Stroke Prediction

Mohammed Akib Iftakher (20215282)

2022-03-25

1. Introduction

This analysis is an assessment of a data collection of patients in order to assign risk variables to strokes and make suggestions to assist prevent this occurrence. Stroke is the second biggest cause of death globally, with around 5.5 million deaths each year. Stroke has a high death rate, but it also has a high morbidity rate, resulting in up to 50% of survivors being permanently impaired[1]. The following report focuses on identifying stroke risk factors.

1.1 Library Installation

1. **tidyverse** - aids in data performance and interaction.
2. **ROSE**- assists with binary categorization, in the presence of unusual classes.
3. **corrplot** - a visual exploration tool for correlation matrices.
4. **caret** - used for Data preparation, model construction, and model assessment.
5. **RandomForest** - framework for classification and regression algorithm.
6. **gridExtra** - adds functionality to the grid system.
7. **ggplot2** - a framework for making graphics declaratively.
8. **ggmosaic** - a framework for create visualizations of categorical data.
9. **corrgram** - creates graphical display of a correlation matrix.
10. **ggpubr** - makes it easier to create stunning ggplot2-based graphs

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ROSE)
```

```
## Warning: package 'ROSE' was built under R version 4.1.3
```

```
## Loaded ROSE 0.0-4
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(caret)
```

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

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.3
```

```
## randomForest 4.7-1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.1.3
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:randomForest':
##
##     combine

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(ggplot2)
library(ggmosaic)
```

```
## Warning: package 'ggmosaic' was built under R version 4.1.3
```

```
library(corrgram)
```

```
## Warning: package 'corrgram' was built under R version 4.1.3

##
## Attaching package: 'corrgram'

## The following object is masked from 'package:lattice':
##
##     panel.fill
```

```
library(wesanderson)
```

```
## Warning: package 'wesanderson' was built under R version 4.1.3
```

```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.1.3
```

```
data <- read.csv("D:\\personal\\R_programming\\Stroke\\archive\\healthcare-dataset-stroke-data.csv")
str(data)
```

1.2 Importing Dataset

```
## 'data.frame':    5110 obs. of  12 variables:
##  $ id               : int  9046 51676 31112 60182 1665 56669 53882 10434 27419 60491 ...
##  $ gender           : chr  "Male" "Female" "Male" "Female" ...
##  $ age              : num  67 61 80 49 79 81 74 69 59 78 ...
##  $ hypertension     : int  0 0 0 0 1 0 1 0 0 0 ...
##  $ heart_disease    : int  1 0 1 0 0 0 1 0 0 0 ...
##  $ ever_married     : chr  "Yes" "Yes" "Yes" "Yes" ...
##  $ work_type        : chr  "Private" "Self-employed" "Private" "Private" ...
##  $ Residence_type   : chr  "Urban" "Rural" "Rural" "Urban" ...
##  $ avg_glucose_level: num  229 202 106 171 174 ...
##  $ bmi              : chr  "36.6" "N/A" "32.5" "34.4" ...
##  $ smoking_status   : chr  "formerly smoked" "never smoked" "never smoked" "smokes" ...
##  $ stroke           : int  1 1 1 1 1 1 1 1 1 1 ...
```

1.3 Checking for Empty cell

```
data1 <- data # Duplicating the dataset
paste0("Empty data in the set: ", is.null(data1))
```

```
## [1] "Empty data in the set: FALSE"
```

So, it can be stated that there is no cell which is remain unoccupied in the corresponding columns.

1.4 Viewing all the unique data in the dataset

```
lapply(subset(data1, select = c(gender, ever_married, work_type, Residence_type, bmi, smoking_status)),
```

```
## $gender
## [1] "Male"   "Female" "Other"
##
## $ever_married
## [1] "Yes" "No"
##
## $work_type
## [1] "Private"       "Self-employed" "Govt_job"      "children"
## [5] "Never_worked"
##
## $Residence_type
## [1] "Urban" "Rural"
##
## $bmi
##   [1] "36.6" "N/A"  "32.5" "34.4" "24"   "29"   "27.4" "22.8" "24.2" "29.7"
##  [11] "36.8" "27.3" "28.2" "30.9" "37.5" "25.8" "37.8" "22.4" "48.9" "26.6"
##  [21] "27.2" "23.5" "28.3" "44.2" "25.4" "22.2" "30.5" "26.5" "33.7" "23.1"
##  [31] "32"   "29.9" "23.9" "28.5" "26.4" "20.2" "33.