DiffWalk                   : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 1 2 2 1 ...
##  $ Sex                        : Factor w/ 2 levels "Male","Female": 1 1 1 1 1 2 1 1 1 2 .
##  $ Age                        : num  9 7 9 11 11 10 9 11 9 8 ...
##  $ Education                  : num  4 6 4 3 5 6 6 4 5 4 ...
##  $ Income                     : num  3 1 8 6 4 8 7 4 1 3 ...
##  $ FruitsOrVeggies            : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 1 2 2 2 ...
##  $ AnyHealthcareorNoDocbcCost: Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...

## 'data.frame':    253680 obs. of  20 variables:
##  $ Diabetes_012               : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 2 1 ...
##  $ HighBP                     : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 2 2 2 1 ...
##  $ HighChol                   : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 1 1 1 1 ...
##  $ CholCheck                  : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 2 2 2 2 ...
##  $ BMI                        : num  40 25 27 27 24 27 30 25 27 24 ...
##  $ Smoker                     : Factor w/ 2 levels "No","Yes": 2 2 1 1 1 2 2 2 2 1 ...
##  $ Stroke                     : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ HeartDiseaseorAttack       : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 2 1 ...
##  $ PhysActivity               : Factor w/ 2 levels "No","Yes": 1 2 1 2 2 2 1 2 1 2 ...
##  $ HvyAlcoholConsump          : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ GenHlth                    : num  5 2 5 2 2 2 3 3 5 2 ...
##  $ MentHlth                   : num  18 0 0 0 3 0 0 0 30 0 ...
##  $ PhysHlth                   : num  15 0 0 0 0 0 0 0 0 0 ...
##  $ DiffWalk                   : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 2 1 ...
##  $ Sex                        : Factor w/ 2 levels "Male","Female": 1 1 1 1 1 1 1 1 1 2 .
##  $ Age                        : num  9 7 9 11 11 10 9 11 8 8 ...
##  $ Education                  : num  5 6 4 5 5 6 5 4 5 4 ...
##  $ Income                     : num  3 7 7 6 4 7 7 7 1 7 ...
##  $ FruitsOrVeggies            : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 1 2 2 2 ...
##  $ AnyHealthcareorNoDocbcCost: Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
```

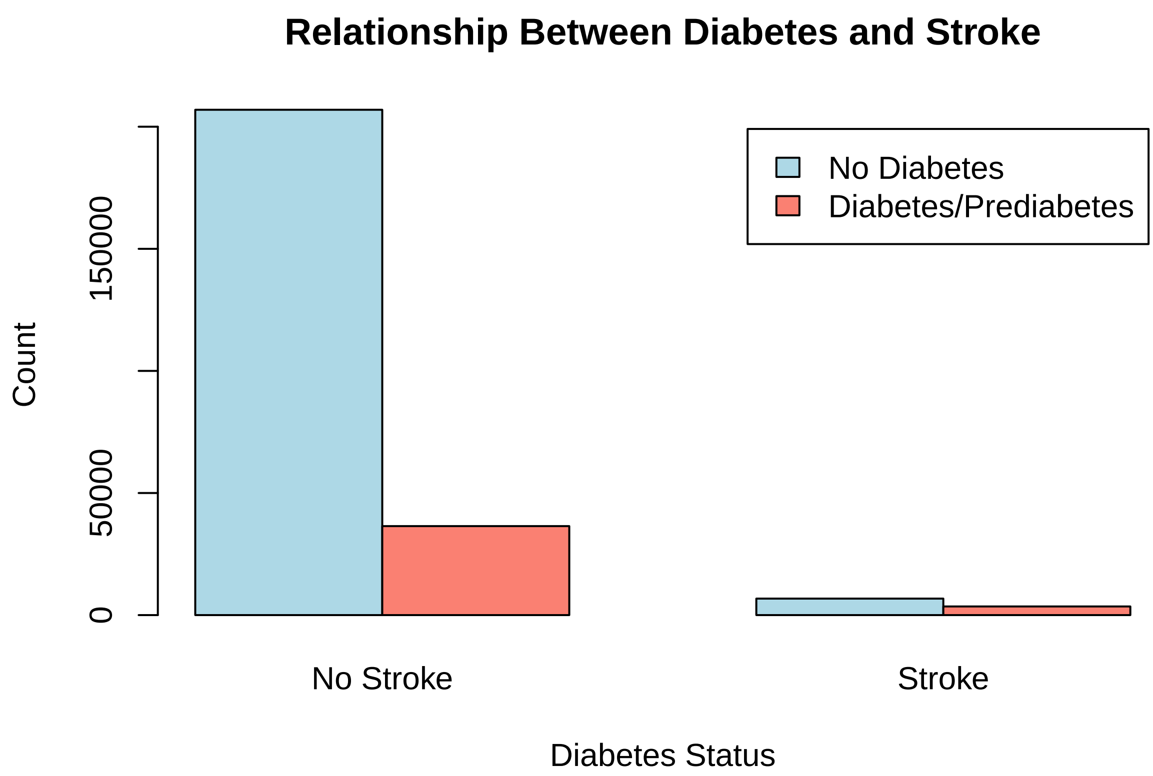
Explatory data plots

```
# Summarize the data
diabetes_bp_table <- table(raw_data$Diabetes_012, raw_data$HighBP)
colnames(diabetes_bp_table) <- c("No High BP", "High BP")
```

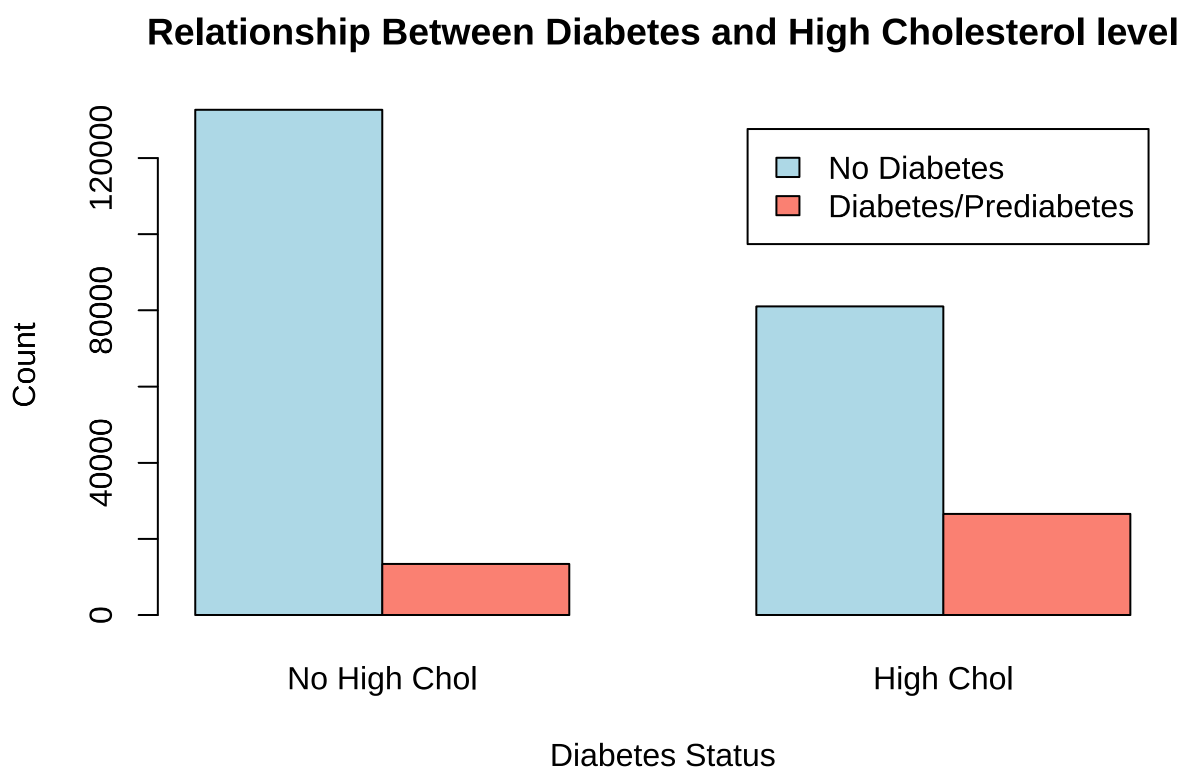
```r
rownames(diabetes_bp_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_bp_table, beside = TRUE,
        main = "Relationship Between Diabetes and High Blood Pressure",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_bp_table))
```



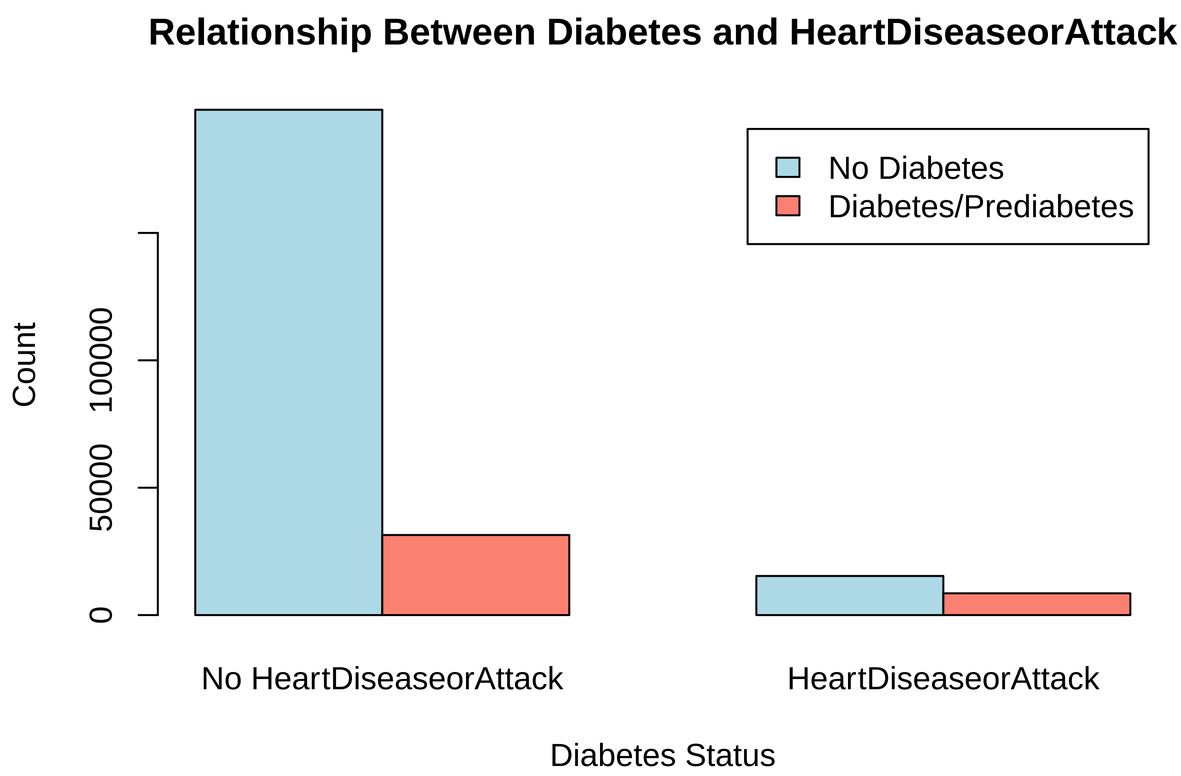

```r
# Summarize the data
diabetes_hc_table <- table(raw_data$Diabetes_012, raw_data$Stroke)
colnames(diabetes_hc_table) <- c("No Stroke", "Stroke")
rownames(diabetes_hc_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_hc_table, beside = TRUE,
        main = "Relationship Between Diabetes and Stroke",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_hc_table))
```



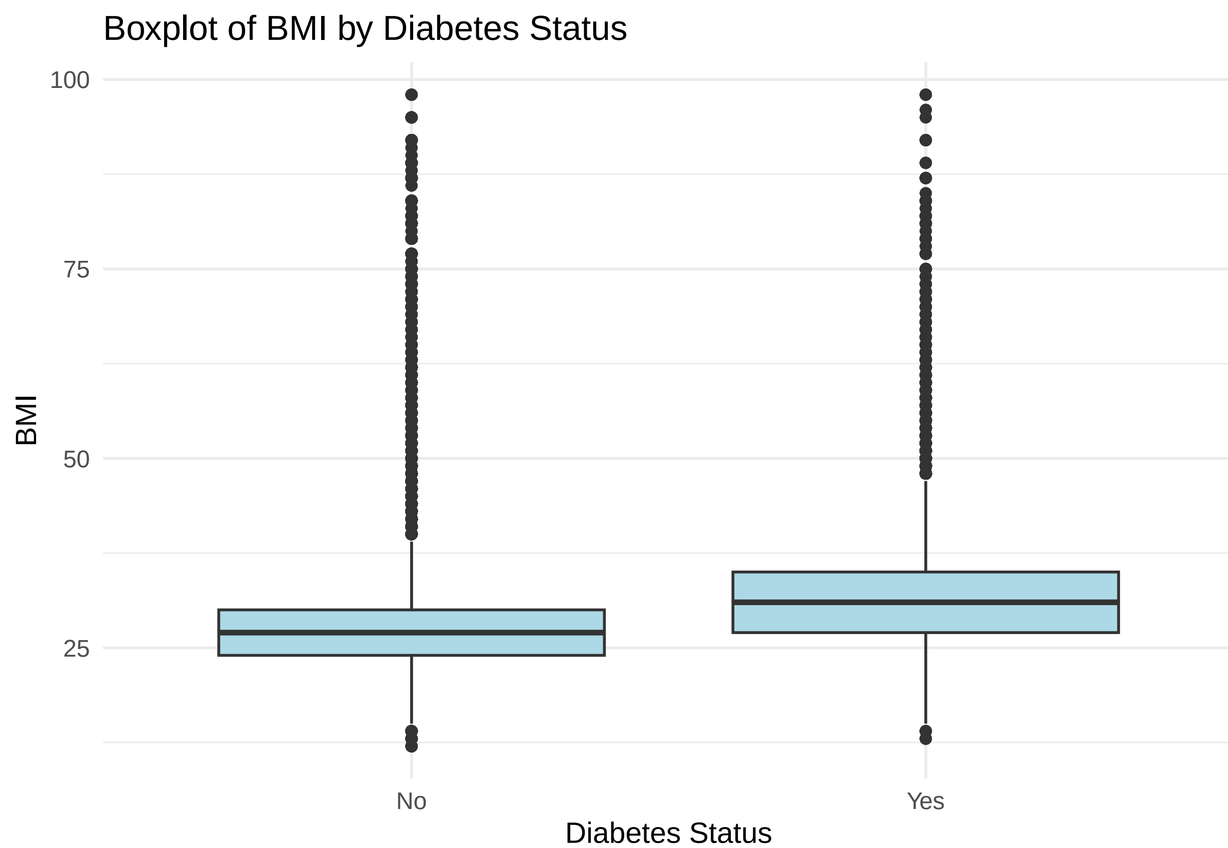

```r
# Summarize the data
diabetes_hc_table <- table(raw_data$Diabetes_012, raw_data$HighChol)
colnames(diabetes_hc_table) <- c("No High Chol", "High Chol")
rownames(diabetes_hc_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_hc_table, beside = TRUE,
        main = "Relationship Between Diabetes and High Cholesterol level",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_hc_table))
```



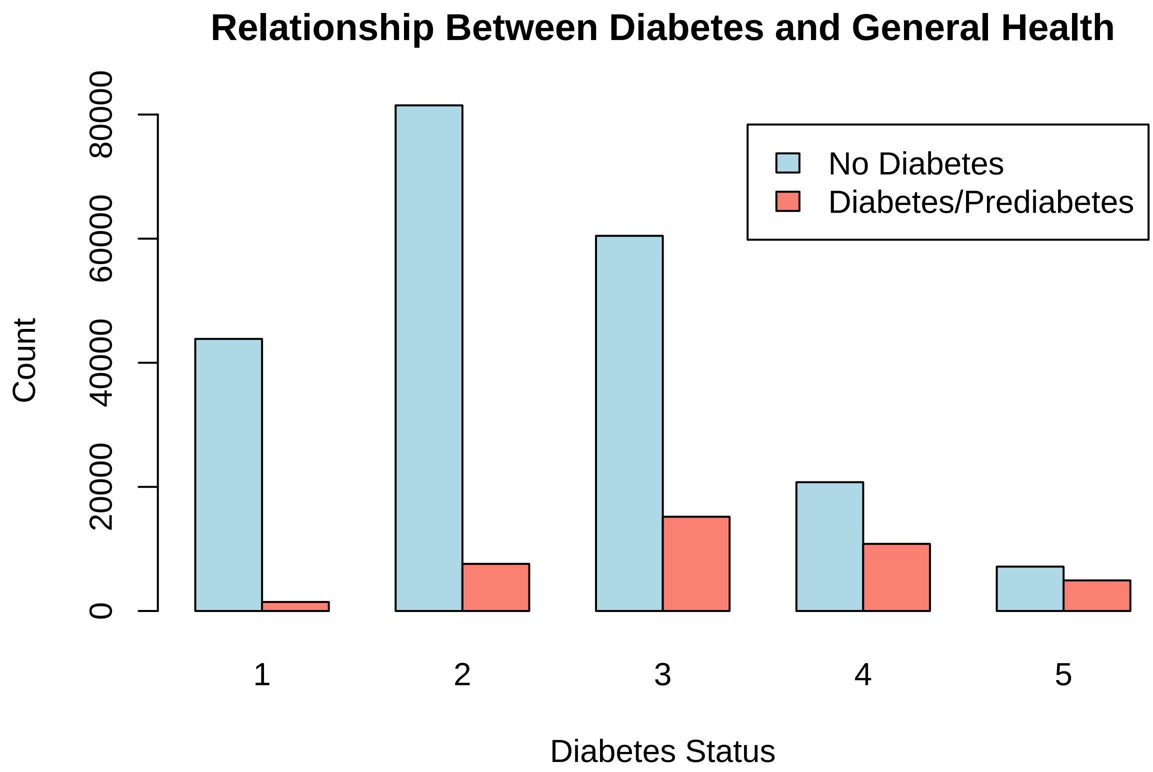

```r
# Summarize the data
diabetes_heart_table <- table(raw_data$Diabetes_012, raw_data$HeartDiseaseorAttack)
colnames(diabetes_heart_table) <- c("No HeartDiseaseorAttack", "HeartDiseaseorAttack")
rownames(diabetes_heart_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_heart_table, beside = TRUE,
        main = "Relationship Between Diabetes and HeartDiseaseorAttack",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_heart_table))
```



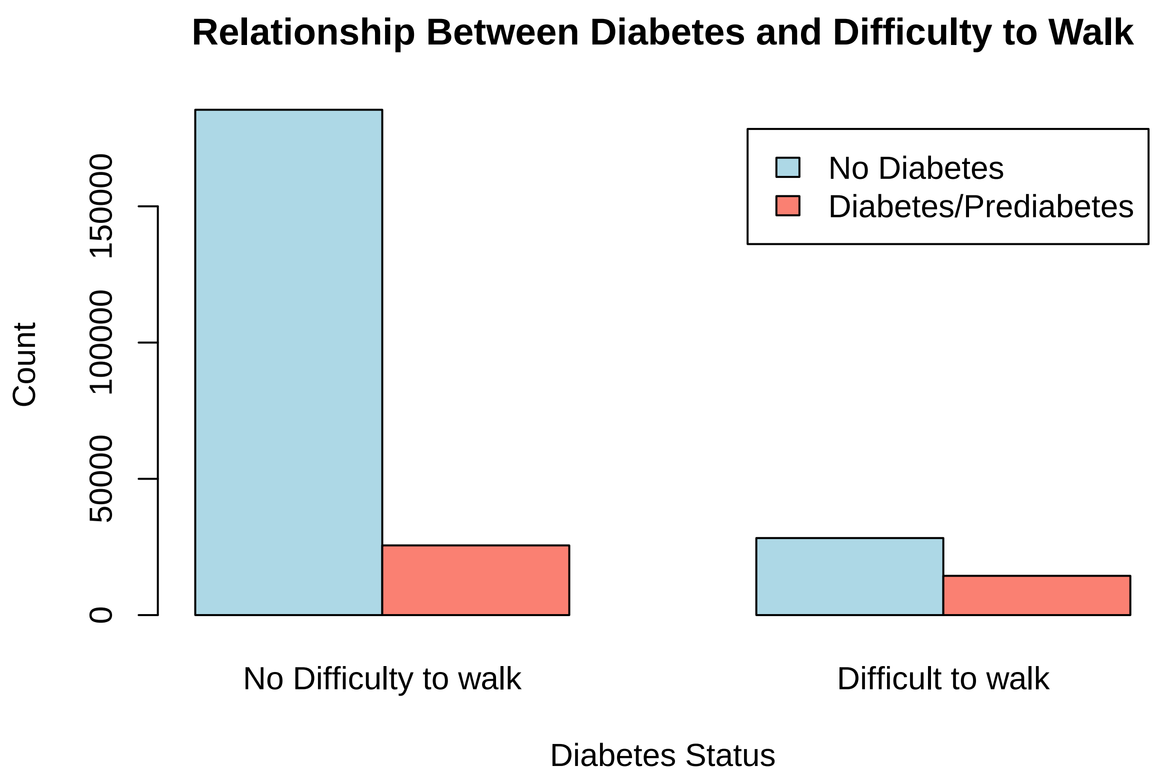

```r
# Boxplot of BMI by Diabetes Status
ggplot(raw_data1, aes(x = Diabetes_012, y = BMI)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Boxplot of BMI by Diabetes Status", x = "Diabetes Status", y = "BMI") +
  theme_minimal()
```



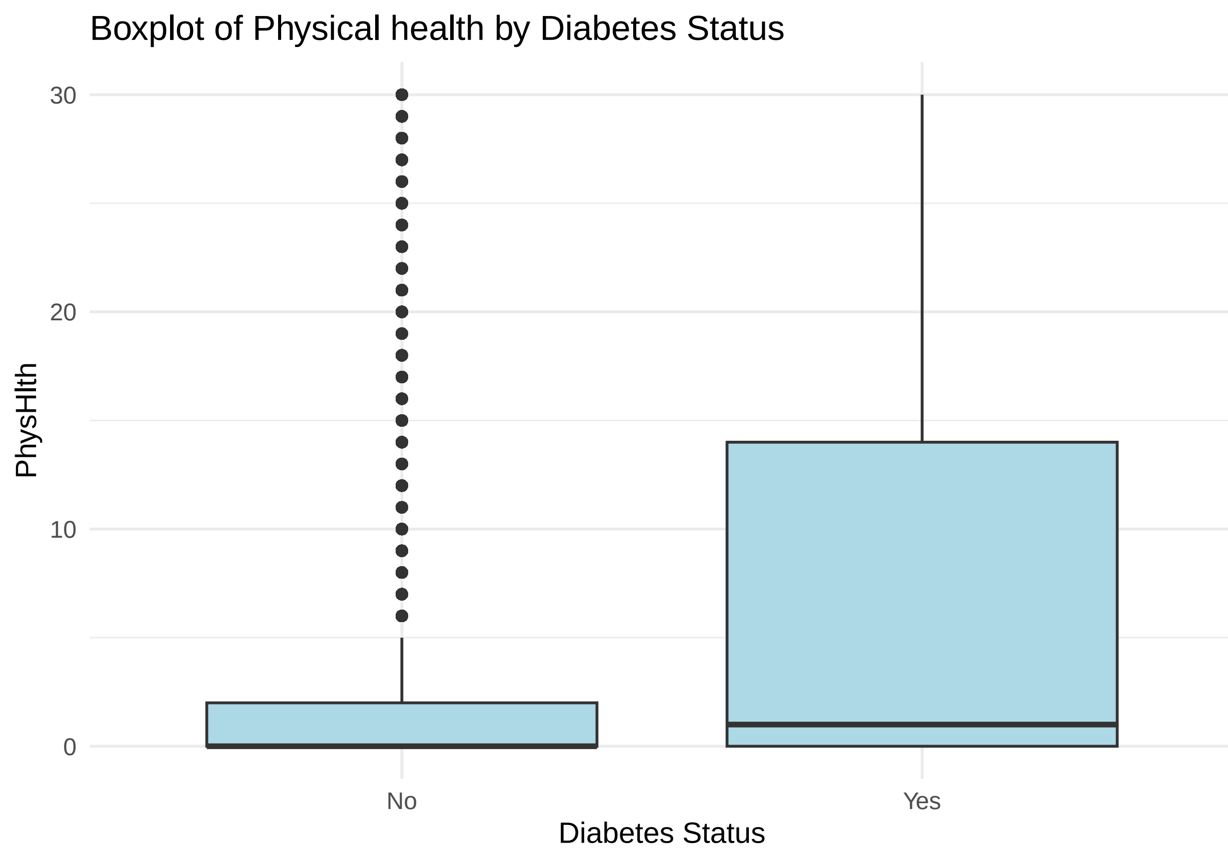

```r
# Summarize the data
diabetes_ghealth_table <- table(raw_data$Diabetes_012, raw_data$GenHlth)
colnames(diabetes_ghealth_table) <- c("1", "2", "3", "4", "5")
rownames(diabetes_ghealth_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_ghealth_table, beside = TRUE,
        main = "Relationship Between Diabetes and General Health",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_ghealth_table))
```



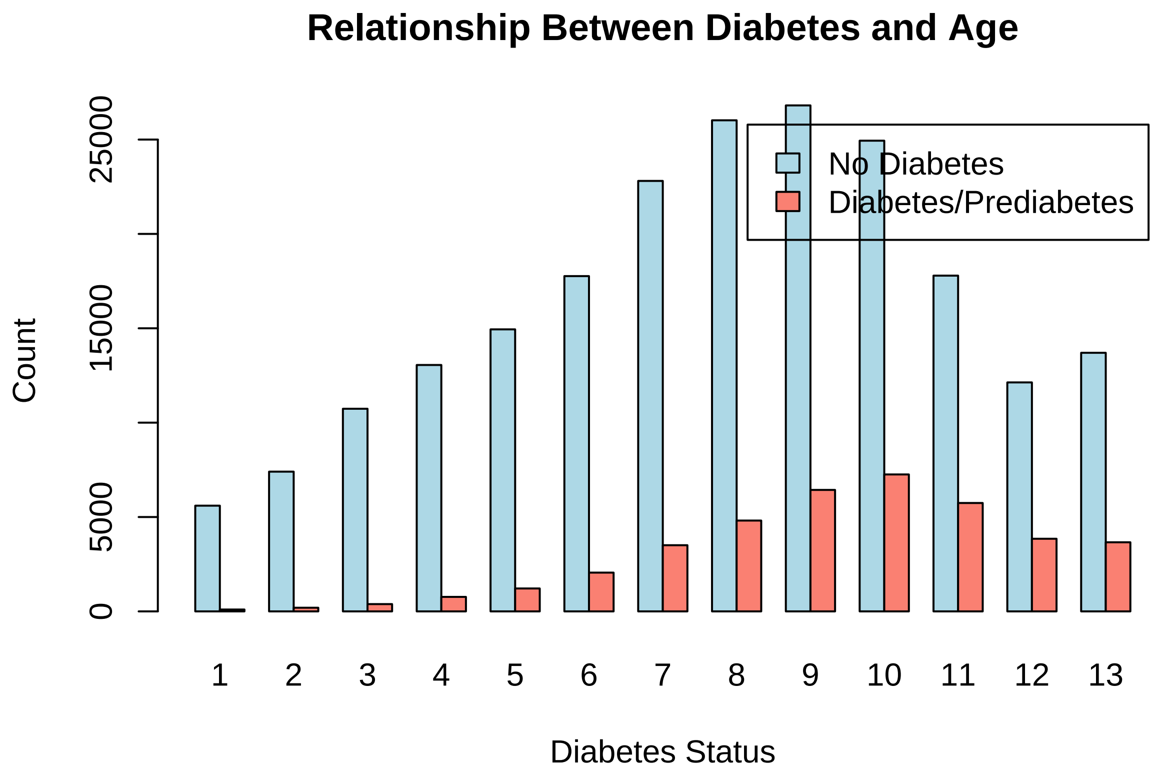

```r
# Summarize the data
diabetes_walk_table <- table(raw_data$Diabetes_012, raw_data$DiffWalk)
colnames(diabetes_walk_table) <- c("No Difficulty to walk", "Difficult to walk")
rownames(diabetes_walk_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_walk_table, beside = TRUE,
        main = "Relationship Between Diabetes and Difficulty to Walk",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_walk_table))
```



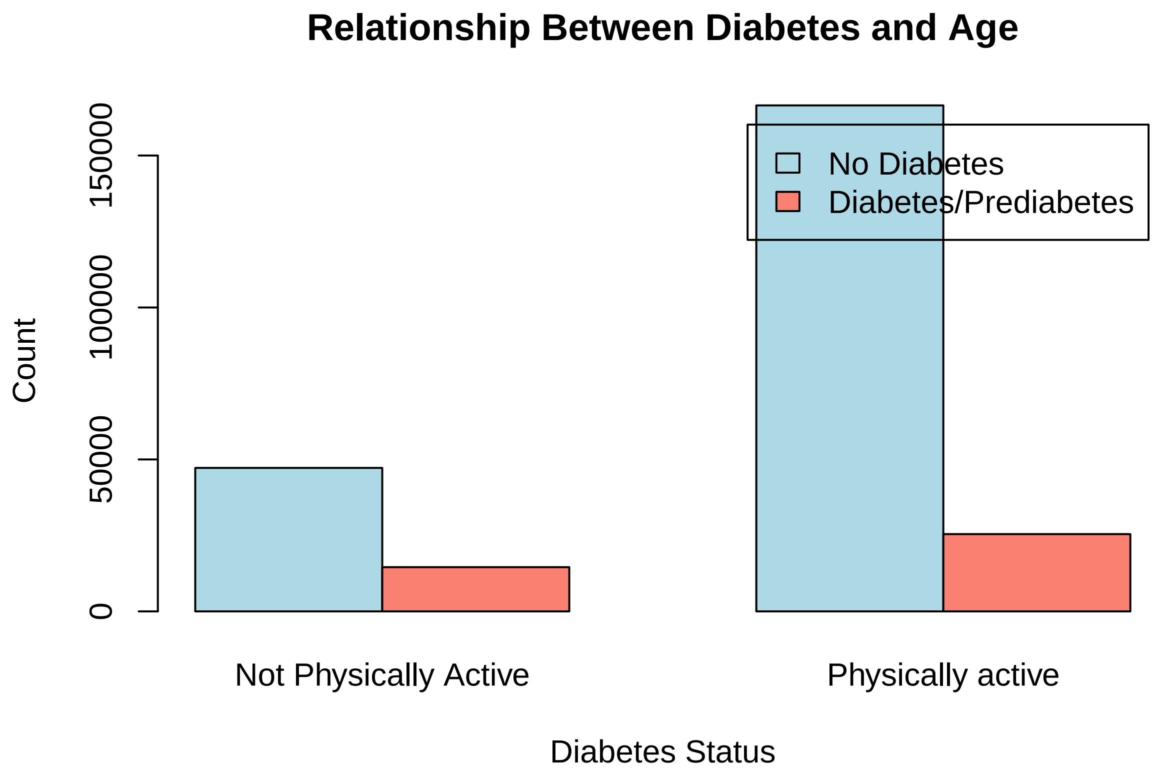

```
ggplot(raw_data1, aes(x = Diabetes_012, y = PhysHlth)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Boxplot of Physical health by Diabetes Status", x = "Diabetes Status", y = "
  theme_minimal()
```




```r
# Summarize the data
diabetes_age_table <- table(raw_data$Diabetes_012, raw_data$Age)
colnames(diabetes_age_table) <- c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11",
rownames(diabetes_age_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_age_table, beside = TRUE,
        main = "Relationship Between Diabetes and Age",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_age_table))
```




```r
# Summarize the data
diabetes_pact_table <- table(raw_data$Diabetes_012, raw_data$PhysActivity)
colnames(diabetes_pact_table) <- c("Not Physically Active", "Physically active")
rownames(diabetes_pact_table) <- c("No Diabetes", "Diabetes/Prediabetes")

# Create the bar plot
barplot(diabetes_pact_table, beside = TRUE,
        main = "Relationship Between Diabetes and Age",
        xlab = "Diabetes Status", ylab = "Count",
        col = c("lightblue", "salmon"),
        legend = rownames(diabetes_pact_table))
```



We can make an observation of how each feature may affect diagnosis of diabetes. For example BMI, Lower the BMI lower the mean patient is being diagnosed with diabetes. Additionally, Age also plays a factor in diabetes diagnsosis. From what we see, higher the age category the greater the chance of being diagnosed. People with higher BP, higher Chol, worse general health, difficulty in walking also have a greater chance of being diagnosed with Diabetes.

Normality and Distribution evaluation

Before we start I want to see if the data set is normally distributed for raw_data

```
# Check if the values follow a normal distribution
samp_subset <- raw_data[sample(nrow(raw_data1),500,replace = FALSE),]


# Assuming raw_data is your dataframe
for (col in names(samp_subset)) {
  # Check if the column is numeric (continuous data)
  if (is.numeric(samp_subset[[col]])) {
    # Perform Shapiro-Wilk test
    test_result <- shapiro.test(samp_subset[[col]])

    # Print results
    cat("Shapiro-Wilk test for", col, ":\n")
    print(test_result)
```

```
    cat("\n")
  }
}
```

```
## Shapiro-Wilk test for Diabetes_012 :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.42033, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HighBP :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.63129, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HighChol :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.63372, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for CholCheck :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.14413, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for BMI :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.84572, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Smoker :
##
##  Shapiro-Wilk normality test
```

```
##
## data:  samp_subset[[col]]
## W = 0.62978, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Stroke :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.15174, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HeartDiseaseorAttack :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.33811, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for PhysActivity :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.52048, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HvyAlcoholConsump :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.24073, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for GenHlth :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.897, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for MentHlth :
##
##  Shapiro-Wilk normality test
```

```
##
## data:  samp_subset[[col]]
## W = 0.4565, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for PhysHlth :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.50434, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for DiffWalk :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.44304, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Sex :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.63346, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Age :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.95449, p-value = 2.712e-11
##
##
## Shapiro-Wilk test for Education :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.8286, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Income :
##
##  Shapiro-Wilk normality test
```

```
##
## data:  samp_subset[[col]]
## W = 0.85334, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for FruitsOrVeggies :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.36822, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for AnyHealthcareorNoDocbcCost :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset[[col]]
## W = 0.15914, p-value < 2.2e-16
```

```r
# Check if the values follow a normal distribution
samp_subset2 <- raw_data[sample(nrow(raw_data2),500,replace = FALSE),]


# Assuming raw_data is your dataframe
for (col in names(samp_subset2)) {
  # Check if the column is numeric (continuous data)
  if (is.numeric(samp_subset2[[col]])) {
    # Perform Shapiro-Wilk test
    test_result <- shapiro.test(samp_subset2[[col]])

    # Print results
    cat("Shapiro-Wilk test for", col, ":\n")
    print(test_result)
    cat("\n")
  }
}
```

```
## Shapiro-Wilk test for Diabetes_012 :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.43194, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HighBP :
```

```
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.6262, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HighChol :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.63197, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for CholCheck :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.15914, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for BMI :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.86685, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Smoker :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.63197, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Stroke :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.20593, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HeartDiseaseorAttack :
```

```
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.34204, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for PhysActivity :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.53499, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for HvyAlcoholConsump :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.23521, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for GenHlth :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.89469, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for MentHlth :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.49638, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for PhysHlth :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.5188, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for DiffWalk :
```

```
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.40508, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Sex :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.63018, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Age :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.96252, p-value = 5.577e-10
##
##
## Shapiro-Wilk test for Education :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.82267, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for Income :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.85086, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for FruitsOrVeggies :
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.34204, p-value < 2.2e-16
##
##
## Shapiro-Wilk test for AnyHealthcareorNoDocbcCost :
```

```
##
##  Shapiro-Wilk normality test
##
## data:  samp_subset2[[col]]
## W = 0.14413, p-value < 2.2e-16
```

As seen above, in both the data sets the pvalue for all features is less than 0.05 and thus none of the columns follows a normal distribution.

Now we are going to see if there is any dependency significance significance between the target column Diabetes-012 and all the other features

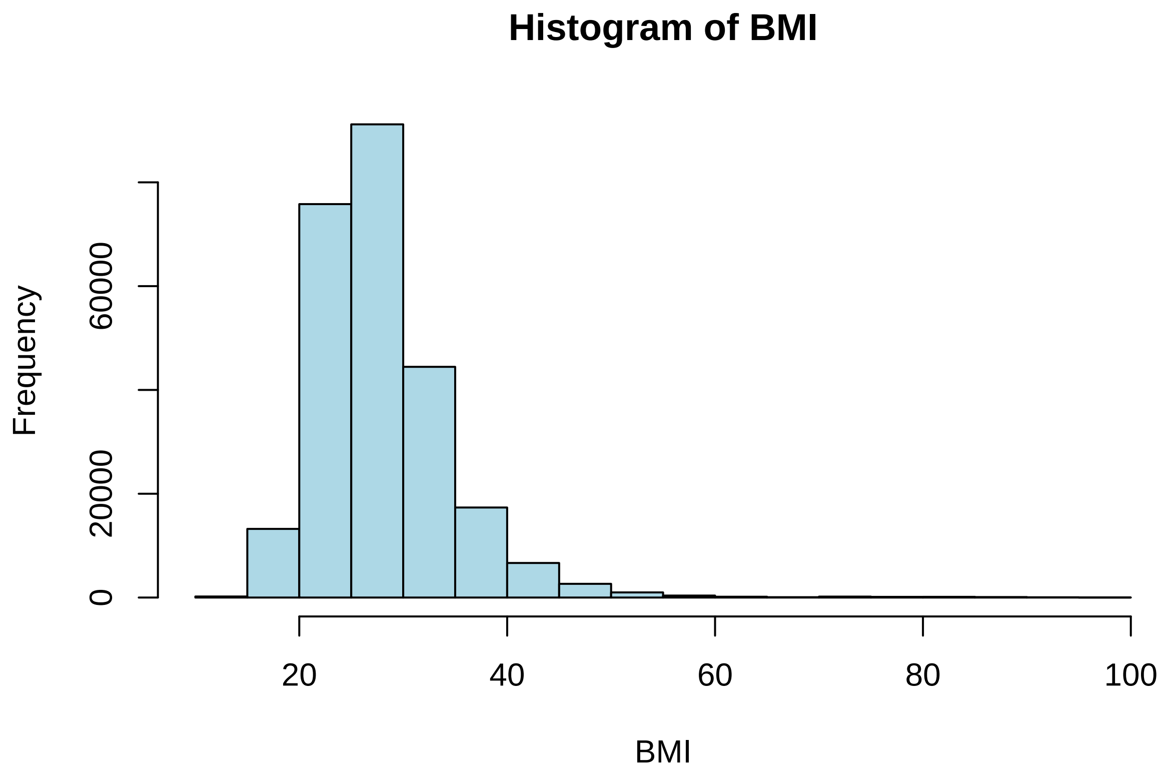
```
cols <- c("HighBP", "HighChol", "CholCheck", "Smoker", "Stroke", "HeartDiseaseorAttack", "Ph

for (col in cols) {
  # Create a temporary data frame excluding NA values
  temp_data <- na.omit(data.frame(raw_data1$Diabetes_012, raw_data1[[col]]))

  # Perform Chi-squared test
  chi_test_result <- suppressWarnings(chisq.test(temp_data[[1]], temp_data[[2]]))

  # Extract p-value
  p_value <- chi_test_result$p.value

  # Check if p-value is significant
  if (!is.na(p_value) && p_value <= 0.05) {
    print(paste0("In raw_data1 there is a significant association between Diabetes_012 and
  } else {
    print(paste0("No significant association found between Diabetes_012 and ", col))
  }
}
```

```
## [1] "In raw_data1 there is a significant association between Diabetes_012 and HighBP"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and HighChol"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and CholCheck"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and Smoker"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and Stroke"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and HeartDiseas
## [1] "In raw_data1 there is a significant association between Diabetes_012 and PhysActivit
## [1] "In raw_data1 there is a significant association between Diabetes_012 and FruitsOrVeg
## [1] "In raw_data1 there is a significant association between Diabetes_012 and HvyAlcohol(
## [1] "In raw_data1 there is a significant association between Diabetes_012 and AnyHealthca
## [1] "In raw_data1 there is a significant association between Diabetes_012 and DiffWalk"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and Sex"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and Age"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and Education"
## [1] "In raw_data1 there is a significant association between Diabetes_012 and Income"
```

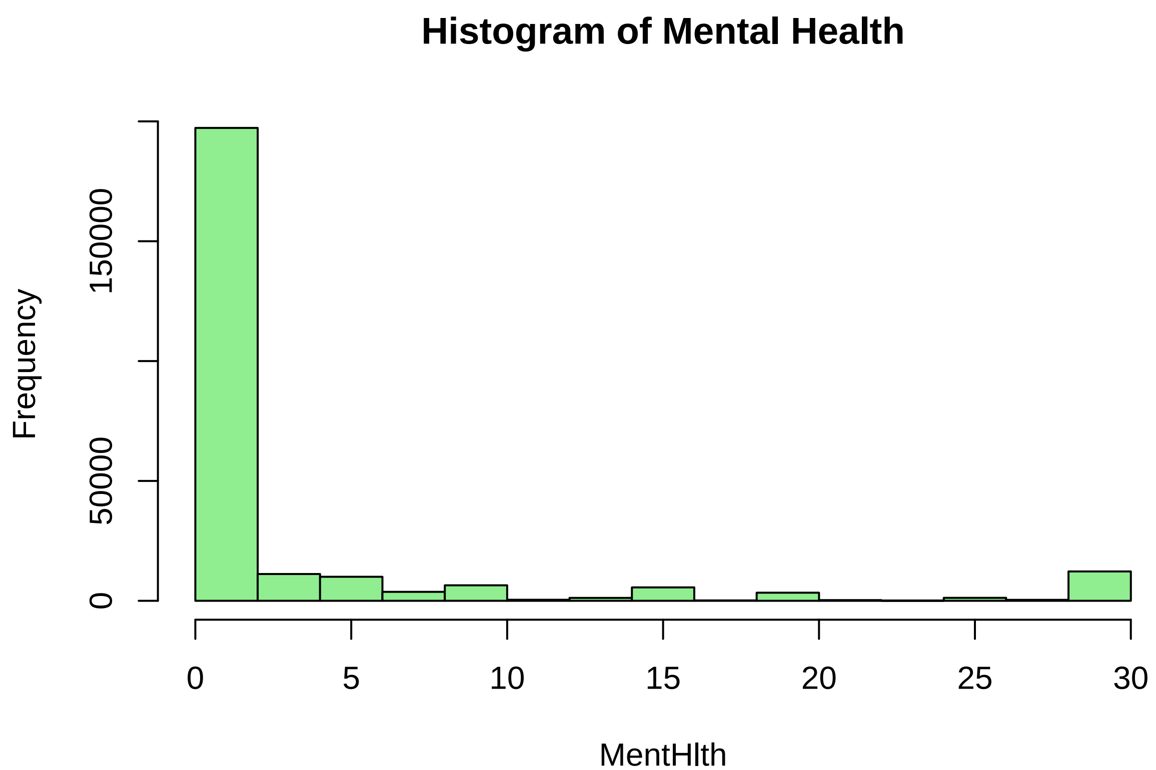
```
for (col in cols) {
  # Create a temporary data frame excluding NA values
  temp_data <- na.omit(data.frame(raw_data2$Diabetes_012, raw_data2[[col]]))

  # Perform Chi-squared test
  chi_test_result <- suppressWarnings(chisq.test(temp_data[[1]], temp_data[[2]]))

  # Extract p-value
  p_value <- chi_test_result$p.value

  # Check if p-value is significant
  if (!is.na(p_value) && p_value <= 0.05) {
    print(paste0("In raw_data2 there is a significant association between Diabetes_012 and
  } else {
    print(paste0("No significant association found between Diabetes_012 and ", col))
  }
}
```

```
## [1] "In raw_data2 there is a significant association between Diabetes_012 and HighBP"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and HighChol"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and CholCheck"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and Smoker"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and Stroke"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and HeartDiseas
## [1] "In raw_data2 there is a significant association between Diabetes_012 and PhysActivit
## [1] "In raw_data2 there is a significant association between Diabetes_012 and FruitsOrVeg
## [1] "In raw_data2 there is a significant association between Diabetes_012 and HvyAlcohol(
## [1] "In raw_data2 there is a significant association between Diabetes_012 and AnyHealthca
## [1] "In raw_data2 there is a significant association between Diabetes_012 and DiffWalk"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and Sex"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and Age"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and Education"
## [1] "In raw_data2 there is a significant association between Diabetes_012 and Income"
```

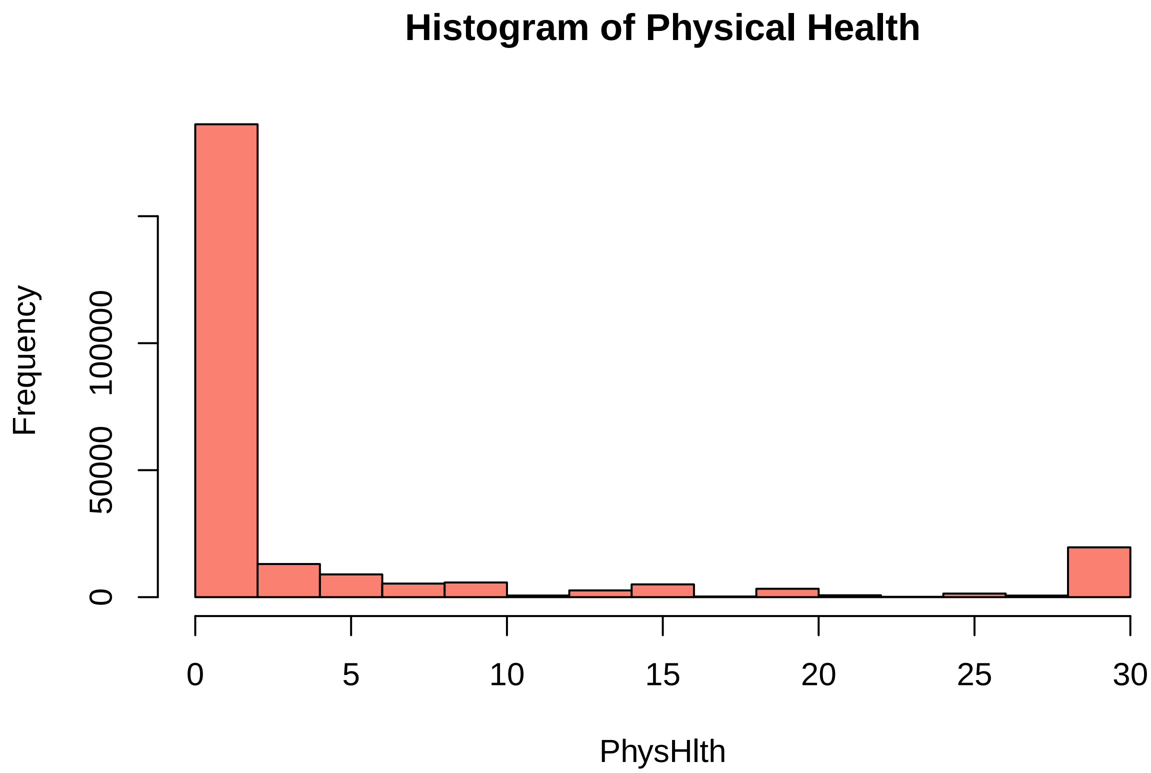
```
# Histogram for BMI
if (is.numeric(raw_data1$BMI)) {
    hist(raw_data$BMI, main = "Histogram of BMI", xlab = "BMI", col = "lightblue", border =
    # Histogram for MentHlth
    hist(raw_data1$MentHlth, main = "Histogram of Mental Health", xlab = "MentHlth", col =
    # Histogram for PhysHlth
    hist(raw_data1$PhysHlth, main = "Histogram of Physical Health", xlab = "PhysHlth", col =
} else {
    cat("The column is not numeric or does not exist in raw_data2.")
}
```



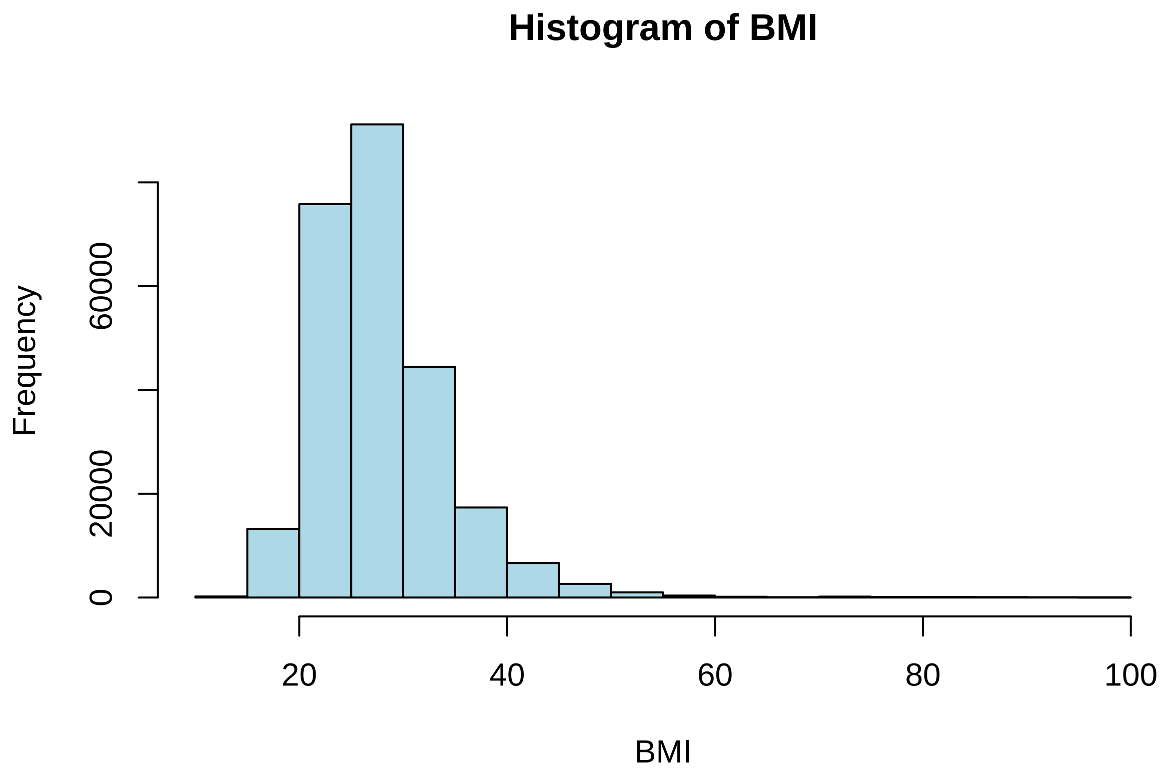
Histogram of BMI
Frequency
0
20000
60000
20
40
60
80
100
BMI

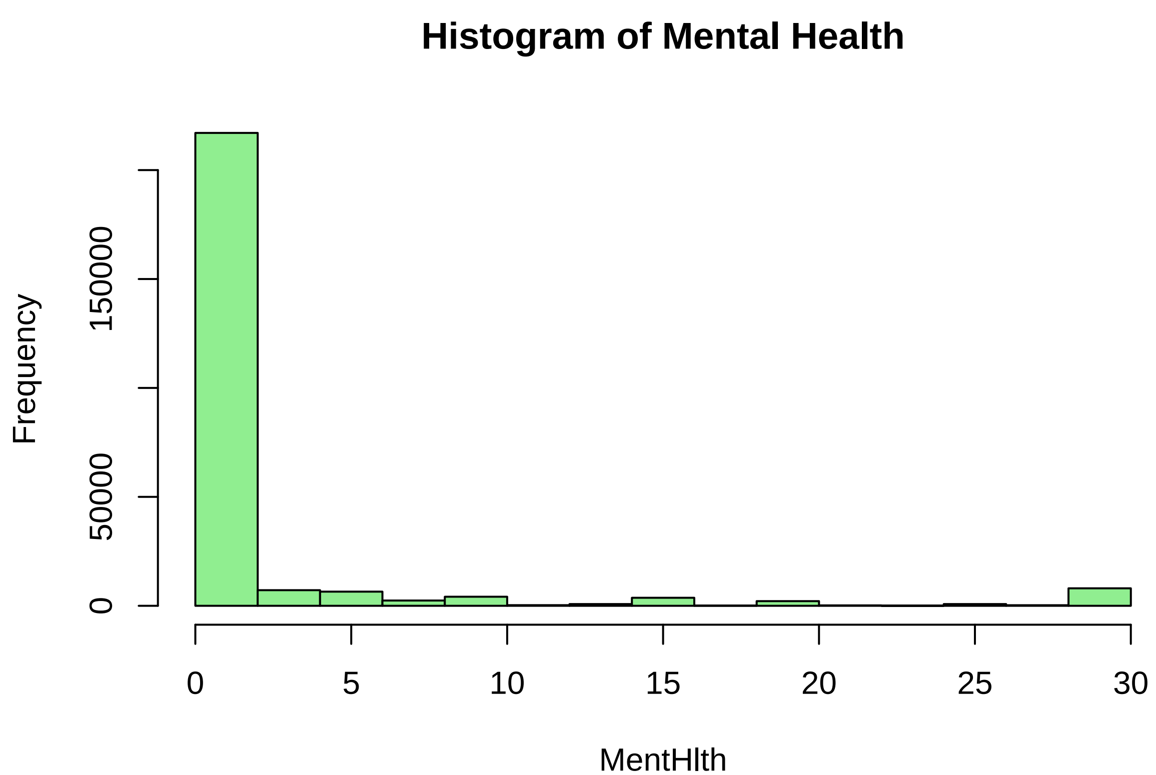


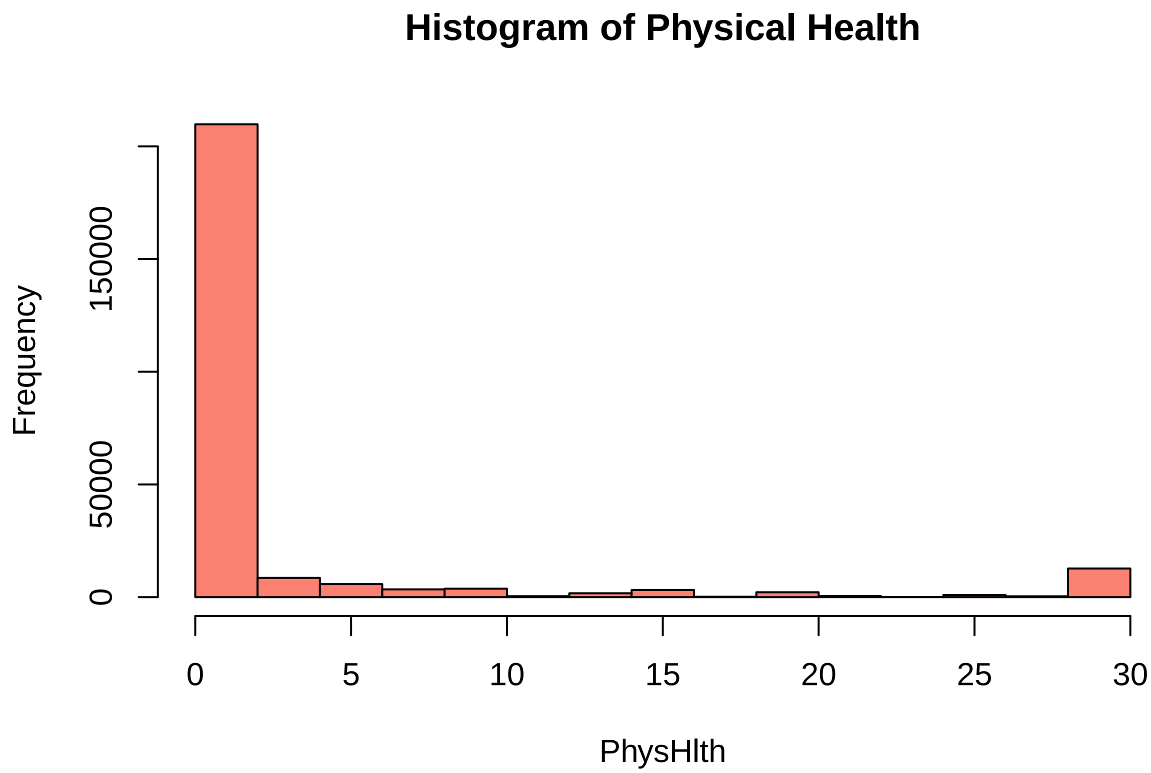
Histogram of Mental Health
Frequency
0
50000
150000
0
5
10
15
20
25
30
MentHlth




```r
# Histogram for BMI
if (is.numeric(raw_data2$BMI)) {
    hist(raw_data$BMI, main = "Histogram of BMI", xlab = "BMI", col = "lightblue", border =
    # Histogram for MentHlth
    hist(raw_data2$MentHlth, main = "Histogram of Mental Health", xlab = "MentHlth", col =
    # Histogram for PhysHlth
    hist(raw_data2$PhysHlth, main = "Histogram of Physical Health", xlab = "PhysHlth", col =
} else {
    cat("The column is not numeric or does not exist in raw_data2.")
}
```


Histogram of BMI
Frequency
0
20000
60000
20
40
60
80
100
BMI

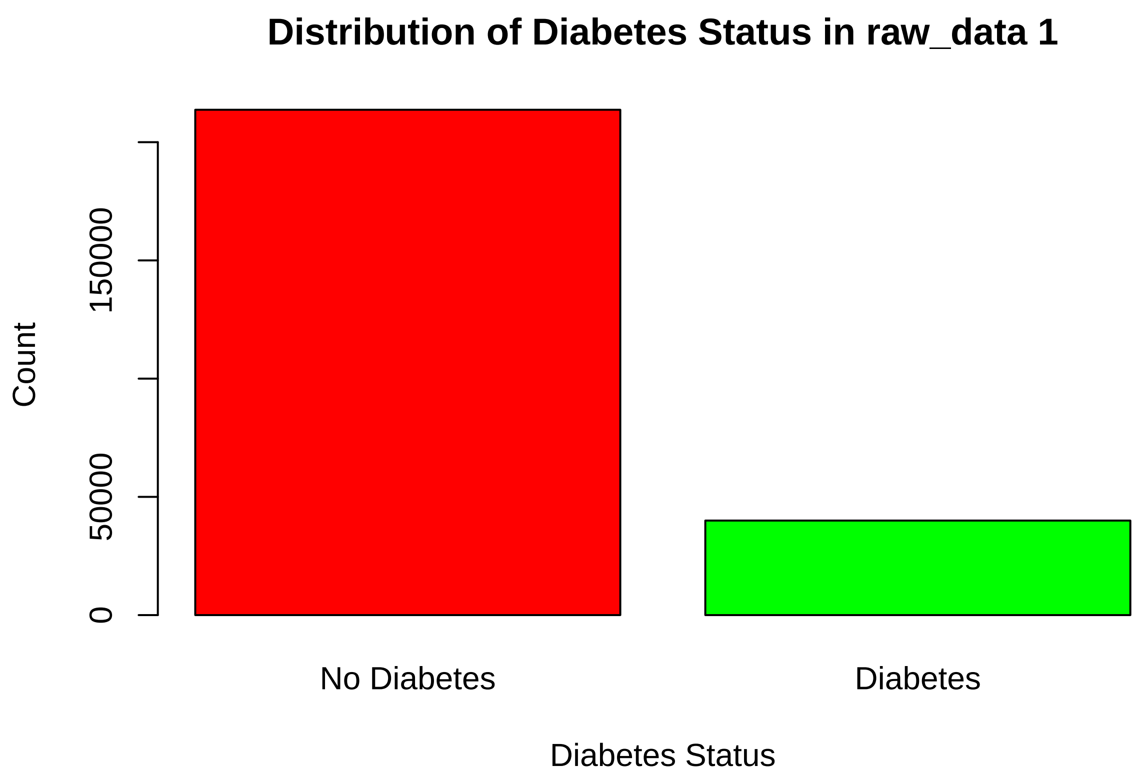

Histogram of Mental Health
Frequency
0
50000
150000
0
5
10
15
20
25
30
MentHlth



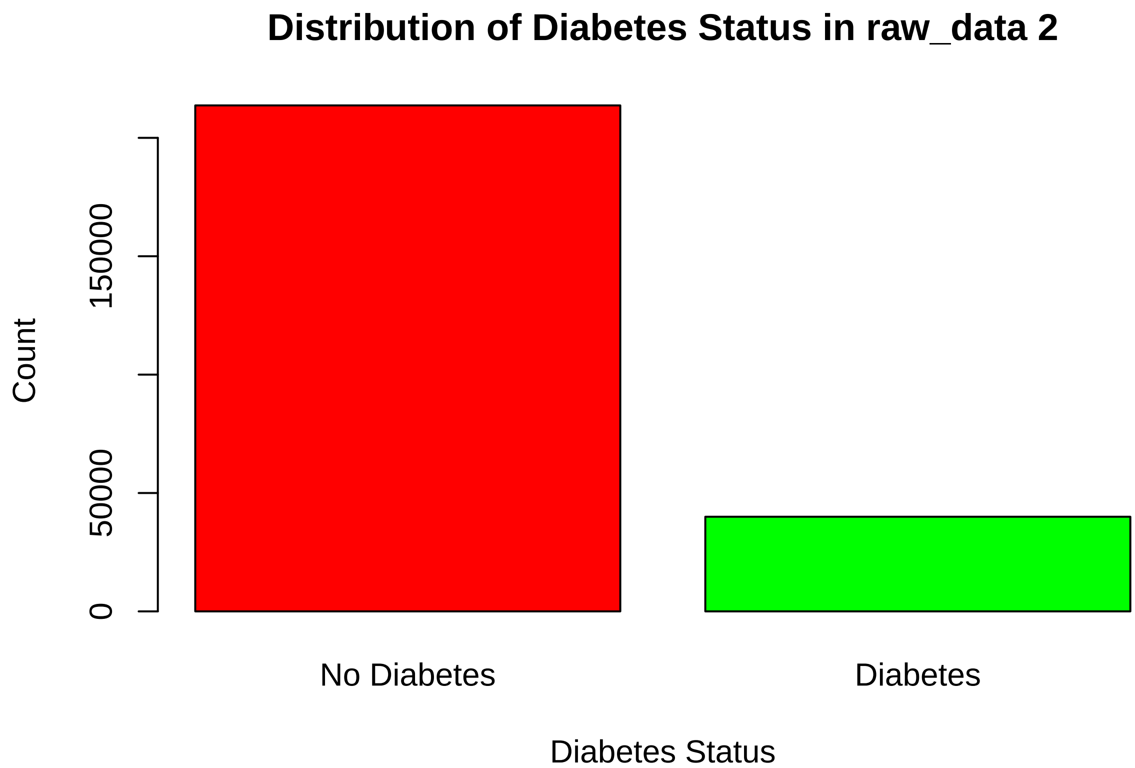
As we can see the dat distribution for the continous data in our dataset are right skewed

The below code chunk will produce the barplot depicting the number of patients who are diabetic or are not diabetic.

```
# Counting the occurrences of each diabetes status
diabetes_counts <- table(raw_data1$Diabetes_012)

# Convert to a data frame for plotting
diabetes_df <- as.data.frame(diabetes_counts)
names(diabetes_df) <- c("Diabetes_Status", "Count")

# Create the bar plot using base R's barplot function
barplot(diabetes_df$Count,
        names.arg = c("No Diabetes", "Diabetes"),
        col = rainbow(3),
        main = "Distribution of Diabetes Status in raw_data 1",
        xlab = "Diabetes Status",
        ylab = "Count")
```




```r
# Counting the occurrences of each diabetes status
diabetes_counts <- table(raw_data2$Diabetes_012)

# Convert to a data frame for plotting
diabetes_df2 <- as.data.frame(diabetes_counts)
names(diabetes_df2) <- c("Diabetes_Status", "Count")

# Create the bar plot using base R's barplot function
barplot(diabetes_df2$Count,
        names.arg = c("No Diabetes", "Diabetes"),
        col = rainbow(3),
        main = "Distribution of Diabetes Status in raw_data 2",
        xlab = "Diabetes Status",
        ylab = "Count")
```



In raw_data1 we can see that there are 213703 patients are non diabetic 39977 are diabetic. In raw_data 2 we can see that there are 213703 patients are non diabetic 39977 that are diabetic. As we can see there are equal number of diabetic and non-diabetic patients in both data sets.

Now we are going to find the correlation between the target feature and the other features in the data set

Outlier detection

The below code chunk will check if there are any outliers.

We are now going to find any outliers in the updated data set. Since the Dataset does not follow a normal distribution, we will be using the IQR method to find the outliers.

```r
# Function to find outliers using IQR
find_outliers <- function(data) {
  Q1 <- quantile(data, 0.25, na.rm = TRUE)
  Q3 <- quantile(data, 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  return(data < lower_bound | data > upper_bound)
}
```

```r
# Loop through each column and find outliers
outliers_list1 <- lapply(raw_data1[, sapply(raw_data1, is.numeric)], find_outliers)

# Summarize the number of outliers in each column
outlier_counts1 <- sapply(outliers_list1, function(x) sum(x, na.rm = TRUE))
print(outlier_counts1)
```

```
##       BMI   GenHlth  MentHlth  PhysHlth       Age Education    Income
##      9847     12081     36208     40949         0         0         0
```

```r
# Loop through each column and find outliers
outliers_list2 <- lapply(raw_data2[, sapply(raw_data2, is.numeric)], find_outliers)

# Summarize the number of outliers in each column
outlier_counts2 <- sapply(outliers_list2, function(x) sum(x, na.rm = TRUE))
print(outlier_counts2)
```

```
##       BMI   GenHlth  MentHlth  PhysHlth       Age Education    Income
##     42010      7814     50650     60858     27016      8919     37540
```

As we can see both the data sets have a few outliers. Prior to building our models we will have to remove any problematic outliers. However given the fact that the data is not normally distributed we do not need to remove the outliers. However we can lessen the impact they have on the dataset by performing log transformation and then standardizing the value after to ensure the data is ideal for our models. Interstingly enough there are more outliers in AGE, Education and Income columns of raw_data2. This could be due to the fact those columns had missing values and were imputed with the median.

Log transformation

We can see based on the graph that the data is not normall distributed and certain features are right skewed. In order to get a more normal distribution we can perform a log transformation to ensure an even distribution. This also ensure any of our modelling for our machine learning algorithms are not effected.

```r
# Select only numeric columns for log transformation
numeric_columns <- sapply(raw_data1, is.numeric)

# Check for zero or negative values and add a constant if necessary
raw_data1[numeric_columns] <- lapply(raw_data1[numeric_columns], function(x) {
    if(any(x <= 0)) {
        x + 1 # Add 1 to handle zero or negative values
    } else {
        x
    }
```

```r
})

# Apply log transformation
raw_data1[numeric_columns] <- lapply(raw_data1[numeric_columns], log)

# Check the transformed data
head(raw_data1)
```

```
##   Diabetes_012 HighBP HighChol CholCheck      BMI Smoker Stroke
## 1           No    Yes      Yes       Yes 3.688879    Yes     No
## 2           No     No       No        No 3.218876    Yes     No
## 3           No    Yes      Yes       Yes 3.332205     No     No
## 4           No    Yes       No       Yes 3.295837     No     No
## 5           No    Yes      Yes       Yes 3.178054     No     No
## 6           No    Yes      Yes       Yes 3.218876    Yes     No
##   HeartDiseaseorAttack PhysActivity HvyAlcoholConsump   GenHlth MentHlth
## 1                   No           No                No 1.6094379 2.944439
## 2                   No          Yes                No 1.0986123 0.000000
## 3                   No           No                No 1.6094379 3.433987
## 4                   No          Yes                No 0.6931472 0.000000
## 5                   No          Yes                No 0.6931472 1.386294
## 6                   No          Yes                No 0.6931472 0.000000
##   PhysHlth DiffWalk    Sex      Age Education   Income FruitsOrVeggies
## 1 2.772589      Yes   Male 2.197225  1.386294 1.098612             Yes
## 2 0.000000       No   Male 1.945910  1.791759 0.000000              No
## 3 3.433987      Yes   Male 2.197225  1.386294 2.079442             Yes
## 4 0.000000       No   Male 2.397895  1.098612 1.791759             Yes
## 5 0.000000       No   Male 2.397895  1.609438 1.386294             Yes
## 6 1.098612       No Female 2.302585  1.791759 2.079442             Yes
##   AnyHealthcareorNoDocbcCost
## 1                        Yes
## 2                        Yes
## 3                        Yes
## 4                        Yes
## 5                        Yes
## 6                        Yes
```

```
##   Diabetes_012 HighBP HighChol CholCheck BMI Smoker Stroke HeartDiseaseorAttack
## 1           No    Yes       No       Yes  40    Yes     No                   No
## 2           No     No       No        No  25    Yes     No                   No
## 3           No    Yes       No       Yes  27     No     No                   No
## 4           No    Yes       No       Yes  27     No     No                   No
## 5           No    Yes      Yes       Yes  24     No     No                   No
## 6           No    Yes      Yes       Yes  27    Yes     No                   No
##   PhysActivity HvyAlcoholConsump GenHlth MentHlth PhysHlth DiffWalk  Sex Age
## 1           No                No       5       18       15      Yes Male   9
## 2          Yes                No       2        0        0       No Male   7
```

```
## 3          No                 No       5        0        0       No Male   9
## 4         Yes                 No       2        0        0       No Male  11
## 5         Yes                 No       2        3        0       No Male  11
## 6         Yes                 No       2        0        0       No Male  10
##   Education Income FruitsOrVeggies AnyHealthcareorNoDocbcCost
## 1         5      3             Yes                        Yes
## 2         6      7              No                        Yes
## 3         4      7             Yes                        Yes
## 4         5      6             Yes                        Yes
## 5         5      4             Yes                        Yes
## 6         6      7             Yes                        Yes
```

```
# Select only numeric columns for log transformation
numeric_columns <- sapply(raw_data2, is.numeric)

# Check for zero or negative values and add a constant if necessary
raw_data2[numeric_columns] <- lapply(raw_data2[numeric_columns], function(x) {
    if(any(x <= 0)) {
        x + 1 # Add 1 to handle zero or negative values
    } else {
        x
    }
})

# Apply log transformation
raw_data2[numeric_columns] <- lapply(raw_data2[numeric_columns], log)

# Check the transformed data
head(raw_data2)
```

```
##   Diabetes_012 HighBP HighChol CholCheck      BMI Smoker Stroke
## 1           No    Yes       No       Yes 3.688879    Yes     No
## 2           No     No       No        No 3.218876    Yes     No
## 3           No    Yes       No       Yes 3.295837     No     No
## 4           No    Yes       No       Yes 3.295837     No     No
## 5           No    Yes      Yes       Yes 3.178054     No     No
## 6           No    Yes      Yes       Yes 3.295837    Yes     No
##   HeartDiseaseorAttack PhysActivity HvyAlcoholConsump   GenHlth MentHlth
## 1                   No           No                No 1.6094379 2.944439
## 2                   No          Yes                No 0.6931472 0.000000
## 3                   No           No                No 1.6094379 0.000000
## 4                   No          Yes                No 0.6931472 0.000000
## 5                   No          Yes                No 0.6931472 1.386294
## 6                   No          Yes                No 0.6931472 0.000000
##   PhysHlth DiffWalk  Sex      Age Education   Income FruitsOrVeggies
## 1 2.772589      Yes Male 2.197225  1.609438 1.098612             Yes
## 2 0.000000       No Male 1.945910  1.791759 1.945910              No
```

```
## 3 0.000000       No Male 2.197225  1.386294 1.945910            Yes
## 4 0.000000       No Male 2.397895  1.609438 1.791759            Yes
## 5 0.000000       No Male 2.397895  1.609438 1.386294            Yes
## 6 0.000000       No Male 2.302585  1.791759 1.945910            Yes
##   AnyHealthcareorNoDocbcCost
## 1                        Yes
## 2                        Yes
## 3                        Yes
## 4                        Yes
## 5                        Yes
## 6                        Yes
```

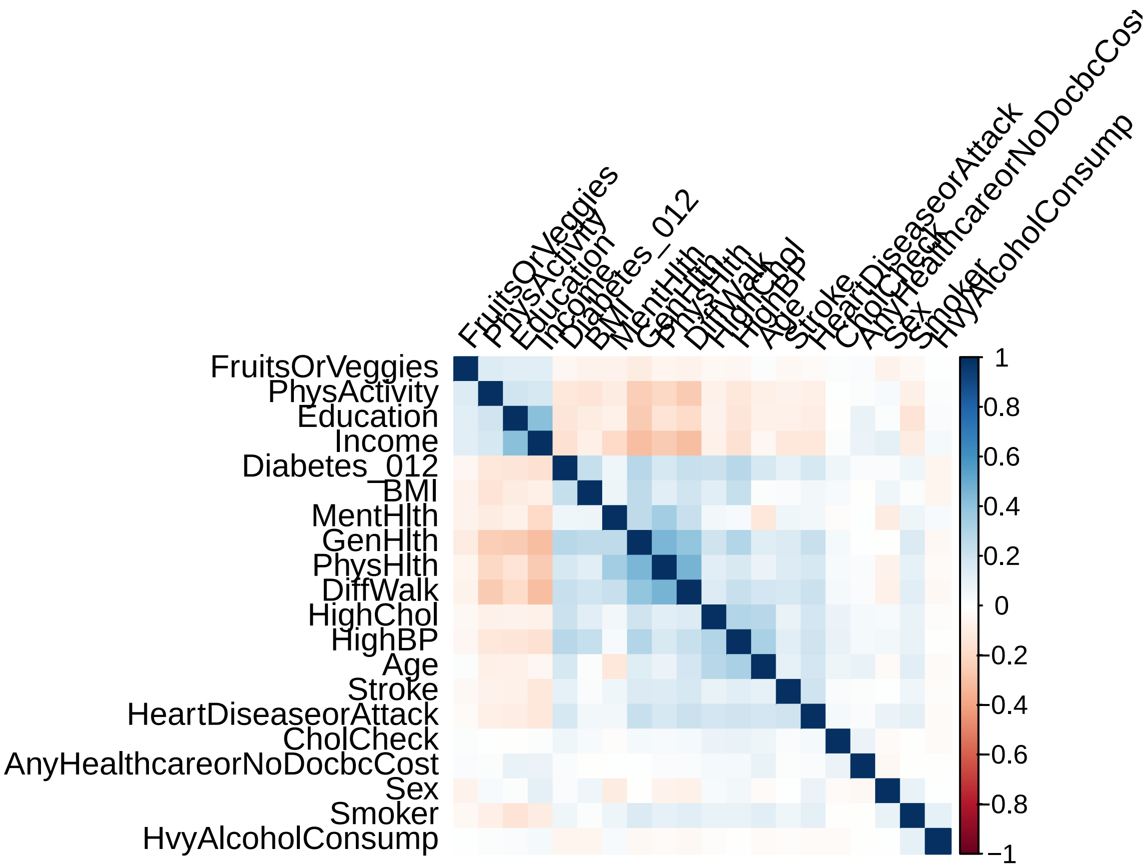
Bindary encoding and Correlation

In order to get the correlation of all the features in the dataset, we will have to convert all categorical data into 0 and 1. We will do this via Binary encoding.

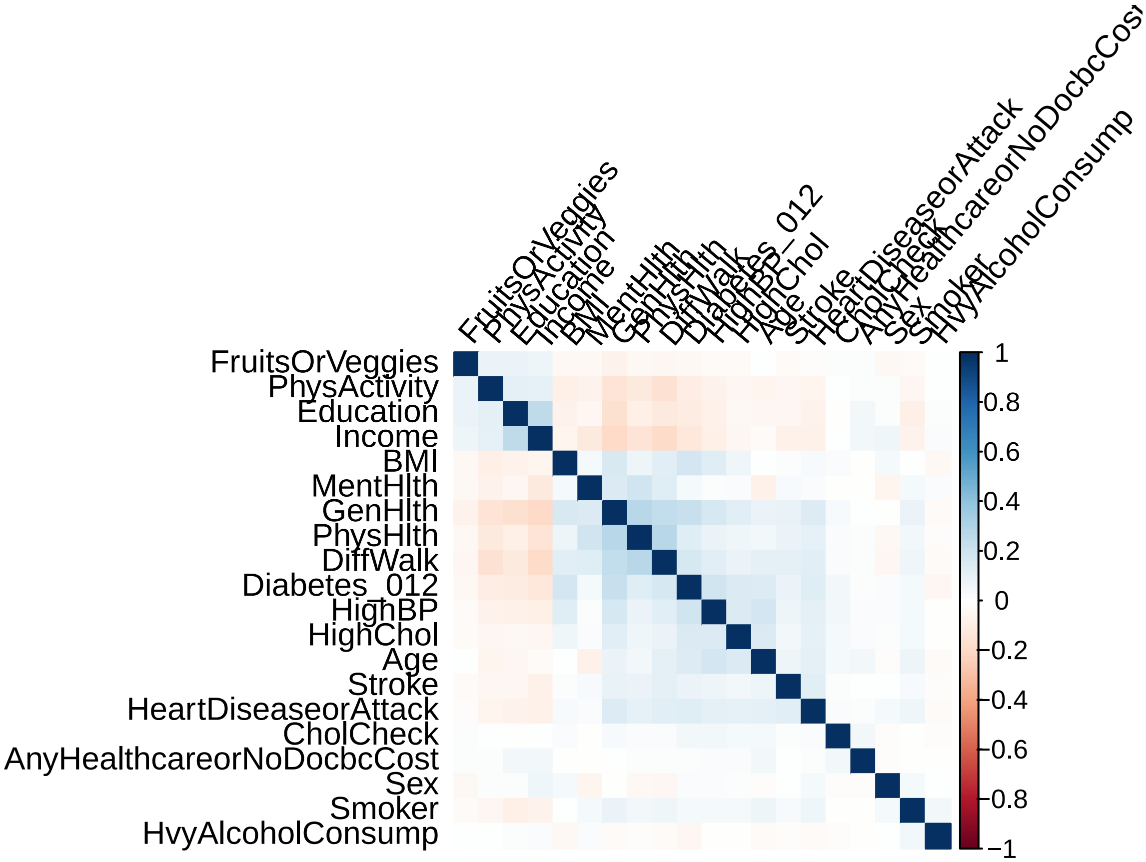
```
raw_data1a <- raw_data1
raw_data1a$Diabetes_012 <- ifelse(raw_data1a$Diabetes_012 == "No", 0, 1)
raw_data1a$HighBP <- ifelse(raw_data1a$HighBP == "No", 0, 1)
raw_data1a$HighChol <- ifelse(raw_data1a$HighChol == "No", 0, 1)
raw_data1a$CholCheck <- ifelse(raw_data1a$CholCheck == "No", 0, 1)
raw_data1a$Smoker <- ifelse(raw_data1a$Smoker == "No", 0, 1)
raw_data1a$Stroke <- ifelse(raw_data1a$Stroke == "No", 0, 1)
raw_data1a$HeartDiseaseorAttack <- ifelse(raw_data1a$HeartDiseaseorAttack == "No", 0, 1)
raw_data1a$PhysActivity <- ifelse(raw_data1a$PhysActivity == "No", 0, 1)
raw_data1a$FruitsOrVeggies <- ifelse(raw_data1a$Fruits == "No", 0, 1)
raw_data1a$HvyAlcoholConsump <- ifelse(raw_data1a$HvyAlcoholConsump == "No", 0, 1)
raw_data1a$AnyHealthcareorNoDocbcCost <- ifelse(raw_data1a$AnyHealthcare == "No", 0, 1)
raw_data1a$DiffWalk <- ifelse(raw_data1a$DiffWalk == "No", 0, 1)
raw_data1a$Sex <- ifelse(raw_data1a$Sex == "Male", 0, 1)
```

```
raw_data2a <- raw_data2
raw_data2a$Diabetes_012 <- ifelse(raw_data2a$Diabetes_012 == "No", 0, 1)
raw_data2a$HighBP <- ifelse(raw_data2a$HighBP == "No", 0, 1)
raw_data2a$HighChol <- ifelse(raw_data2a$HighChol == "No", 0, 1)
raw_data2a$CholCheck <- ifelse(raw_data2a$CholCheck == "No", 0, 1)
raw_data2a$Smoker <- ifelse(raw_data2a$Smoker == "No", 0, 1)
raw_data2a$Stroke <- ifelse(raw_data2a$Stroke == "No", 0, 1)
raw_data2a$HeartDiseaseorAttack <- ifelse(raw_data2a$HeartDiseaseorAttack == "No", 0, 1)
raw_data2a$PhysActivity <- ifelse(raw_data2a$PhysActivity == "No", 0, 1)
raw_data2a$FruitsOrVeggies <- ifelse(raw_data2a$Fruits == "No", 0, 1)
raw_data2a$HvyAlcoholConsump <- ifelse(raw_data2a$HvyAlcoholConsump == "No", 0, 1)
raw_data2a$AnyHealthcareorNoDocbcCost <- ifelse(raw_data2a$AnyHealthcare == "No", 0, 1)
raw_data2a$DiffWalk <- ifelse(raw_data2a$DiffWalk == "No", 0, 1)
raw_data2a$Sex <- ifelse(raw_data2a$Sex == "Male", 0, 1)
```

```r
library(ggplot2)
library(corrplot)

# Select only continuous variables if necessary
continuous_data1 <- raw_data1a[, sapply(raw_data1a, is.numeric)]

# Compute the correlation matrix
cor_matrix1 <- cor(continuous_data1, use = "complete.obs")  # 'use' parameter handles miss

# Create a square correlation heatmap
corrplot(cor_matrix1, method = "color", type = "full", order = "hclust",
         tl.col = "black", tl.srt = 50)
```




```r
# Select only continuous variables if necessary
continuous_data <- raw_data2a[, sapply(raw_data2a, is.numeric)]

# Compute the correlation matrix
cor_matrix <- cor(continuous_data, use = "complete.obs")  # 'use' parameter handles missing

# Create a square correlation heatmap
corrplot(cor_matrix, method = "color", type = "full", order = "hclust",
         tl.col = "black", tl.srt = 50)
```



We can see that both data sets have similar correlation with the same features. However it also evident that Diabetes has a stronger positive relationshp with BMI and GenHlth in raw_data2. As mentioned before raw_data2 contains the missing values which havent been treated yet. This indicates that the missing values may play a role in our classification model.

Z score standardization

We will now standardized the data frame. It is a good practice to standardize the data for any PCA or machine learning algorithms

```
standardize <- function(x) {
    return ((x - mean(x, na.rm = TRUE)) / sd(x, na.rm = TRUE))
}
```

Now we will use Z score standardization for Columns BMI, MentHlth and PhysHlth

```
# Standardize the columns
# Apply the standardization to each numeric column in the dataframe
for (column_name in names(raw_data1a)) {
  if (is.numeric(raw_data1[[column_name]])) {
    raw_data1a[[column_name]] <- standardize(raw_data1a[[column_name]])
  }
}
```

```r
# Check the first few rows of the modified data frame
head(raw_data1a, 10)
```

```
##    Diabetes_012 HighBP HighChol CholCheck         BMI Smoker Stroke
## 1             0      1        1         1  1.74823388      1      0
## 2             0      0        0         0 -0.49512169      1      0
## 3             0      1        1         1  0.04580293      0      0
## 4             0      1        0         1 -0.12778202      0      0
## 5             0      1        1         1 -0.68996753      0      0
## 6             0      1        1         1 -0.49512169      1      0
## 7             0      1        0         1  0.37511006      1      0
## 8             0      1        1         1 -0.49512169      1      0
## 9             1      1        1         1  0.37511006      1      0
## 10            0      0        0         1 -0.68996753      0      0
##    HeartDiseaseorAttack PhysActivity HvyAlcoholConsump    GenHlth   MentHlth
## 1                     0            0                 0  1.6886421  2.1905641
## 2                     0            1                 0  0.5957167 -0.5840274
## 3                     0            0                 0  1.6886421  2.6518732
## 4                     0            1                 0 -0.2717870 -0.5840274
## 5                     0            1                 0 -0.2717870  0.7222998
## 6                     0            1                 0 -0.2717870 -0.5840274
## 7                     0            0                 0  0.5957167 -0.5840274
## 8                     0            1                 0  0.5957167 -0.5840274
## 9                     1            0                 0  1.6886421  2.6518732
## 10                    0            0                 0 -0.2717870 -0.5840274
##      PhysHlth DiffWalk Sex        Age   Education     Income FruitsOrVeggies
## 1   1.7203376        1   0  0.4185072 -0.93006552 -1.2067709               1
## 2  -0.6585269        0   0 -0.0574126  0.86321974 -3.3905566               0
## 3   2.2878136        1   0  0.4185072 -0.93006552  0.7428894               1
## 4  -0.6585269        0   0  0.7985218 -2.20242167  0.1710444               1
## 5  -0.6585269        0   0  0.7985218  0.05685057 -0.6349259               1
## 6   0.2840759        0   1  0.6180307  0.86321974  0.7428894               1
## 7   1.6649640        0   0  0.4185072  0.86321974  0.4774601               0
## 8  -0.6585269        1   0  0.7985218 -0.93006552 -0.6349259               1
## 9   2.2878136        1   0  0.4185072  0.05685057 -3.3905566               1
## 10 -0.6585269        0   1  0.1954588 -0.93006552 -1.2067709               1
##    AnyHealthcareorNoDocbcCost
## 1                           1
## 2                           1
## 3                           1
## 4                           1
## 5                           1
## 6                           1
## 7                           1
## 8                           1
## 9                           1
## 10                          1
```

```r
# Standardize the columns
# Apply the standardization to each numeric column in the dataframe
for (column_name in names(raw_data2a)) {
  if (is.numeric(raw_data2a[[column_name]])) {
    raw_data2a[[column_name]] <- standardize(raw_data2a[[column_name]])
  }
}

# Check the first few rows of the modified data frame
head(raw_data2a, 10)
```

```
##    Diabetes_012 HighBP HighChol CholCheck        BMI Smoker Stroke
## 1             0      1        0         1  2.2160064      1      0
## 2             0      0        0         0 -0.5581689      1      0
## 3             0      1        0         1 -0.1039099      0      0
## 4             0      1        0         1 -0.1039099      0      0
## 5             0      1        1         1 -0.7991189      0      0
## 6             0      1        1         1 -0.1039099      1      0
## 7             0      1        0         1  0.5179759      1      0
## 8             0      1        0         1 -0.5581689      1      0
## 9             1      1        0         1 -0.1039099      1      0
## 10            0      0        0         1 -0.7991189      0      0
##    HeartDiseaseorAttack PhysActivity HvyAlcoholConsump    GenHlth   MentHlth
## 1                     0            0                 0  2.1842617  2.8050699
## 2                     0            1                 0 -0.2169301 -0.4447664
## 3                     0            0                 0  2.1842617 -0.4447664
## 4                     0            1                 0 -0.2169301 -0.4447664
## 5                     0            1                 0 -0.2169301  1.0853145
## 6                     0            1                 0 -0.2169301 -0.4447664
## 7                     0            0                 0  0.8456142 -0.4447664
## 8                     0            1                 0  0.8456142 -0.4447664
## 9                     1            0                 0  2.1842617  3.3453941
## 10                    0            1                 0 -0.2169301 -0.4447664
##      PhysHlth DiffWalk Sex        Age   Education       Income FruitsOrVeggies
## 1   2.2561835        1   0  0.4323543  0.04750712 -1.634701031               1
## 2  -0.4946554        0   0 -0.1540546  1.04707250  0.370448513               0
## 3  -0.4946554        0   0  0.4323543 -1.17586203  0.370448513               1
## 4  -0.4946554        0   0  0.9005928  0.04750712  0.005647443               1
## 5  -0.4946554        0   0  0.9005928  0.04750712 -0.953894902               1
## 6  -0.4946554        0   0  0.6781991  1.04707250  0.370448513               1
## 7  -0.4946554        0   0  0.4323543  0.04750712  0.370448513               0
## 8  -0.4946554        0   0  0.9005928 -1.17586203  0.370448513               1
## 9  -0.4946554        1   0  0.1575232  0.04750712 -4.234591851               1
## 10 -0.4946554        0   1  0.1575232 -1.17586203  0.370448513               1
##    AnyHealthcareorNoDocbcCost
## 1                           1
## 2                           1
```

```
## 3                              1
## 4                              1
## 5                              1
## 6                              1
## 7                              1
## 8                              1
## 9                              1
## 10                             1
```

```r
summary(raw_data1a)
```

```
##  Diabetes_012 HighBP     HighChol    CholCheck       BMI            Smoker
##  0:213703     0:144851   0:146089   0:  9470   Min.   :-3.9984   0:141257
##  1: 39977     1:108829   1:107591   1:244210   1st Qu.:-0.6900   1:112423
##                                                Median :-0.1278
##                                                Mean   : 0.0000
##                                                3rd Qu.: 0.5316
##                                                Max.   : 6.0253
##  Stroke     HeartDiseaseorAttack PhysActivity HvyAlcoholConsump
##  0:243388   0:229787             0: 61760     0:239424
##  1: 10292   1: 23893             1:191920     1: 14256
##
##
##
##
##     GenHlth            MentHlth            PhysHlth         DiffWalk    Sex
##  Min.   :-1.7548   Min.   :-0.5840   Min.   :-0.6585   0:211005   0:141974
##  1st Qu.:-0.2718   1st Qu.:-0.5840   1st Qu.:-0.6585   1: 42675   1:111706
##  Median :-0.2718   Median :-0.5840   Median :-0.6585
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.5957   3rd Qu.: 0.4512   3rd Qu.: 0.5309
##  Max.   : 1.6886   Max.   : 2.6519   Max.   : 2.2878
##       Age             Education            Income         FruitsOrVeggies
##  Min.   :-3.7424   Min.   :-7.06135   Min.   :-3.3906   0: 29653
##  1st Qu.:-0.3493   1st Qu.:-0.93007   1st Qu.:-0.1914   1:224027
##  Median : 0.1955   Median : 0.05685   Median : 0.4775
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.6180   3rd Qu.: 0.86322   3rd Qu.: 0.7429
##  Max.   : 1.1149   Max.   : 0.86322   Max.   : 0.7429
##  AnyHealthcareorNoDocbcCost
##  0:  7838
##  1:245842
##
##
##
##
```

```r
summary(raw_data2a)
```

```
##  Diabetes_012 HighBP     HighChol    CholCheck        BMI             Smoker    
##  0:213703     0:182738   0:183885   0:  6137   Min.   :-4.8904   0:180521  
##  1: 39977     1: 70942   1: 69795   1:247543   1st Qu.:-0.3267   1: 73159  
##                                                Median :-0.1039             
##                                                Mean   : 0.0000             
##                                                3rd Qu.: 0.3179             
##                                                Max.   : 7.5051             
##  Stroke     HeartDiseaseorAttack PhysActivity HvyAlcoholConsump
##  0:247048   0:238205             0: 40158     0:244439         
##  1:  6632   1: 15475             1:213522     1:  9241         
##                                                                
##                                                                
##                                                                
##                                                                
##     GenHlth           MentHlth          PhysHlth       DiffWalk   Sex       
##  Min.   :-2.0334   Min.   :-0.4448   Min.   :-0.4947   0:225832   0:181149  
##  1st Qu.:-0.2169   1st Qu.:-0.4448   1st Qu.:-0.4947   1: 27848   1: 72531  
##  Median :-0.2169   Median :-0.4448   Median :-0.4947                        
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000                        
##  3rd Qu.: 0.8456   3rd Qu.:-0.4448   3rd Qu.:-0.4947                        
##  Max.   : 2.1843   Max.   : 3.3454   Max.   : 2.9124                        
##       Age            Education            Income          FruitsOrVeggies
##  Min.   :-4.6946   Min.   :-8.77613   Min.   :-4.234592   0: 19268       
##  1st Qu.:-0.1541   1st Qu.: 0.04751   1st Qu.: 0.005647   1:234412       
##  Median : 0.1575   Median : 0.04751   Median : 0.370449                  
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.000000                  
##  3rd Qu.: 0.4324   3rd Qu.: 1.04707   3rd Qu.: 0.370449                  
##  Max.   : 1.2904   Max.   : 1.04707   Max.   : 0.686454                  
##  AnyHealthcareorNoDocbcCost
##  0:  5164                 
##  1:248516                 
##                           
##                           
##                           
##                           
```

As we can see the standardization was applied to the two data sets. One convention for outlier detection is to remove values with z-scores greater than 3 or less than -3. As we can see above, the BMI, Education and MentHlth columns have outliers that are either greater than 3 or less than -3. We can Calculate the number of observations that have a z-score either greater than 3 or less than -3. Additionally there are large number of observations with z-score values greater than 3 or less than -3 in both data sets. We determined in the data exploration that some feature in our data have skewed distributions. We remove them with the below code chunk

```r
# Function to count outliers based on Z-score
count_outliers <- function(column) {
  sum(column > 3 | column < -3, na.rm = TRUE)
}

# Apply the outlier counting function to specific columns
outliers_BMI <- count_outliers(raw_data1a$BMI)
outliers_Education <- count_outliers(raw_data1a$Education)
outliers_Income <- count_outliers(raw_data1a$Income)

# Print the number of outliers for each column
cat("Number of outliers in BMI in raw_data1:", outliers_BMI, "\n")
```

```
## Number of outliers in BMI in raw_data1: 1990
```

```r
cat("Number of outliers in Education in raw_data1:", outliers_Education, "\n")
```

```
## Number of outliers in Education in raw_data1: 4217
```

```r
cat("Number of outliers in Income in raw_data1:", outliers_Income, "\n")
```

```
## Number of outliers in Income in raw_data1: 9811
```

```r
# Function to count outliers based on Z-score
count_outliers <- function(column) {
  sum(column > 3 | column < -3, na.rm = TRUE)
}

# Apply the outlier counting function to specific columns
outliers2_BMI <- count_outliers(raw_data2a$BMI)
outliers2_Education <- count_outliers(raw_data2a$Education)
outliers2_MentHlth <- count_outliers(raw_data2a$MentHlth)
outliers2_inc <- count_outliers(raw_data2a$Income)
outliers2_age <- count_outliers(raw_data2a$Age)


# Print the number of outliers for each column
cat("Number of outliers in BMI in raw_data2:", outliers2_BMI, "\n")
```

```
## Number of outliers in BMI in raw_data2: 3469
```

```r
cat("Number of outliers in Education in raw_data2:", outliers2_Education, "\n")
```

```
## Number of outliers in Education in raw_data2: 2744
```

```r
cat("Number of outliers in MentHlth in raw_data2:", outliers2_MentHlth, "\n")
```

```
## Number of outliers in MentHlth in raw_data2: 9182
```

```r
cat("Number of outliers in Age in raw_data2:", outliers2_age, "\n")
```

```
## Number of outliers in Age in raw_data2: 8575
```

```r
cat("Number of outliers in Income in raw_data2:", outliers2_inc, "\n")
```

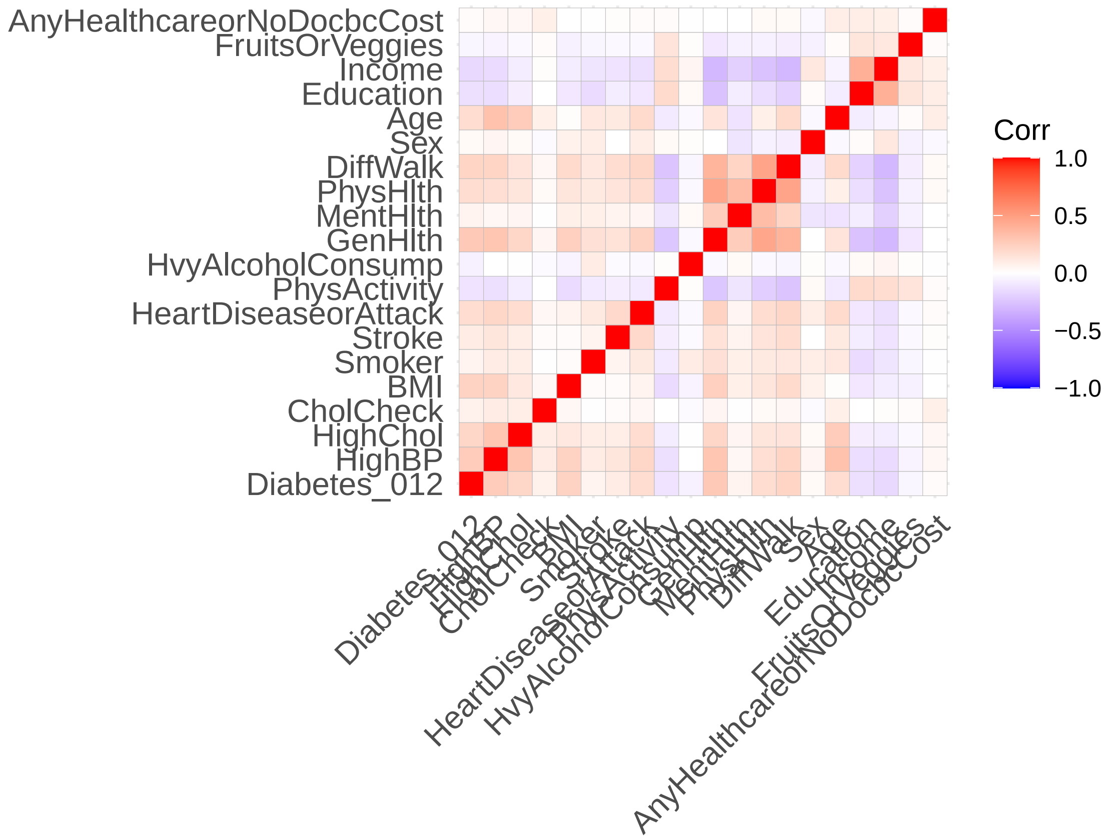
```
## Number of outliers in Income in raw_data2: 6439
```

Since the data is rightly skewed and also not normally distributed. It makes sense to keep in the outliers.

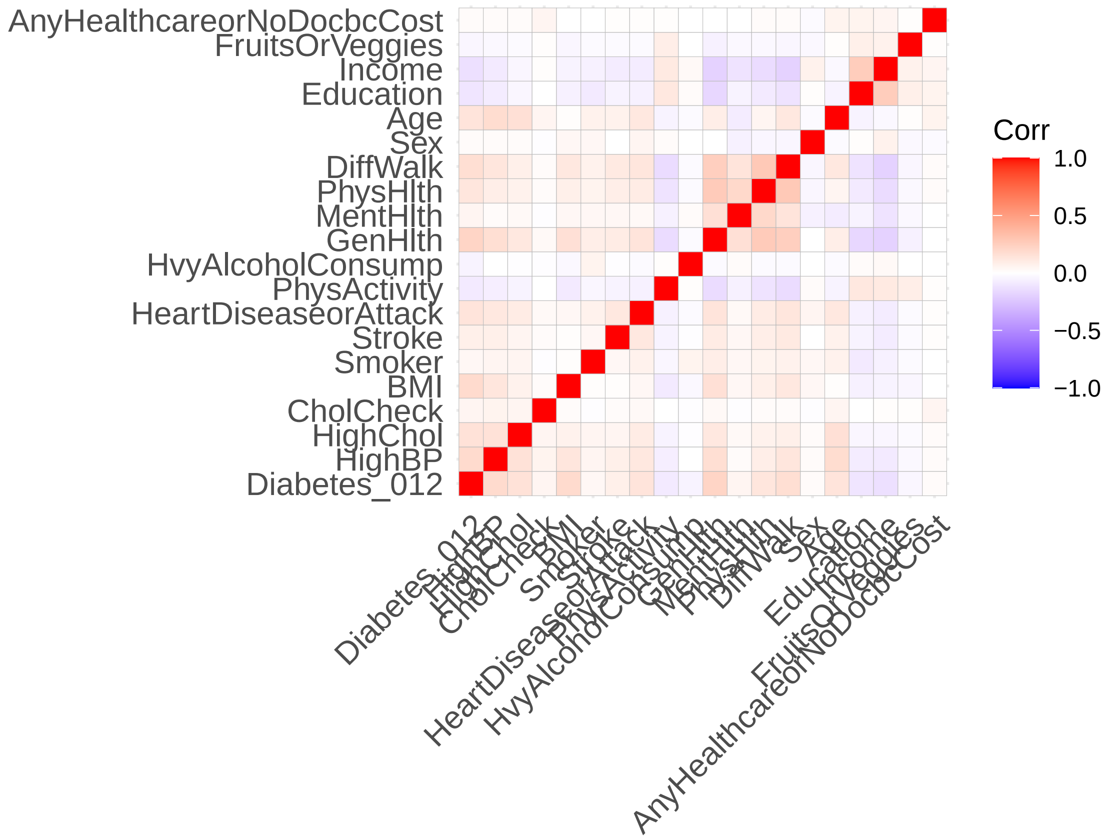
PCA

PCA (Principal Component Analysis) is a statistical technique that transforms a dataset into a set of orthogonal components which capture the most variance in the data.

```r
# Assuming raw_data is your data frame
library(stats)

# Corr matrix
corr_matrix <- cor(raw_data1a)
#install.packages("ggcorrplot")
library(ggcorrplot)
ggcorrplot(corr_matrix)
```




```
# Corr matrix
corr2_matrix <- cor(raw_data2a)
#install.packages("ggcorrplot")
library(ggcorrplot)
ggcorrplot(corr2_matrix)
```



As we can see from the above graph, from the PCA correlation, we can se that Diabetes has a pretty strong correlation with High Chol, High BP, Heart Disease Or Attack, Phys Hlt, DiffWalk and Age features. However if we go based on color intensity these are more prevelant in raw_data1 specifically with diabetes and Gen Hlth. Additionally there was a negative relationship with Phys Activity, Education and income. Howeveer the color intensity related to income was more prelevant in raw_data1 data set. This is because we imputed the values in raw_data2 with the median values.

```
data.pca <- princomp(corr_matrix)
summary(data.pca)
```

```
## Importance of components:
##                           Comp.1     Comp.2     Comp.3    Comp.4     Comp.5
## Standard deviation     0.6932957 0.33079950 0.27380468 0.2600699 0.23287079
## Proportion of Variance 0.4039992 0.09197571 0.06301228 0.0568491 0.04557992
## Cumulative Proportion  0.4039992 0.49597488 0.55898716 0.6158363 0.66141618
##                            Comp.6     Comp.7     Comp.8     Comp.9    Comp.10
## Standard deviation     0.22278356 0.21329666 0.20233265 0.19388912 0.19129559
## Proportion of Variance 0.04171668 0.03823945 0.03440927 0.03159733 0.03075767
## Cumulative Proportion  0.70313286 0.74137230 0.77578157 0.80737890 0.83813657
##                           Comp.11   Comp.12    Comp.13    Comp.14    Comp.15
## Standard deviation     0.17512946 0.1675962 0.16652088 0.15916231 0.14653460
## Proportion of Variance 0.02577875 0.0236087 0.02330671 0.02129237 0.01804778
## Cumulative Proportion  0.86391532 0.8875240 0.91083072 0.93212309 0.95017087
##                           Comp.16    Comp.17    Comp.18     Comp.19 Comp.20
```

```
## Standard deviation     0.13269221 0.13100872 0.11647809 0.104626303       0
## Proportion of Variance 0.01479907 0.01442593 0.01140334 0.009200793       0
## Cumulative Proportion  0.96496994 0.97939587 0.99079921 1.000000000       1
```

```r
data.pca2 <- princomp(corr2_matrix)
summary(data.pca2)
```

```
## Importance of components:
##                           Comp.1     Comp.2     Comp.3     Comp.4    Comp.5
## Standard deviation     0.5136836 0.28763505 0.25191278 0.24693578 0.2285018
## Proportion of Variance 0.2574902 0.08073334 0.06192552 0.05950279 0.0509505
## Cumulative Proportion  0.2574902 0.33822357 0.40014909 0.45965189 0.5106024
##                            Comp.6     Comp.7     Comp.8     Comp.9    Comp.10
## Standard deviation     0.22333935 0.21709732 0.21080532 0.20621105 0.20343149
## Proportion of Variance 0.04867431 0.04599157 0.04336431 0.04149475 0.04038366
## Cumulative Proportion  0.55927669 0.60526826 0.64863257 0.69012732 0.73051098
##                           Comp.11    Comp.12   Comp.13    Comp.14    Comp.15
## Standard deviation     0.19438355 0.19061845 0.1886751 0.18151259 0.17171660
## Proportion of Variance 0.03687129 0.03545677 0.0347375 0.03215014 0.02877358
## Cumulative Proportion  0.76738226 0.80283903 0.8375765 0.86972667 0.89850025
##                          Comp.16    Comp.17    Comp.18    Comp.19      Comp.20
## Standard deviation     0.1686505 0.16632716 0.15700025 0.15250621 2.671087e-09
## Proportion of Variance 0.0277552 0.02699577 0.02405304 0.02269574 6.962183e-18
## Cumulative Proportion  0.9262555 0.95325122 0.97730426 1.00000000 1.000000e+00
```

As we can see there is a significant differnce with the standard deviation, proporiton of variance and cumulative proprotion value between the two data sets. Again this could be due to the missing values we replaced.

Standard Deviation: This value for each principal component (PC) indicates the amount of variance captured by that component. Higher values mean more variance is captured.

Proportion of Variance: This shows the fraction of the total variance in the dataset that is captured by each PC.

Cumulative Proportion: This indicates the cumulative variance explained by the PCs up to that point.

as we can see here raw_data1 has more proportion of variance compared to raw_data2.

```r
data.pca$loadings[, 1:12]
```

```
##                                Comp.1     Comp.2        Comp.3      Comp.4
## Diabetes_012              0.227062943 0.20907474  5.409179e-02  0.24287265
## HighBP                    0.235029796 0.35002494  2.748727e-02  0.06299737
## HighChol                  0.163958929 0.32149233  5.742283e-02 -0.02827675
## CholCheck                 0.004931334 0.12423838  2.811794e-01  0.05744183
## BMI                       0.174757941 0.06036556 -4.412762e-02  0.60853346
```

```
## Smoker                       0.119322021 -0.01579100 -4.474700e-01 -0.31623839
## Stroke                       0.139093553  0.02613731  8.834851e-05 -0.28881526
## HeartDiseaseorAttack         0.184878783  0.17619045 -8.138620e-02 -0.20540421
## PhysActivity                -0.286420867 -0.01102284  2.908709e-02 -0.07203644
## HvyAlcoholConsump           -0.073930956 -0.17154160 -3.461106e-01 -0.24910227
## GenHlth                      0.373698706 -0.09341589 -2.497790e-03  0.10081718
## MentHlth                     0.186539511 -0.51504543  5.740741e-02  0.05130022
## PhysHlth                     0.320155747 -0.31279481  1.398328e-01 -0.01197250
## DiffWalk                     0.348079235 -0.13668321  1.367174e-01 -0.04507941
## Sex                         -0.063919332  0.17893218 -5.479020e-01  0.27703140
## Age                          0.134946504  0.46011548  1.518471e-01 -0.32419231
## Education                   -0.315874614 -0.05202408  1.993569e-01  0.11805477
## Income                      -0.363439764  0.09880662  2.035805e-02  0.15336721
## FruitsOrVeggies             -0.160060571 -0.03675095  2.538534e-01 -0.17904842
## AnyHealthcareorNoDocbcCost -0.051001715  0.04178638  3.392938e-01 -0.05868482
##                                  Comp.5       Comp.6       Comp.7      Comp.8
## Diabetes_012                 0.021294446  0.180847656  0.024950706  0.05824429
## HighBP                      -0.134658809  0.105588087  0.075113521  0.14845338
## HighChol                    -0.147988218  0.053459497  0.116084450  0.20660446
## CholCheck                   -0.205294527 -0.080164945 -0.816420877  0.03680469
## BMI                         -0.093063203  0.141790778  0.065046746  0.02820924
## Smoker                      -0.218367229 -0.008541256 -0.046487702 -0.44598720
## Stroke                       0.502007127 -0.179995688 -0.105363842  0.39360535
## HeartDiseaseorAttack         0.388325463 -0.132318283 -0.014220709  0.03474015
## PhysActivity                 0.197657512  0.287269685 -0.228469239 -0.01111807
## HvyAlcoholConsump           -0.503790195  0.058405978  0.052959648  0.48022755
## GenHlth                      0.001695496  0.022594093  0.022535234 -0.06380385
## MentHlth                    -0.038076913 -0.016931278 -0.008124261  0.08388144
## PhysHlth                     0.066792276 -0.077523995  0.102561741 -0.08998629
## DiffWalk                     0.048431252 -0.047853025  0.118080444 -0.08127622
## Sex                          0.252204274 -0.188109271 -0.077196980 -0.24235243
## Age                         -0.141170522  0.002536227  0.209570314 -0.06398123
## Education                    0.082644182 -0.192692343  0.309199398  0.17560638
## Income                      -0.019426437 -0.193321956  0.220229532  0.04805834
## FruitsOrVeggies              0.108204016  0.607697406  0.134877868 -0.31795114
## AnyHealthcareorNoDocbcCost -0.233730888 -0.550048391  0.061690489 -0.34795961
##                                  Comp.9     Comp.10     Comp.11      Comp.12
## Diabetes_012                 0.041954033  0.23371965  0.04116824  0.473159410
## HighBP                      -0.057403901  0.06355853  0.01123577 -0.108111576
## HighChol                    -0.525256958 -0.07798791 -0.10568606 -0.448844944
## CholCheck                    0.091861014 -0.35151361  0.02230429 -0.007462301
## BMI                          0.283053351  0.15589896 -0.17604522 -0.134919525
## Smoker                      -0.042066859 -0.08377555 -0.43873553  0.119505660
## Stroke                       0.327293031  0.19419595 -0.45313979 -0.172651410
## HeartDiseaseorAttack        -0.051785204 -0.08493176  0.56144640 -0.044345167
## PhysActivity                -0.418759389  0.42377098  0.03184620  0.289928691
## HvyAlcoholConsump            0.252984462  0.10270147  0.33954725  0.003535406
## GenHlth                     -0.032345279 -0.02652520  0.07590054 -0.006786430
```

```
## MentHlth                    -0.370561718  0.01164616 -0.08492878 -0.203493053
## PhysHlth                    -0.078316498 -0.20397917  0.10871859  0.114168014
## DiffWalk                     0.160991384 -0.15141211  0.06489410  0.226614294
## Sex                         -0.023745815 -0.09434198  0.20588077 -0.177695870
## Age                         -0.006827045 -0.12634107 -0.07042887  0.227331174
## Education                   -0.084681335 -0.29317546 -0.07868063  0.194242760
## Income                       0.044443507 -0.28186707 -0.09917491  0.029756840
## FruitsOrVeggies              0.302470237 -0.09770123  0.12035787 -0.384779146
## AnyHealthcareorNoDocbcCost   0.088886106  0.52927547  0.13893408 -0.197011963
```

```
data.pca2$loadings[, 1:14]
```

```
##                                  Comp.1      Comp.2       Comp.3      Comp.4
## Diabetes_012                0.277936260  0.22839763  0.052301558  0.21950641
## HighBP                      0.212388431  0.32992781  0.006710672  0.05787807
## HighChol                    0.143344919  0.29559633  0.024774999 -0.03926210
## CholCheck                  -0.005097243  0.12450328  0.279895642  0.04540583
## BMI                         0.173325165  0.06040636 -0.039332762  0.61419149
## Smoker                      0.093798967 -0.05378514 -0.453737761 -0.31437016
## Stroke                      0.132706715  0.01573645 -0.010721086 -0.29736216
## HeartDiseaseorAttack        0.178327222  0.16541392 -0.092676790 -0.21124663
## PhysActivity               -0.294069952 -0.02099401  0.019422030 -0.06421824
## HvyAlcoholConsump          -0.090387085 -0.20896732 -0.358243111 -0.21670439
## GenHlth                     0.369773410 -0.09916291  0.004419938  0.10278329
## MentHlth                    0.164716003 -0.53231980  0.070132582  0.06001846
## PhysHlth                    0.296018414 -0.32174942  0.159693910 -0.01486518
## DiffWalk                    0.335963288 -0.14939010  0.141691104 -0.04492582
## Sex                        -0.073027622  0.14806778 -0.539356188  0.30120440
## Age                         0.128967925  0.45009368  0.120230422 -0.33212922
## Education                  -0.335283272 -0.04656694  0.198252783  0.12115900
## Income                     -0.375618164  0.09141608  0.022846782  0.15789784
## FruitsOrVeggies            -0.173869522 -0.03974837  0.250853725 -0.14879656
## AnyHealthcareorNoDocbcCost -0.064094734  0.04683164  0.338805308 -0.07401643
##                                  Comp.5       Comp.6      Comp.7       Comp.8
## Diabetes_012                0.003085207  0.127596105  0.03711339  0.016036161
## HighBP                     -0.146949374  0.111367720  0.11206457  0.138846079
## HighChol                   -0.154904862  0.055523675  0.15581617  0.204655922
## CholCheck                  -0.229838811  0.006681656 -0.80248201  0.155640526
## BMI                        -0.103149280  0.124800619  0.07107794 -0.001846616
## Smoker                     -0.214741612  0.002040630 -0.09587054 -0.474638763
## Stroke                      0.485694445 -0.156486794 -0.10796415  0.408916031
## HeartDiseaseorAttack        0.382731845 -0.096204119 -0.02985794  0.042464124
## PhysActivity                0.172424876  0.350126123 -0.20146373 -0.033647004
## HvyAlcoholConsump          -0.503629676  0.026921835  0.12098575  0.455980097
## GenHlth                    -0.001261793  0.027717194  0.01204671 -0.058723593
## MentHlth                   -0.024118262  0.001538520 -0.01101086  0.074363975
## PhysHlth                    0.084605591 -0.087683254  0.08376881 -0.080459302
```

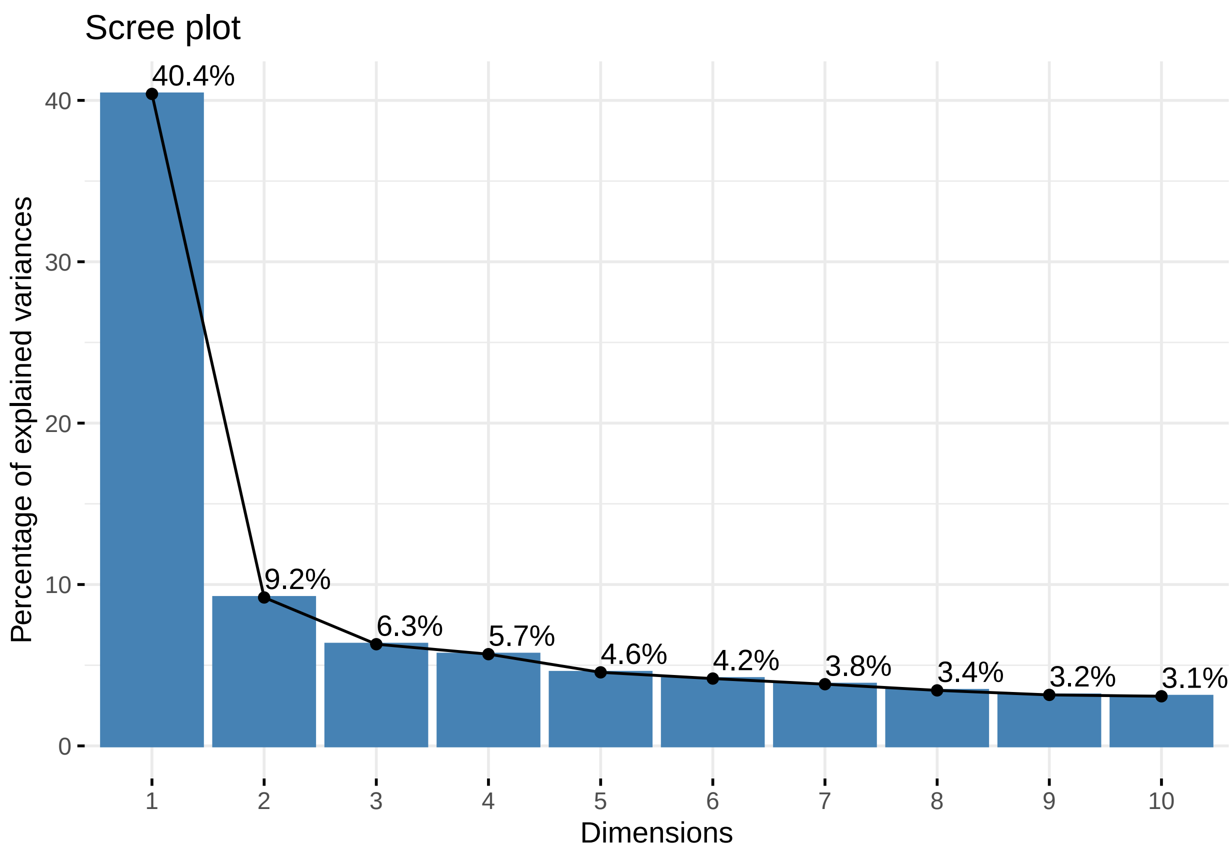
```
## DiffWalk                     0.056537322 -0.063898865  0.11900726 -0.081649122
## Sex                          0.286871947 -0.173030787 -0.14863872 -0.195166234
## Age                         -0.125053881 -0.017313360  0.20427759 -0.089572902
## Education                    0.103642577 -0.204627058  0.29215175  0.143174399
## Income                       0.002049554 -0.217371384  0.20093667  0.042403244
## FruitsOrVeggies              0.109709512  0.601159687  0.15181492 -0.287466474
## AnyHealthcareorNoDocbcCost -0.217741315 -0.549778312 -0.04179502 -0.371267976
##                                   Comp.9     Comp.10      Comp.11      Comp.12
## Diabetes_012                 0.002717322  0.14654109  0.012378517  0.004751096
## HighBP                      -0.031016557  0.13966842 -0.002887686  0.222228306
## HighChol                    -0.549199139 -0.19136795  0.017547042 -0.523099620
## CholCheck                    0.133217173 -0.33601330 -0.008126521  0.009706594
## BMI                          0.225871322  0.22042859  0.242658280 -0.061009320
## Smoker                      -0.033414941 -0.13110490  0.482863912 -0.014648310
## Stroke                       0.241719925  0.27109235  0.444953629 -0.200738851
## HeartDiseaseorAttack        -0.010418194 -0.08413233 -0.449459718 -0.053624879
## PhysActivity                -0.453734773  0.40484345 -0.021599208  0.406044723
## HvyAlcoholConsump            0.257627248  0.15964136 -0.315548288  0.016322954
## GenHlth                     -0.016306714 -0.01755755 -0.077631060  0.043553810
## MentHlth                    -0.357308268 -0.03898602  0.039461437 -0.182459971
## PhysHlth                    -0.031199655 -0.20437963 -0.121683642  0.216119604
## DiffWalk                     0.173072241 -0.12174206 -0.037806821  0.247053144
## Sex                          0.015967691 -0.10364461 -0.304898562 -0.082423025
## Age                          0.019273792 -0.12276732  0.032606106  0.291329210
## Education                   -0.045515803 -0.25796580  0.130874199  0.151242427
## Income                       0.076942447 -0.26186107  0.146161807  0.034501484
## FruitsOrVeggies              0.359494376 -0.06725943 -0.148486056 -0.388685454
## AnyHealthcareorNoDocbcCost   0.005343787  0.49988589 -0.170302963 -0.234261846
##                                  Comp.13      Comp.14
## Diabetes_012                 0.157827514  0.178632977
## HighBP                      -0.088567624 -0.725967961
## HighChol                    -0.147403678  0.272387713
## CholCheck                    0.016071573 -0.011866602
## BMI                          0.236142137  0.163925241
## Smoker                       0.236481696 -0.022832492
## Stroke                      -0.187304824 -0.003339796
## HeartDiseaseorAttack         0.680346962 -0.040156093
## PhysActivity                -0.012059535  0.204706199
## HvyAlcoholConsump           -0.001017218  0.137272488
## GenHlth                     -0.033428435  0.033922984
## MentHlth                     0.061488993 -0.414694157
## PhysHlth                    -0.208775160  0.121349773
## DiffWalk                    -0.142652741  0.257697603
## Sex                         -0.445486179 -0.078807077
## Age                         -0.167372411  0.035544762
## Education                    0.141789436 -0.045157646
## Income                       0.098833367 -0.017156684
## FruitsOrVeggies             -0.099164790 -0.111824454
```

```
## AnyHealthcareorNoDocbcCost -0.021429769 -0.019798131
```

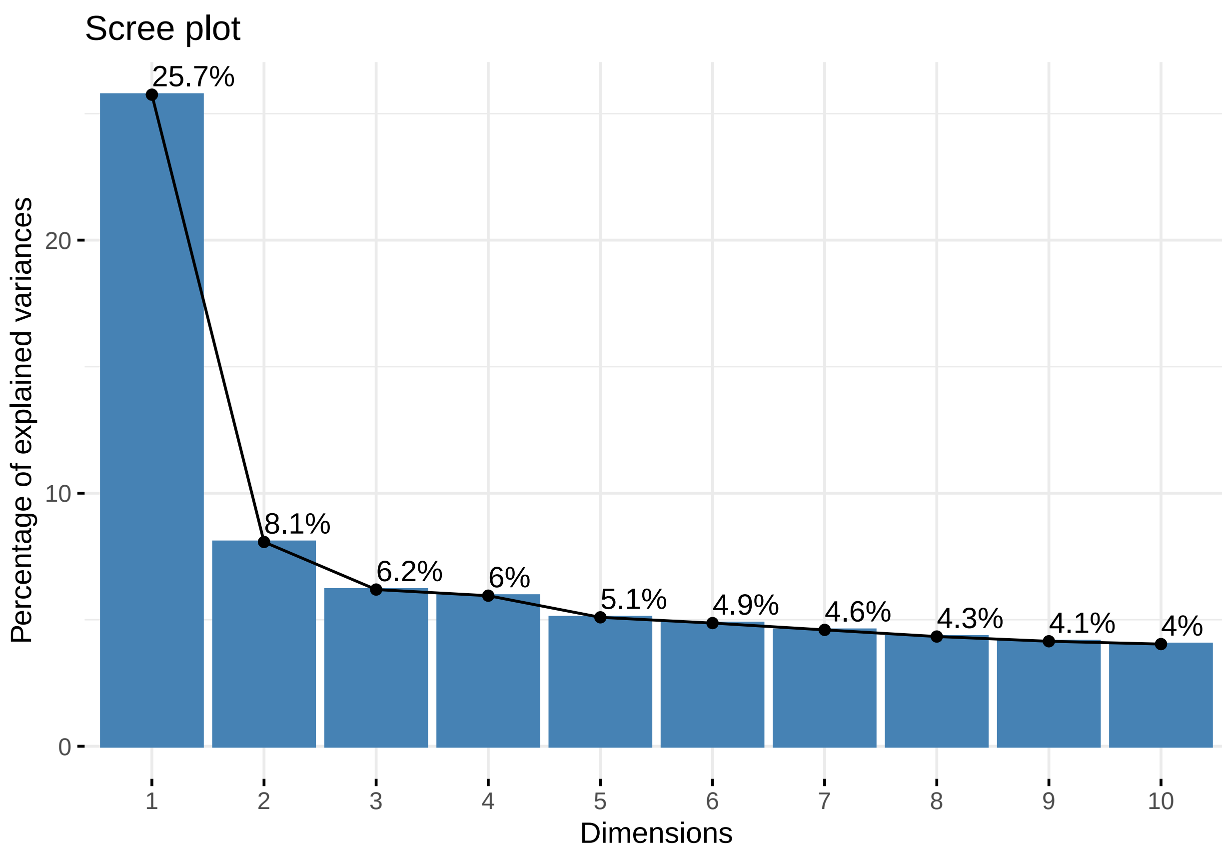
A higher absolute value of a loading indicates that the corresponding original variable has a stronger influence on that principal component.

Positive values suggest a direct correlation with the principal component, while negative values indicate an inverse correlation. For instance, GenHlth has a high positive loading (0.379951917) (0.303198017) on Comp.1 in both data_sets meaning it strongly influences and is positively correlated with Comp.1 in both data sets.

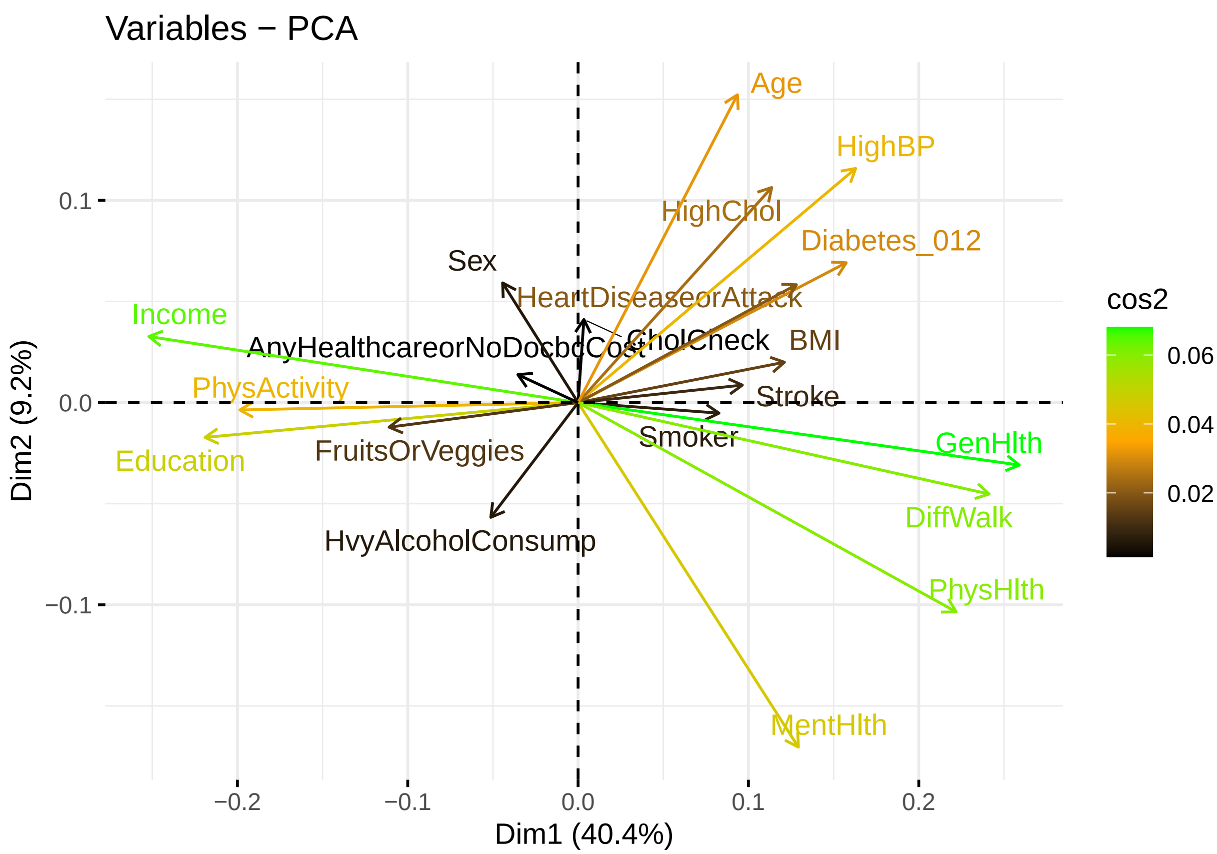
```r
fviz_eig(data.pca, addlabels = TRUE)
```



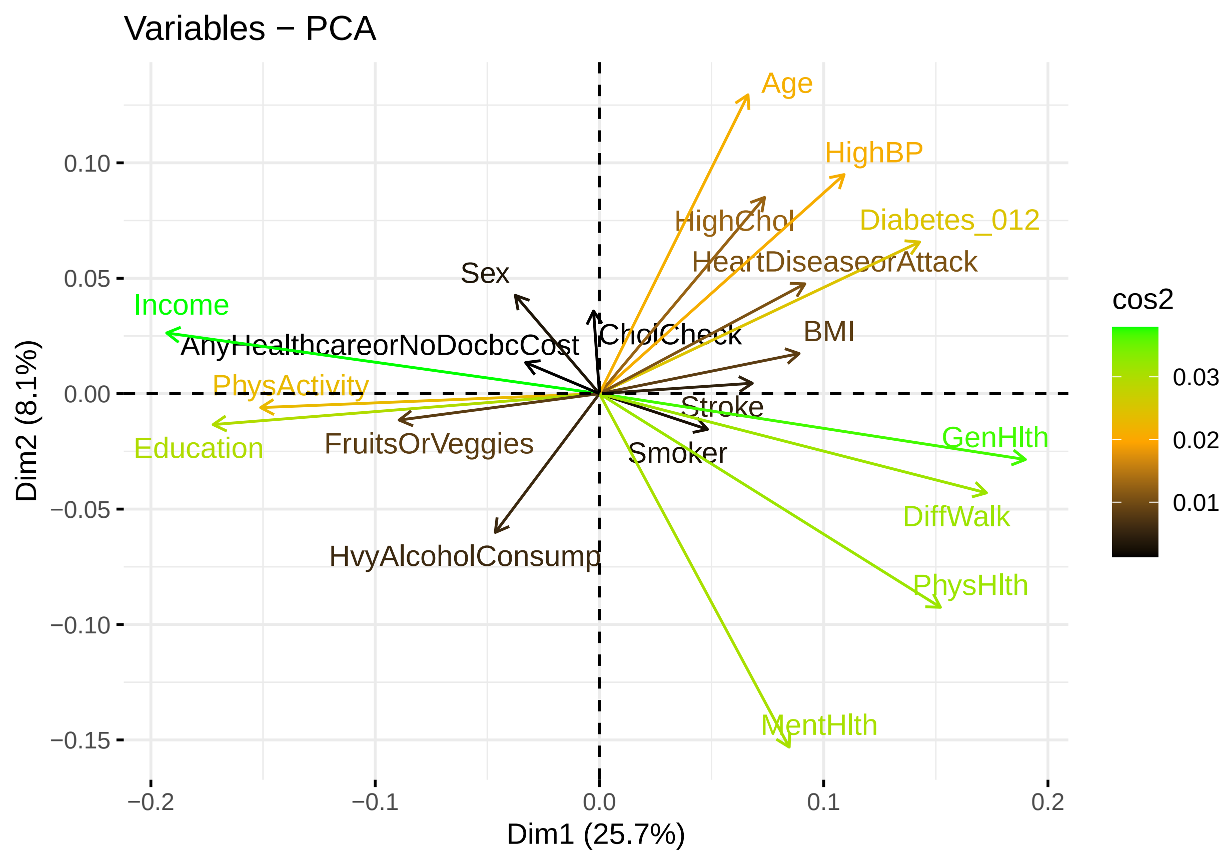

```r
# Install factoextra package if not already installed
if (!requireNamespace("factoextra", quietly = TRUE)) {
    install.packages("factoextra")
}
library(factoextra)
fviz_eig(data.pca2, addlabels = TRUE)
```



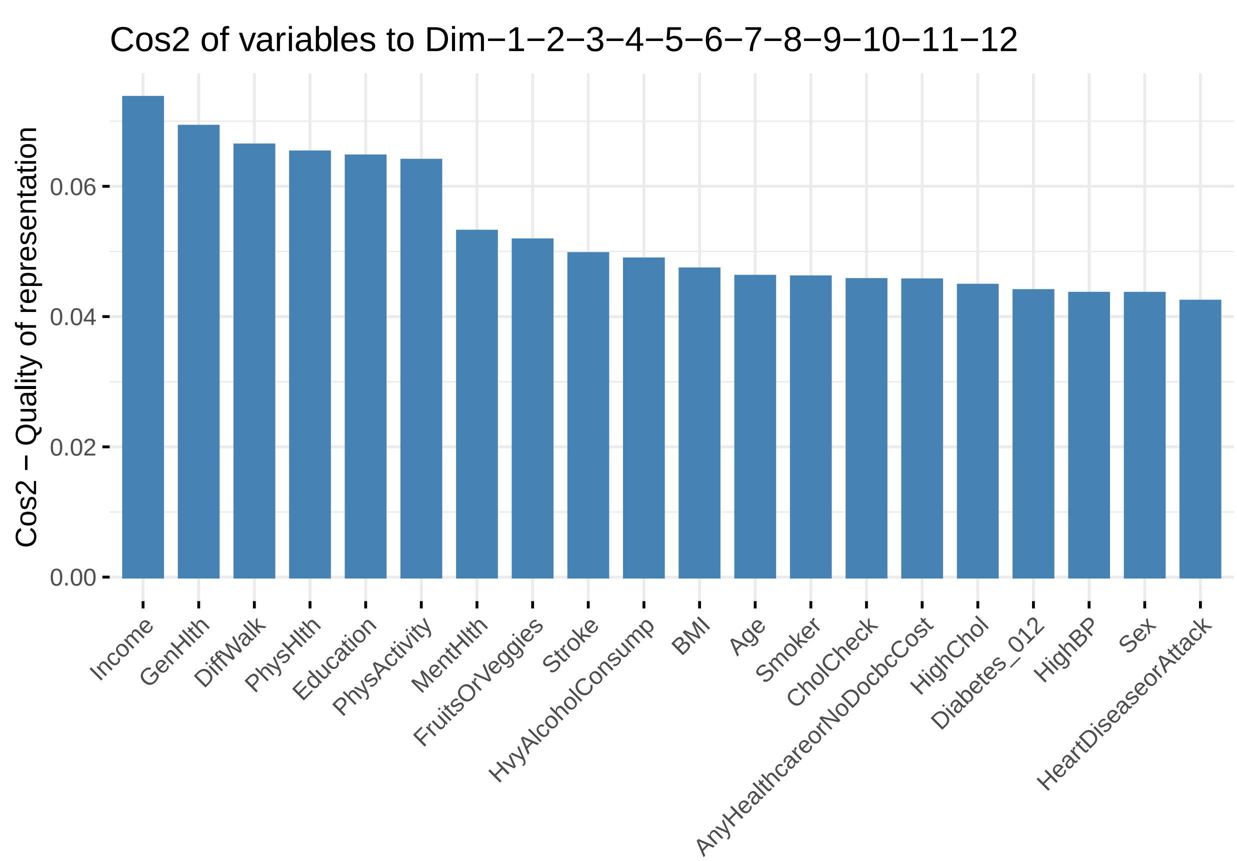

```r
# Graph of the variables
fviz_pca_var(data.pca, col.var = "cos2",
            gradient.cols = c("black", "orange", "green"),
            repel = TRUE)
```



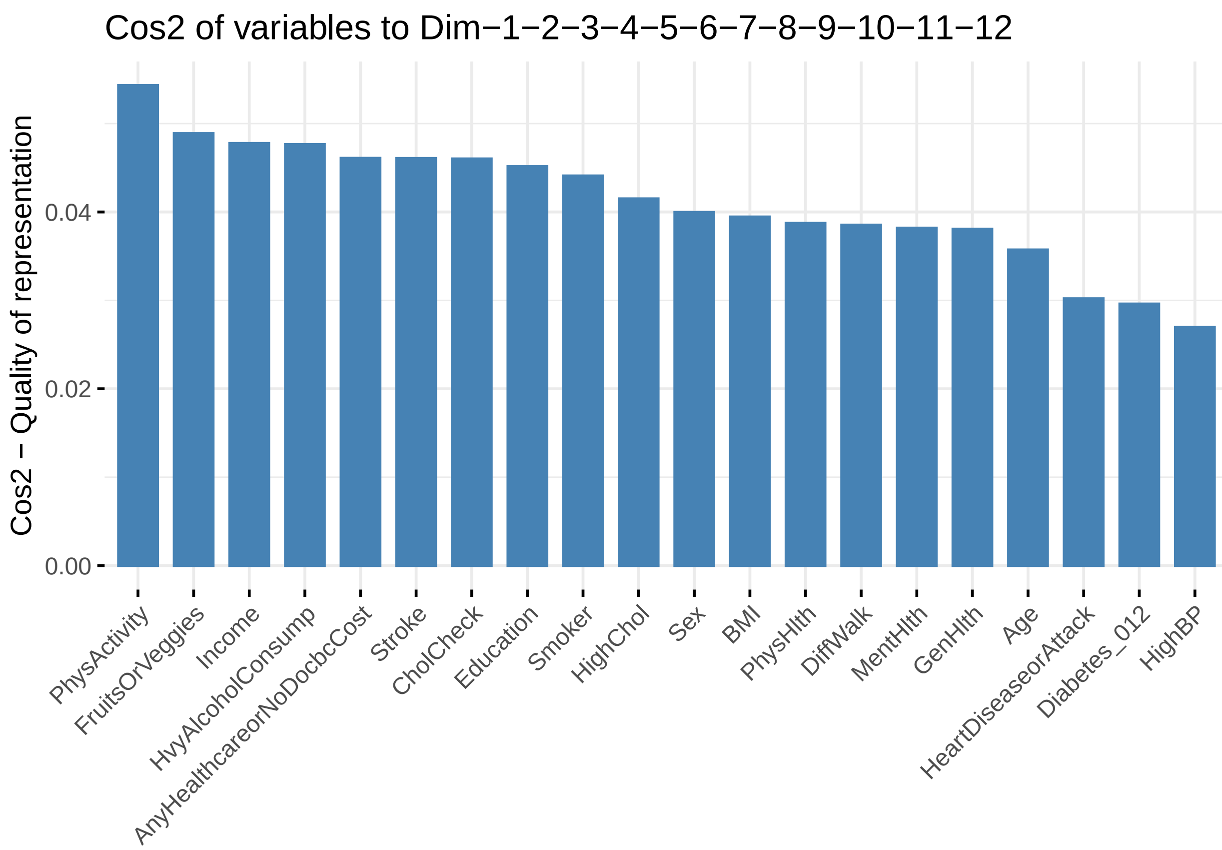

```r
# Graph of the variables
fviz_pca_var(data.pca2, col.var = "cos2",
             gradient.cols = c("black", "orange", "green"),
             repel = TRUE)
```




```r
fviz_cos2(data.pca, choice = "var", axes = 1:12)
```



```
fviz_cos2(data.pca2, choice = "var", axes = 1:12)
```



Interstingly enough the quality of representation in both the data set differs. With raw_data1, we see that GenHlth, Income are the two best represented features in the data. But in raw_data2 we see that PhysActivity and Income are the two best. It is an interesting observation and the only explanation we can come up with for the difference is probably because of the missing data we imputed in the raw_data2 dataset. Also the overall quality of representation is low in both data sets.

For the classification model, we are going to use the data sets raw_data1, raw_data2. Usually one would use the the Standardized data. However considering the fact that certain outliers for example in the BMI column are usefull in the diagnosis of Diabetes. For example its general knowledge that the higher the BMI the higher the chance of being diagnosed Diabetes. Additionally factors like old age is known to effect diagnosis of diabetes, for example and older age has a higher chance of being diagnosed. These type of outliers and variance are required in this case to perform an accurate diagnosis. However seeing the effect of missing values does affect the prediction. Therefore we will be producing two models to see as to how they are affecting prediction.

```
# Convert to factors
columns_to_convert <- c("Diabetes_012")

for (column_name in columns_to_convert) {
  raw_data1a[[column_name]] <- factor(raw_data1a[[column_name]])
}
```

```
for (column_name in columns_to_convert) {
  raw_data2a[[column_name]] <- factor(raw_data2a[[column_name]])
}
```

feature engineering: removal of unnecessary features

Before we start we can see that there are some features that are not required for our prediction. Features such as FruitsORVeggies, AnyHealthcareorNoDocbcCost, Chol check, HvyAlcoholConsump. Additionally if we were to use our analysis from the PCA, we can see that MentHlth also does not play a major affect in predicting diabetes. I believe that these columns can be removed from the data set. The binned data will be used for our naive bayes modelling

```
raw_data1a$AnyHealthcareorNoDocbcCost <- NULL
raw_data1a$CholCheck <- NULL
raw_data1a$HvyAlcoholConsump <- NULL
raw_data1a$MentHlth <- NULL
```

```
head(raw_data1a)
```

```
##   Diabetes_012 HighBP HighChol         BMI Smoker Stroke HeartDiseaseorAttack
## 1            0      1        1  1.74823388      1      0                    0
## 2            0      0        0 -0.49512169      1      0                    0
## 3            0      1        1  0.04580293      0      0                    0
## 4            0      1        0 -0.12778202      0      0                    0
## 5            0      1        1 -0.68996753      0      0                    0
## 6            0      1        1 -0.49512169      1      0                    0
##   PhysActivity    GenHlth   PhysHlth DiffWalk Sex        Age   Education
## 1            0  1.6886421  1.7203376        1   0  0.4185072 -0.93006552
## 2            1  0.5957167 -0.6585269        0   0 -0.0574126  0.86321974
## 3            0  1.6886421  2.2878136        1   0  0.4185072 -0.93006552
## 4            1 -0.2717870 -0.6585269        0   0  0.7985218 -2.20242167
## 5            1 -0.2717870 -0.6585269        0   0  0.7985218  0.05685057
## 6            1 -0.2717870  0.2840759        0   1  0.6180307  0.86321974
##       Income FruitsOrVeggies
## 1 -1.2067709               1
## 2 -3.3905566               0
## 3  0.7428894               1
## 4  0.1710444               1
## 5 -0.6349259               1
## 6  0.7428894               1
```

```
raw_data2a$AnyHealthcareorNoDocbcCost <- NULL
raw_data2a$CholCheck <- NULL
raw_data2a$HvyAlcoholConsump <- NULL
raw_data2a$MentHlth <- NULL
```

```r
head(raw_data2a)
```

```
##   Diabetes_012 HighBP HighChol        BMI Smoker Stroke HeartDiseaseorAttack
## 1            0      1        0  2.2160064      1      0                    0
## 2            0      0        0 -0.5581689      1      0                    0
## 3            0      1        0 -0.1039099      0      0                    0
## 4            0      1        0 -0.1039099      0      0                    0
## 5            0      1        1 -0.7991189      0      0                    0
## 6            0      1        1 -0.1039099      1      0                    0
##   PhysActivity    GenHlth   PhysHlth DiffWalk Sex        Age   Education
## 1            0  2.1842617  2.2561835        1   0  0.4323543  0.04750712
## 2            1 -0.2169301 -0.4946554        0   0 -0.1540546  1.04707250
## 3            0  2.1842617 -0.4946554        0   0  0.4323543 -1.17586203
## 4            1 -0.2169301 -0.4946554        0   0  0.9005928  0.04750712
## 5            1 -0.2169301 -0.4946554        0   0  0.9005928  0.04750712
## 6            1 -0.2169301 -0.4946554        0   0  0.6781991  1.04707250
##         Income FruitsOrVeggies
## 1 -1.634701031               1
## 2  0.370448513               0
## 3  0.370448513               1
## 4  0.005647443               1
## 5 -0.953894902               1
## 6  0.370448513               1
```

We have now removed the unnecessary columns

Modeling

We will train and compare three machine learning classification models for predicting diabetes diagnosis The classification algorithms I will use are Naive Bayes, Decision Trees, and Logistic Regression.

Appropriateness: Logistic Regression and Decision Trees are chosen for their interpretability, which is valuable in a medical context. Naive Bayes is chosen for its simplicity and effectiveness in baseline comparisons.

For our modelling we will be using classification models such as Naive Bayes, Logistic Regression and Decision Tree. These models are suitable for this data set because the data set is large. Naive Bayes is known for its high-dimensional data and can be particularly fast. Logistic and Decision Trees are highly effective for their robustness. Additionally Decision Tree handles imbalanced data the best.

Training and testing data set

Prior to the model training phase, it's essential to segregate the dataset into separate training and testing subsets. The purpose of this is to train models using the training subset and then assess their performance with the testing subset. Utilizing the entire dataset for both training and testing can result in overfitting, where the model performs well on the data it has seen but poorly on new, unseen data.

The caret package in R provides a useful function called createDataPartition(). This function is designed to split data into balanced subsets, particularly useful when you have a specific factor that needs balanced representation. In this scenario, we aim to partition the cervical cancer dataset into two parts: 75% for training and 25% for testing, ensuring that the age distribution is well-represented in both subsets.

```
# Define the split_data function
split_data <- function(data, target_column, p = 0.75, list = FALSE) {
  # Create indices for the training set
  train_indices <- createDataPartition(data[[target_column]], p = p, list = list)

  # Split the data into training and testing sets
  train_set <- data[train_indices, ]
  test_set <- data[-train_indices, ]

  # Return a list containing the training and testing sets
  return(list(train = train_set, test = test_set))
}

# Ensure 'Diabetes_012' is a factor with at least two levels
raw_data1a$Diabetes_012 <- factor(raw_data1a$Diabetes_012)

# Check the structure of 'Diabetes_012' to ensure it's a factor with two levels
str(raw_data1a$Diabetes_012)
```

```
##  Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
```

```
# If 'Diabetes_012' is confirmed to have two levels, you can proceed with the splitting
split_result <- split_data(raw_data1a, "Diabetes_012")

# Extract the training and testing data
train_data <- split_result$train
test_data <- split_result$test

# Optionally, check the dimensions of the split datasets
dim(train_data)
```

```
## [1] 190261     16
```

```
dim(test_data)
```

```
## [1] 63419    16
```

```r
# Apply function to raw_data2

# Ensure 'Diabetes_012' is a factor with at least two levels
raw_data2a$Diabetes_012 <- factor(raw_data2a$Diabetes_012)

# Check the structure of 'Diabetes_012' to ensure it's a factor with two levels
str(raw_data2a$Diabetes_012)
```

```
##  Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
```

```r
# If 'Diabetes_012' is confirmed to have two levels, you can proceed with the splitting
split_result2 <- split_data(raw_data2a, "Diabetes_012")

# Extract the training and testing data
train_data2 <- split_result2$train
test_data2 <- split_result2$test

# Optionally, check the dimensions of the split datasets
dim(train_data2)
```

```
## [1] 190261     16
```

```r
dim(test_data2)
```

```
## [1] 63419    16
```

Smote

As you can probably see, the data set is imbalanced. We can see that in both data sets, there were a lot ore no diabetes recorded compared to diabetes (about 84% to 16%). Normally one would use the Smote function. However we need to consider if balancing the data set is better or not. While it's clear that heavily imbalanced datasets present challenges for learning algorithms, the optimal strategy for managing this imbalance remains debatable. Some even suggest that the most effective strategy might be to take no action at all. The core issue revolves around whether artificially balancing the dataset genuinely enhances a learning algorithm's overall performance, or if it merely shifts the balance from reducing specificity to increasing sensitivity. Given that a learning algorithm trained on a balanced dataset will ultimately be applied to the original, imbalanced dataset, the act of balancing might just be altering the algorithm's perception of the cost associated with different types of errors. Consequently, it seems paradoxical to believe that discarding data could lead to a more intelligent model – that is, one more adept at accurately distinguishing between different outcomes [1].

Naive Bayes Modelling

```r
nb_model <- NaiveBayes(Diabetes_012 ~ ., data = train_data, laplace = 1)

# Making predictions on the validation set without showing warnings
predictions <- suppressWarnings(predict(nb_model, newdata = test_data))
```

```r
nb_model2 <- NaiveBayes(Diabetes_012 ~ ., data = train_data2, laplace = 1)

# Making predictions on the validation set without showing warnings
predictions2 <- suppressWarnings(predict(nb_model2, newdata = test_data2))
```

Evaluation of Naive Bayes models

```r
# Creating the confusion matrix
the_matrix_is_real <- table(predictions$class, test_data$Diabetes_012)
the_matrix_is_real
```

```
##
##         0     1
##   0 44177  4760
##   1  9248  5234
```

```r
d <- CrossTable(the_matrix_is_real,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     44177 |      4760 |     48937 |
##              |     0.697 |     0.075 |           |
## -------------|-----------|-----------|-----------|
```

```
##            1 |      9248 |      5234 |     14482 |
##              |     0.146 |     0.083 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```
d
```

```
## $t
##
##         0     1
##   0 44177  4760
##   1  9248  5234
##
## $prop.row
##
##              0          1
##   0 0.90273208 0.09726792
##   1 0.63858583 0.36141417
##
## $prop.col
##
##             0         1
##   0 0.8268975 0.4762858
##   1 0.1731025 0.5237142
##
## $prop.tbl
##
##              0          1
##   0 0.69658935 0.07505637
##   1 0.14582381 0.08253047
```

```
true_negative <- the_matrix_is_real[1, 1]  # Correctly predicted No
true_positive <- the_matrix_is_real[2, 2]  # Correctly predicted Yes
false_positive <- the_matrix_is_real[2, 1] # Incorrectly predicted Yes
false_negative <- the_matrix_is_real[1, 2] # Incorrectly predicted No

accuracy <- (true_positive + true_negative) / (true_negative+true_positive+false_positive+fa

precision_NB<-round(d$t[1]/ sum(d$t[3],d$t[1]),3)

recall <- true_negative / (true_negative+true_positive+false_positive+false_negative)
f1_dec <- 2 * (precision_NB * recall) / (precision_NB + recall)
cat("Accuracy score for Naive Bayes model for raw_data1:", round(accuracy * 100, 2), "%\n")
```

```
## Accuracy score for Naive Bayes model for raw_data1: 77.91 %
```

```r
cat("Precision score for Naive Bayes model for raw_data1:", round(precision_NB * 100, 2), "
```

```
## Precision score for Naive Bayes model for raw_data1: 90.3 %
```

```r
cat("Recall score for Naive Bayes model for raw_data1:", round(recall * 100, 2), "%\n")
```

```
## Recall score for Naive Bayes model for raw_data1: 69.66 %
```

```r
cat("F1 score for Naive Bayes model for raw_data1:", round(f1_dec * 100, 2), "%\n")
```

```
## F1 score for Naive Bayes model for raw_data1: 78.65 %
```

```r
# Creating the confusion matrix
the_matrix_is_real2 <- table(predictions2$class, test_data2$Diabetes_012)
the_matrix_is_real2
```

```
##
##           0     1
##   0 45091  5648
##   1  8334  4346
```

```r
d2 <- CrossTable(the_matrix_is_real2,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     45091 |      5648 |     50739 |
##              |     0.711 |     0.089 |           |
```

```
## -------------|-----------|-----------|-----------|
##            1 |      8334 |      4346 |     12680 |
##              |     0.131 |     0.069 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```
d2
```

```
## $t
##
##         0     1
##   0 45091  5648
##   1  8334  4346
##
## $prop.row
##
##             0         1
##   0 0.8886852 0.1113148
##   1 0.6572555 0.3427445
##
## $prop.col
##
##             0         1
##   0 0.8440056 0.5651391
##   1 0.1559944 0.4348609
##
## $prop.tbl
##
##              0          1
##   0 0.71100143 0.08905848
##   1 0.13141172 0.06852836
```

```
true_negative2 <- the_matrix_is_real2[1, 1]  # Correctly predicted No
true_positive2 <- the_matrix_is_real2[2, 2]  # Correctly predicted Yes
false_positive2 <- the_matrix_is_real2[2, 1] # Incorrectly predicted Yes
false_negative2 <- the_matrix_is_real2[1, 2] # Incorrectly predicted No

accuracy2 <- (true_positive2 + true_negative2) / (true_negative2+true_positive2+false_posit

precision_NB2<-round(d2$t[1]/ sum(d2$t[3],d2$t[1]),3)

recall2 <- true_negative2 / (true_negative2+true_positive2+false_positive2+false_negative2)
f1_dec2 <- 2 * (precision_NB2 * recall2) / (precision_NB2 + recall2)
cat("Accuracy score for Naive Bayes model for raw_data2:", round(accuracy2 * 100, 2), "%\n"
```

```
## Accuracy score for Naive Bayes model for raw_data2: 77.95 %
```

```r
cat("Precision score for Naive Bayes model for raw_data2:", round(precision_NB2 * 100, 2),
```

```
## Precision score for Naive Bayes model for raw_data2: 88.9 %
```

```r
cat("Recall score for Naive Bayes model for raw_data2:", round(recall2 * 100, 2), "%\n")
```

```
## Recall score for Naive Bayes model for raw_data2: 71.1 %
```

```r
cat("F1 score for Naive Bayes model for raw_data2:", round(f1_dec2 * 100, 2), "%\n")
```

```
## F1 score for Naive Bayes model for raw_data2: 79.01 %
```

We can see based on the confusion matrix that the missing values are playing an effect on the Naive Bayes model in Predicting diabetes diagnosis. We can see that with raw_data1 which was not modified to include missing values predicted True Positive values better than the model that had the data that had missing values with values of 5234 and 4346 . However it we can see based on the Accuracy, Precision, Recall and F1 score that the naive bayes model performed better when missing values were involved indicating the effect that the missing values do have an effect on how Naive Bayes model performs.

Logistic Regression

```r
logistic_model <- glm(Diabetes_012 ~ ., data = train_data, family = binomial(link = "logit"
predictions_prob <- suppressWarnings(predict(logistic_model, newdata = test_data, type = "r
```

```r
logistic_model2 <- glm(Diabetes_012 ~ ., data = train_data2, family = binomial(link = "logi
predictions_prob2 <- suppressWarnings(predict(logistic_model2, newdata = test_data2, type =
```

Evaluation of Logistic Regression Model

```r
# Creating the confusion matrix
# Convert probabilities to binary predictions using a threshold
threshold <- 0.5
binary_predictions <- ifelse(predictions_prob > threshold, 1, 0)


# Now create the confusion matrix
logistic_matrix <- table(Predicted = binary_predictions, Actual = test_data$Diabetes_012)

# Print the confusion matrix
logistic_matrix
```

```
##          Actual
## Predicted     0     1
##         0 52034  8155
##         1  1391  1839
```

```r
d3 <- CrossTable(logistic_matrix,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     52034 |      8155 |     60189 |
##              |     0.820 |     0.129 |           |
## -------------|-----------|-----------|-----------|
##            1 |      1391 |      1839 |      3230 |
##              |     0.022 |     0.029 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```r
d3
```

```
## $t
##          Actual
## Predicted     0     1
##         0 52034  8155
##         1  1391  1839
##
## $prop.row
##          Actual
## Predicted         0         1
```

```
##          0 0.8645101 0.1354899
##          1 0.4306502 0.5693498
##
## $prop.col
##           Actual
## Predicted         0         1
##          0 0.9739635 0.8159896
##          1 0.0260365 0.1840104
##
## $prop.tbl
##           Actual
## Predicted          0          1
##          0 0.82047967 0.12858922
##          1 0.02193349 0.02899762
```

```
true_negative3 <- logistic_matrix[1, 1]  # Correctly predicted No
true_positive3 <- logistic_matrix[2, 2]  # Correctly predicted Yes
false_positive3 <- logistic_matrix[2, 1] # Incorrectly predicted Yes
false_negative3 <- logistic_matrix[1, 2] # Incorrectly predicted No

accuracy3 <- (true_positive3 + true_negative3) / (true_negative3+true_positive3+false_posit

precision_NB3<-round(d3$t[1]/ sum(d3$t[3],d3$t[1]),3)

recall3 <- true_negative3 / (true_negative3+true_positive3+false_positive3+false_negative3)
f1_dec3 <- 2 * (precision_NB3 * recall3) / (precision_NB3 + recall3)
cat("Accuracy score for Logistic Regression model for raw_data1:", round(accuracy3 * 100, 2
```

```
## Accuracy score for Logistic Regression model for raw_data1: 84.95 %
```

```
cat("Precision score for Logistic Regression model for raw_data1:", round(precision_NB3 * 1
```

```
## Precision score for Logistic Regression model for raw_data1: 86.5 %
```

```
cat("Recall score for Logistic Regression model for raw_data1:", round(recall3 * 100, 2), "
```

```
## Recall score for Logistic Regression model for raw_data1: 82.05 %
```

```
cat("F1 score for Logistic Regression model for raw_data1:", round(f1_dec3 * 100, 2), "%\n"
```

```
## F1 score for Logistic Regression model for raw_data1: 84.22 %
```

```r
# Creating the confusion matrix
# Convert probabilities to binary predictions using a threshold of 0.5
binary_predictions2 <- ifelse(predictions_prob2 > 0.5, 1, 0)


# Now create the confusion matrix
logistic_matrix2 <- table(Predicted = binary_predictions2, Actual = test_data2$Diabetes_012

# Print the confusion matrix
logistic_matrix2
```

```
##          Actual
## Predicted     0     1
##         0 52338  8814
##         1  1087  1180
```

```r
d4 <- CrossTable(logistic_matrix2,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     52338 |      8814 |     61152 |
##              |     0.825 |     0.139 |           |
## -------------|-----------|-----------|-----------|
##            1 |      1087 |      1180 |      2267 |
##              |     0.017 |     0.019 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```r
d4
```

```
## $t
##          Actual
## Predicted     0     1
##         0 52338  8814
##         1  1087  1180
##
## $prop.row
##          Actual
## Predicted         0         1
##         0 0.8558673 0.1441327
##         1 0.4794883 0.5205117
##
## $prop.col
##          Actual
## Predicted          0          1
##         0 0.97965372 0.88192916
##         1 0.02034628 0.11807084
##
## $prop.tbl
##          Actual
## Predicted          0          1
##         0 0.82527318 0.13898043
##         1 0.01713997 0.01860641
```

```r
true_negative4 <- logistic_matrix2[1, 1]  # Correctly predicted No
true_positive4 <- logistic_matrix2[2, 2]  # Correctly predicted Yes
false_positive4 <- logistic_matrix2[2, 1] # Incorrectly predicted Yes
false_negative4 <- logistic_matrix2[1, 2] # Incorrectly predicted No

accuracy4 <- (true_positive4 + true_negative4) / (true_negative4+true_positive4+false_posit

precision_NB4<-round(d4$t[1]/ sum(d4$t[3],d4$t[1]),3)

recall4 <- true_negative4 / (true_negative4+true_positive4+false_positive4+false_negative4)
f1_dec4 <- 2 * (precision_NB4 * recall4) / (precision_NB4 + recall4)
cat("Accuracy score for Logistic Regression model for raw_data2:", round(accuracy4 * 100, 2
```

```
## Accuracy score for Logistic Regression model for raw_data2: 84.39 %
```

```r
cat("Precision score for Logistic Regression model for raw_data2:", round(precision_NB4 * 1
```

```
## Precision score for Logistic Regression model for raw_data2: 85.6 %
```

```r
cat("Recall score for Logistic Regression model for raw_data2:", round(recall4 * 100, 2), "
```

```
## Recall score for Logistic Regression model for raw_data2: 82.53 %
```

```r
cat("F1 score for Logistic Regression model for raw_data2:", round(f1_dec4 * 100, 2), "%\n"
```

```
## F1 score for Logistic Regression model for raw_data2: 84.04 %
```

Similar to the Naive bayes mode, the Logistic regression predicts more true postive values with raw_data1 as can be seen in the confusion matrix (1839, compared to 1180). Additionally the Recall is also higher in the Logistic Regression model for raw_data2. However the Model for raw_data2 is not the better performing model as the F1 score for the Logistic regression is higher with the model being performed on raw_data1 as it has a higher accuracy, precision and f1 score. Thus we can conclude that missing values affect Logistic regression models negatively

Decision Tree

We will be using a cost matrix for our decision tree. This is because a false negative diagnosis of diabetes could be costly the patient..

```r
matrix_dimensions <- list(c(0, 1), c(0, 1))
names(matrix_dimensions) <- c("predicted", "actual")

error_cost <- matrix(c(0, 2, 4, 0), nrow = 2,
                    dimnames = matrix_dimensions)

tree_model <- C5.0(Diabetes_012 ~ ., data = train_data, costs = error_cost)
#summary(tree_model)
# Make predictions on the validation set
tree_predictions <- predict(tree_model, newdata = test_data)
```

```r
tree_model2 <- C5.0(Diabetes_012 ~ ., data = train_data2, costs = error_cost)
#summary(tree_model)
# Make predictions on the validation set
tree_predictions2 <- predict(tree_model2, newdata = test_data2)
```

Evaluation of Model

```r
# Create a confusion matrix to evaluate the model
confusion_matrix_tree <- table(Predicted = tree_predictions, Actual = test_data$Diabetes_01

# Print the confusion matrix
confusion_matrix_tree
```

```
##          Actual
## Predicted     0     1
##         0 47185  5002
##         1  6240  4992
```

```r
d5 <- CrossTable(confusion_matrix_tree,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     47185 |      5002 |     52187 |
##              |     0.744 |     0.079 |           |
## -------------|-----------|-----------|-----------|
##            1 |      6240 |      4992 |     11232 |
##              |     0.098 |     0.079 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```r
d5
```

```
## $t
##          Actual
## Predicted     0     1
##         0 47185  5002
##         1  6240  4992
##
## $prop.row
##          Actual
## Predicted          0          1
```

```
##         0 0.90415238 0.09584762
##         1 0.55555556 0.44444444
##
## $prop.col
##          Actual
## Predicted         0         1
##         0 0.8832007 0.5005003
##         1 0.1167993 0.4994997
##
## $prop.tbl
##          Actual
## Predicted          0          1
##         0 0.74401993 0.07887226
##         1 0.09839323 0.07871458
```

```
true_negative5 <- confusion_matrix_tree[1, 1]  # Correctly predicted No
true_positive5 <- confusion_matrix_tree[2, 2]  # Correctly predicted Yes
false_positive5 <- confusion_matrix_tree[1, 2] # Incorrectly predicted Yes
false_negative5 <- confusion_matrix_tree[2, 1] # Incorrectly predicted No

accuracy5 <- (true_positive5 + true_negative5) / (true_negative5+true_positive5+false_posit

precision_NB5<-round(d5$t[1]/ sum(d5$t[3],d5$t[1]),3)

recall5 <- true_negative5 / (true_negative5+true_positive5+false_positive5+false_negative5)
f1_dec5 <- 2 * (precision_NB5 * recall5) / (precision_NB5 + recall5)
cat("Accuracy score for Decision tree model for raw_data1:", round(accuracy5 * 100, 2), "%\
```

```
## Accuracy score for Decision tree model for raw_data1: 82.27 %
```

```
cat("Precision score for Decision tree model for raw_data1:", round(precision_NB5 * 100, 2)
```

```
## Precision score for Decision tree model for raw_data1: 90.4 %
```

```
cat("Recall score for Decision tree model for raw_data1:", round(recall5 * 100, 2), "%\n")
```

```
## Recall score for Decision tree model for raw_data1: 74.4 %
```

```
cat("F1 score for Decision tree model with raw_data1:", round(f1_dec5 * 100, 2), "%\n")
```

```
## F1 score for Decision tree model with raw_data1: 81.62 %
```

```r
# Create a confusion matrix to evaluate the model
confusion_matrix_tree2 <- table(Predicted = tree_predictions2, Actual = test_data2$Diabetes_

# Print the confusion matrix
confusion_matrix_tree2
```

```
##          Actual
## Predicted     0     1
##         0 48672  6567
##         1  4753  3427
```

```r
d6 <- CrossTable(confusion_matrix_tree2,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     48672 |      6567 |     55239 |
##              |     0.767 |     0.104 |           |
## -------------|-----------|-----------|-----------|
##            1 |      4753 |      3427 |      8180 |
##              |     0.075 |     0.054 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```r
d6
```

```
## $t
##          Actual
```

```
## Predicted     0     1
##         0 48672  6567
##         1  4753  3427
##
## $prop.row
##          Actual
## Predicted         0         1
##         0 0.8811166 0.1188834
##         1 0.5810513 0.4189487
##
## $prop.col
##          Actual
## Predicted          0          1
##         0 0.91103416 0.65709426
##         1 0.08896584 0.34290574
##
## $prop.tbl
##          Actual
## Predicted          0          1
##         0 0.76746716 0.10354941
##         1 0.07494599 0.05403743
```

```
true_negative6 <- confusion_matrix_tree2[1, 1]  # Correctly predicted No
true_positive6 <- confusion_matrix_tree2[2, 2]  # Correctly predicted Yes
false_positive6 <- confusion_matrix_tree2[1, 2] # Incorrectly predicted Yes
false_negative6 <- confusion_matrix_tree2[2, 1] # Incorrectly predicted No

accuracy6 <- (true_positive6 + true_negative6) / (true_negative6+true_positive6+false_positi

precision_NB6<-round(d6$t[1]/ sum(d6$t[3],d6$t[1]),3)

recall6 <- true_negative6 / (true_negative6+true_positive6+false_positive6+false_negative6)
f1_dec6 <- 2 * (precision_NB6 * recall6) / (precision_NB6 + recall6)
cat("Accuracy score for Decision tree model for raw_data2:", round(accuracy6 * 100, 2), "%\
```

```
## Accuracy score for Decision tree model for raw_data2: 82.15 %
```

```
cat("Precision score for Decision tree model for raw_data2:", round(precision_NB6 * 100, 2)
```

```
## Precision score for Decision tree model for raw_data2: 88.1 %
```

```
cat("Recall score for Decision tree model for raw_data2:", round(recall6 * 100, 2), "%\n")
```

```
## Recall score for Decision tree model for raw_data2: 76.75 %
```

```
cat("F1 score for Decision tree model with raw_data2:", round(f1_dec6 * 100, 2), "%\n")
```

```
## F1 score for Decision tree model with raw_data2: 82.03 %
```

Similar to the previous 2 models, The model for raw_data1(no missing values) predicted more true positive values 4992 and 3427 respectively. However unlike the other two models, it was hard to determine what the better performing model was. The reason is because while the F1 score and Recall score were abetter in the model performed on raw_data2(missing values), the accuracy and precision were better with the mode performed on raw_data1. However this project is trying to determine the diagnosis of diabetes. As a result the F1 score is more important to us. Therefore we can conclude that the model performed on raw_data2 was better out of the two.

Bagging

We can see above that suprisingly our decision tree model is not performing as well as we expected. One way to make it perform better is to perform a Bootstrap aggregation aka bagging.

```
# Diabetes_012 is a categorical variable and the target column. no conversion is needed fo

set.seed(123)
ctrl <- trainControl(method = "cv", number = 2)
mybag <- bagging(Diabetes_012 ~ ., data = train_data, nbagg = 10, costs = error_cost, trCon
bag_pred <- predict(mybag, newdata = test_data)
table(bag_pred, test_data$Diabetes_012)
```

```
##
## bag_pred     0     1
##        0 49130  7100
##        1  4295  2894
```

```
set.seed(123)
mybag2 <- bagging(Diabetes_012 ~ ., data = train_data2, nbagg = 10, costs = error_cost, trC
bag_pred2 <- predict(mybag2, newdata = test_data2)
table(bag_pred2, test_data2$Diabetes_012)
```

```
##
## bag_pred2     0     1
##         0 49219  7719
##         1  4206  2275
```

Evaluation of bagged ensemble model

```r
bag_matrix <- table(bag_pred, test_data$Diabetes_012)

d12 <- CrossTable(bag_matrix,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     49130 |      7100 |     56230 |
##              |     0.775 |     0.112 |           |
## -------------|-----------|-----------|-----------|
##            1 |      4295 |      2894 |      7189 |
##              |     0.068 |     0.046 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```r
d12
```

```
## $t
##
## bag_pred     0     1
##        0 49130  7100
##        1  4295  2894
##
## $prop.row
##
## bag_pred         0         1
##        0 0.8737329 0.1262671
##        1 0.5974405 0.4025595
##
```

```
## $prop.col
##
## bag_pred          0          1
##        0 0.91960693 0.71042626
##        1 0.08039307 0.28957374
##
## $prop.tbl
##
## bag_pred          0          1
##        0 0.77468897 0.11195383
##        1 0.06772418 0.04563301
```

```
true_negative12 <- bag_matrix[1, 1]  # Correctly predicted No
true_positive12 <- bag_matrix[2, 2]  # Correctly predicted Yes
false_positive12 <- bag_matrix[1, 2] # Incorrectly predicted Yes
false_negative12 <- bag_matrix[2, 1] # Incorrectly predicted No

accuracy12 <- (true_positive12 + true_negative12) / (true_negative12+true_positive12+false_p

precision_NB12<-round(d12$t[1]/ sum(d12$t[3],d12$t[1]),3)

recall12 <- true_negative12 / (true_negative12+true_positive12+false_positive12+false_negati
f1_dec12 <- 2 * (precision_NB12 * recall12) / (precision_NB12 + recall12)
cat("Accuracy score for Bagging model for raw_data1:", round(accuracy12 * 100, 2), "%\n")
```

```
## Accuracy score for Bagging model for raw_data1: 82.03 %
```

```
cat("Precision score for Bagging model for raw_data1:", round(precision_NB12 * 100, 2), "%\
```

```
## Precision score for Bagging model for raw_data1: 87.4 %
```

```
cat("Recall score for Bagging model for raw_data1:", round(recall12 * 100, 2), "%\n")
```

```
## Recall score for Bagging model for raw_data1: 77.47 %
```

```
cat("F1 score for Bagging model for raw_data1:", round(f1_dec12 * 100, 2), "%\n")
```

```
## F1 score for Bagging model for raw_data1: 82.14 %
```

```
bag_matrix2 <- table(bag_pred2, test_data2$Diabetes_012)

d13 <- CrossTable(bag_matrix2,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     49219 |      7719 |     56938 |
##              |     0.776 |     0.122 |           |
## -------------|-----------|-----------|-----------|
##            1 |      4206 |      2275 |      6481 |
##              |     0.066 |     0.036 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```
d13
```

```
## $t
##
## bag_pred2     0     1
##         0 49219  7719
##         1  4206  2275
##
## $prop.row
##
## bag_pred2         0         1
##         0 0.8644315 0.1355685
##         1 0.6489739 0.3510261
##
## $prop.col
##
## bag_pred2          0          1
##         0 0.92127281 0.77236342
##         1 0.07872719 0.22763658
##
## $prop.tbl
```

```
##
## bag_pred2          0          1
##         0 0.77609234 0.12171431
##         1 0.06632082 0.03587253
```

```r
true_negative13 <- bag_matrix2[1, 1]  # Correctly predicted No
true_positive13 <- bag_matrix2[2, 2]  # Correctly predicted Yes
false_positive13 <- bag_matrix2[1, 2] # Incorrectly predicted Yes
false_negative13 <- bag_matrix2[2, 1] # Incorrectly predicted No

accuracy13 <- (true_positive13 + true_negative13) / (true_negative13+true_positive13+false_p

precision_NB13<-round(d13$t[1]/ sum(d13$t[3],d13$t[1]),3)

recall13 <- true_negative13 / (true_negative13+true_positive13+false_positive13+false_negat
f1_dec13 <- 2 * (precision_NB13 * recall13) / (precision_NB13 + recall13)
cat("Accuracy score for Bagging model for raw_data2:", round(accuracy13 * 100, 2), "%\n")
```

```
## Accuracy score for Bagging model for raw_data2: 81.2 %
```

```r
cat("Precision score for Bagging model for raw_data2:", round(precision_NB13 * 100, 2), "%\
```

```
## Precision score for Bagging model for raw_data2: 86.4 %
```

```r
cat("Recall score for Bagging model for raw_data2:", round(recall13 * 100, 2), "%\n")
```

```
## Recall score for Bagging model for raw_data2: 77.61 %
```

```r
cat("F1 score for Bagging model for raw_data2:", round(f1_dec13 * 100, 2), "%\n")
```

```
## F1 score for Bagging model for raw_data2: 81.77 %
```

The bagging model was similar to that of the previous models in the context that it produced more true positive values. We can conclude based on the above results that the model performed on raw_data1 was the better peforming model as it had a higher accuracy,precision and f1. Howeveer the recall for the model on raw_data2 was higher. This isnt all surprising as the baggin ensemble model produced the average output of 10 different decision trees. Additionally we used a K-fold validation to determine the evaluation here therefore probably one of the reasons as to why it doesn't match the evaluation we produced for the single decision tree.

Evaluating the models.

Inorder to find the best performing model, we can use the f1 score of each model to see. We can use a function to print out the F1-scores and compare the models based on that.

```r
f1 <- c(
  NaiveBayes = f1_dec,
  DecisionTree = f1_dec5,
  LogisticRegression = f1_dec3,
  bagging = f1_dec12
)

f2 <- c(
  NaiveBayes = f1_dec2,
  DecisionTree = f1_dec6,
  LogisticRegression = f1_dec4,
  bagging = f1_dec13
)

the_end_is_near <- names(f1)[which.max(f1)]
the_end_is_near2 <- names(f1)[which.max(f2)]
# Print the F1-Scores and compare the models
cat("F1-Scores:\n")
```

```
## F1-Scores:
```

```r
cat(paste(names(f1), ": ", round(f1, 4), "\n"))
```

```
## NaiveBayes :  0.7865
##  DecisionTree :  0.8162
##  LogisticRegression :  0.8422
##  bagging :  0.8214
```

```r
cat("F2-Scores:\n")
```

```
## F2-Scores:
```

```r
cat(paste(names(f2), ": ", round(f2, 4), "\n"))
```

```
## NaiveBayes :  0.7901
##  DecisionTree :  0.8203
##  LogisticRegression :  0.8404
##  bagging :  0.8177
```

```r
cat("\nBest performing Model for data without missing values is: ", the_end_is_near, " (F1-S
```

```
##
## Best performing Model for data without missing values is:  LogisticRegression  (F1-Score
```

```
cat("\nBest performing Model for data with missing values is: ", the_end_is_near2, " (F1-Sco
```

```
##
## Best performing Model for data with missing values is:  LogisticRegression  (F1-Score:  (
```

based on the above code chunk we can conclude that the best performing model isLogisticRegression with an F1-Score of 0.8422 for data without missing values. For data with missing values, the best performing model is LogisticRegression with an F1 score of 0.8404. Based on the f1 level we saw before, we can conclude with the fact that LogisticRegression and LogisticRegression are the best model to use to predict diabetes diagnosis.

Ensemble

An ensemble model in machine learning is a technique that combines the predictions from multiple machine learning algorithms to make more accurate predictions than any individual model. It's a powerful method that can significantly improve model performance, particularly in complex tasks like classification or regression. The key idea is that by combining multiple models, the ensemble model can capitalize on the strengths and minimize the weaknesses of the individual models. Here are some key aspects of ensemble models:

Types of Ensemble Methods:

- Bagging: Short for Bootstrap Aggregating. It involves creating multiple models of the same type, each trained on a different subset of the same data. Each model votes, and the most common prediction is chosen. A well-known example is the Random Forest algorithm.
- Boosting: This method involves sequentially training models, where each new model attempts to correct the errors of the previous one. The models are then weighted and combined to produce a final prediction. Examples include AdaBoost and Gradient Boosting.
- Stacking: This involves training multiple different models and then using another model, often called a meta-model, to combine their predictions. The first-level models are trained on the full dataset, then the meta-model is trained on the outputs of these models.

Advantages:

- Improved Accuracy: By combining multiple models, ensembles often achieve higher accuracy than single models.
- Reduced Overfitting: Ensemble methods can reduce the risk of overfitting, especially if the individual models are overfitting in different ways [1].
- Handling Diverse Data: They can better handle different types of data and relationships because they integrate different kinds of models [1].

Applications:

- Ensemble models are used in various applications like risk assessment, fraud detection, disease diagnosis, and more, where accuracy is crucial, and the cost of wrong predictions is high. This method will be very useful in helping us predict diabetes diagnosis.

Additionally the ensemble model can help us predict if an individual is diagnosed with diabetees

The below code chunk will create an ensemble function that will combine the predictions of all the previous models.

```
ensemblefunction <- function(data) {
    # Assuming logistic_model, decision_tree_model, nb_model, random_model are already tra
    nb_predict <- suppressWarnings(predict(nb_model, data)$class)
    #print("Original NB Predictions:") # Debugging line
    #print(nb_predict)
    # Convert factor predictions to numeric
    nb_predict <- as.numeric(as.character(nb_predict))

    logistic_reg_prob <- suppressWarnings(predict(logistic_model, newdata = data, type = "r
    logistic_reg <- ifelse(logistic_reg_prob > 0.5, 1, 0)

    tree_predictions <- predict(tree_model, newdata = data)
    tree_predictions <- as.numeric(as.character(tree_predictions))

    bag_pred <- predict(mybag, newdata = data)
    bag_pred <- as.numeric(as.character(bag_pred))

    x <- rbind(nb_predict, logistic_reg, tree_predictions, bag_pred)

    # Weights based on model performance
    weight_nb = 0.1  # Weight for Naive Bayes
    weight_lr = 4   # Weight for Logistic Regression
    weight_dt = 2   # Weight for Decision Tree
    weight_bag = 2   # Weight for bag

    # Weighted sum of predictions
    weighted_sum = (nb_predict * weight_nb) +
                   (logistic_reg * weight_lr) +
                   (tree_predictions * weight_dt) +
                   (bag_pred * weight_bag)
    # Final prediction based on weighted sum
    final_prediction = ifelse(weighted_sum > (weight_nb + weight_lr + weight_dt + weight_ba

    # Return a named vector for clarity
    named_result <- setNames(final_prediction, "Predicted_Class")

    return(named_result)
}
```

```r
a <- raw_data[61, -1]
pred <- ensemblefunction(a)
pred
```

```
## Predicted_Class
##               1
```

```r
ensemblefunction2 <- function(data) {
    # Assuming logistic_model, decision_tree_model, nb_model, random_model are already tra
    nb_predict2 <- suppressWarnings(predict(nb_model2, data)$class)
    #print("Original NB Predictions:") # Debugging line
    #print(nb_predict)
    # Convert factor predictions to numeric
    nb_predict2 <- as.numeric(as.character(nb_predict2))

    logistic_reg_prob2 <- suppressWarnings(predict(logistic_model2, newdata = data, type =
    logistic_reg2 <- ifelse(logistic_reg_prob2 > 0.5, 1, 0)

    tree_predictions2 <- predict(tree_model2, newdata = data)
    tree_predictions2 <- as.numeric(as.character(tree_predictions2))

    bag_pred2 <- predict(mybag2, newdata = data)
    bag_pred2 <- as.numeric(as.character(bag_pred2))

    x <- rbind(nb_predict2, logistic_reg2, tree_predictions2, bag_pred2)

    # Weights based on model performance
    weight_nb2 = 0.1  # Weight for Naive Bayes
    weight_lr2 = 4    # Weight for Logistic Regression
    weight_dt2 = 2  # Weight for Decision Tree
    weight_bag2 = 2    # Weight for bag

    # Weighted sum of predictions
    weighted_sum = (nb_predict2 * weight_nb2) +
                   (logistic_reg2 * weight_lr2) +
                   (tree_predictions2 * weight_dt2) +
                   (bag_pred2 * weight_bag2)
    # Final prediction based on weighted sum
    final_prediction = ifelse(weighted_sum > (weight_nb2 + weight_lr2 + weight_dt2 + weight

    # Return a named vector for clarity
    named_result <- setNames(final_prediction, "Predicted_Class")

    return(named_result)
}
```

```
a <- raw_data[61, -1]
pred2 <- ensemblefunction2(a)
pred2
```

```
## Predicted_Class
##               1
```

We can use the ensemble model to predict a diagnosis. As we can see here with the two models its intresting to note that they have predicted two same classes between them for the same Data point in the same data set. Thus indicating that missing values had no affect on the models ability to predict.

Evaluating the Ensemble modell

```
test_matrix <- ensemblefunction(test_data)
# Create a confusion matrix
confusion_matrix2 <- table(Predicted = test_matrix, Actual = test_data$Diabetes_012)
print(confusion_matrix2)
```

```
##          Actual
## Predicted     0     1
##         0 50929  7331
##         1  2496  2663
```

```
d11 <- CrossTable(confusion_matrix2,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     50929 |      7331 |     58260 |
```

```
##               |     0.803 |     0.116 |           |
## --------------|-----------|-----------|-----------|
##             1 |      2496 |      2663 |      5159 |
##               |     0.039 |     0.042 |           |
## --------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## --------------|-----------|-----------|-----------|
##
##
```

```
d11
```

```
## $t
##          Actual
## Predicted     0     1
##         0 50929  7331
##         1  2496  2663
##
## $prop.row
##          Actual
## Predicted         0         1
##         0 0.8741675 0.1258325
##         1 0.4838147 0.5161853
##
## $prop.col
##          Actual
## Predicted         0         1
##         0 0.9532803 0.7335401
##         1 0.0467197 0.2664599
##
## $prop.tbl
##          Actual
## Predicted          0          1
##         0 0.80305587 0.11559627
##         1 0.03935729 0.04199057
```

```
true_negative11 <- confusion_matrix2[1, 1]  # Correctly predicted No
true_positive11 <- confusion_matrix2[2, 2]  # Correctly predicted Yes
false_positive11 <- confusion_matrix2[2, 1] # Incorrectly predicted Yes
false_negative11 <- confusion_matrix2[1, 2] # Incorrectly predicted No

accuracy11 <- (true_positive11 + true_negative11) / (true_negative11+true_positive11+false_

precision_NB11<-round(d11$t[1]/ sum(d11$t[3],d11$t[1]),3)

recall11 <- true_negative11 / (true_negative11+true_positive11+false_positive11+false_negat
f1_dec11 <- 2 * (precision_NB11 * recall11) / (precision_NB11 + recall11)
cat("Accuracy score for ensemble for raw_data1:", round(accuracy11 * 100, 2), "%\n")
```

```
## Accuracy score for ensemble for raw_data1: 84.5 %
```

```r
cat("Precision score for ensemble for raw_data1:", round(precision_NB11 * 100, 2), "%\n")
```

```
## Precision score for ensemble for raw_data1: 87.4 %
```

```r
cat("Recall score for ensemble for raw_data1:", round(recall11 * 100, 2), "%\n")
```

```
## Recall score for ensemble for raw_data1: 80.31 %
```

```r
cat("F1 score for ensemble for raw_data1:", round(f1_dec11 * 100, 2), "%\n")
```

```
## F1 score for ensemble for raw_data1: 83.7 %
```

```r
f3 <- c(
  NaiveBayes = f1_dec,
  DecisionTree = f1_dec5,
  LogisticRegression = f1_dec3,
  bagging = f1_dec12,
  ensemble = f1_dec11
)

the_end_is_near <- names(f3)[which.max(f3)]
#the_end_is_near2 <- names(f1)[which.max(f2)]
# Print the F1-Scores and compare the models
cat("F1-Scores:\n")
```

```
## F1-Scores:
```

```r
cat(paste(names(f3), ": ", round(f3, 3), "\n"))
```

```
## NaiveBayes :  0.786
##  DecisionTree :  0.816
##  LogisticRegression :  0.842
##  bagging :  0.821
##  ensemble :  0.837
```

```r
#cat("F2-Scores:\n")
#cat(paste(names(f2), ": ", round(f2, 4), "\n"))

cat("\nBest performing Model for data without missing values is: ", the_end_is_near, " (F1-S
```

```
##
## Best performing Model for data without missing values is:  LogisticRegression  (F1-Score
```

```r
#cat("\nBest performing Model for data without missing values is: ", the_end_is_near2, "
```

```r
test_matrix2 <- ensemblefunction2(test_data2)
# Create a confusion matrix
confusion_matrix3 <- table(Predicted = test_matrix2, Actual = test_data2$Diabetes_012)
print(confusion_matrix3)
```

```
##          Actual
## Predicted     0     1
##         0 51433  8174
##         1  1992  1820
```

```r
d12 <- CrossTable(confusion_matrix3,
    prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
    dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  63419
##
##
##              | actual
##    predicted |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |     51433 |      8174 |     59607 |
##              |     0.811 |     0.129 |           |
## -------------|-----------|-----------|-----------|
##            1 |      1992 |      1820 |      3812 |
##              |     0.031 |     0.029 |           |
## -------------|-----------|-----------|-----------|
## Column Total |     53425 |      9994 |     63419 |
## -------------|-----------|-----------|-----------|
##
##
```

```r
d12
```

```
## $t
```

```
##           Actual
## Predicted     0     1
##         0 51433  8174
##         1  1992  1820
##
## $prop.row
##           Actual
## Predicted         0         1
##         0 0.8628685 0.1371315
##         1 0.5225603 0.4774397
##
## $prop.col
##           Actual
## Predicted          0          1
##         0 0.96271409 0.81789073
##         1 0.03728591 0.18210927
##
## $prop.tbl
##           Actual
## Predicted          0          1
##         0 0.81100301 0.12888882
##         1 0.03141015 0.02869802
```

```r
true_negative12 <- confusion_matrix3[1, 1]  # Correctly predicted No
true_positive12 <- confusion_matrix3[2, 2]  # Correctly predicted Yes
false_positive12 <- confusion_matrix3[2, 1] # Incorrectly predicted Yes
false_negative12 <- confusion_matrix3[1, 2] # Incorrectly predicted No

accuracy12 <- (true_positive12 + true_negative12) / (true_negative12+true_positive12+false_

precision_NB12<-round(d12$t[1]/ sum(d12$t[3],d12$t[1]),3)

recall12 <- true_negative12 / (true_negative12+true_positive12+false_positive12+false_negat
f1_dec12 <- 2 * (precision_NB12 * recall12) / (precision_NB12 + recall12)
cat("Accuracy score for ensemble for raw_data2:", round(accuracy12 * 100, 2), "%\n")
```

```
## Accuracy score for ensemble for raw_data2: 83.97 %
```

```r
cat("Precision score for ensemble for raw_data2:", round(precision_NB12 * 100, 2), "%\n")
```

```
## Precision score for ensemble for raw_data2: 86.3 %
```

```r
cat("Recall score for ensemble for raw_data2:", round(recall12 * 100, 2), "%\n")
```

```
## Recall score for ensemble for raw_data2: 81.1 %
```

```
cat("F1 score for ensemble for raw_data2:", round(f1_dec12 * 100, 2), "%\n")
```

```
## F1 score for ensemble for raw_data2: 83.62 %
```

```
f4 <- c(
  NaiveBayes = f1_dec2,
  DecisionTree = f1_dec5,
  LogisticRegression = f1_dec4,
  bagging = f1_dec13,
  ensemble = f1_dec12
)

the_end_is_near4 <- names(f4)[which.max(f4)]
#the_end_is_near2 <- names(f1)[which.max(f2)]
# Print the F1-Scores and compare the models
cat("F1-Scores:\n")
```

```
## F1-Scores:
```

```
cat(paste(names(f4), ": ", round(f4, 4), "\n"))
```

```
## NaiveBayes :  0.7901
##  DecisionTree :  0.8162
##  LogisticRegression :  0.8404
##  bagging :  0.8177
##  ensemble :  0.8362
```

```
#cat("F2-Scores:\n")
#cat(paste(names(f2), ": ", round(f2, 4), "\n"))

cat("\nBest performing Model for data without missing values is: ", the_end_is_near, " (F1-S
```

```
##
## Best performing Model for data without missing values is:  LogisticRegression  (F1-Score
```

```
#cat("\nBest performing Model for data without missing values is: ", the_end_is_near2, " (
```

Interestingly enough the missing values did not play a role in determining which model was the best performing model. In both cases, the logistic regression model was the best performing model. While usually the ensemble model is the best performing model, it is not entirely surprising that a model such as the Logistic model is the best performing model. The effectiveness of any machine learning model, including ensemble models, depends on various factors such as the nature of the dataset, the complexity of the problem, feature relationships, and the specific configuration of the models. Additionally logistic regression models are generally easier to interpret and understand compared to complex ensemble models. In some cases, the interpretability of a model can be more important than achieving the absolute highest accuracy [3].

References

1. Lantz, B. (n.d.). Machine Learning with R - Fourth Edition. O'Reilly Online Learning. https://learning.oreilly.com/library/view/machine-learning-with/9781801071321/Text/Chapter_13.xhtml#_idParaDest-312

2. Brownlee, J. (2021, April 26). Why use ensemble learning? MachineLearningMastery.com. https://machinelearningmastery.com/why-use-ensemble-learning/

3. When is using logistic regression better than ensembles? (n.d.). Quora. https://www.quora.com/When-is-using-logistic-regression-better-than-ensembles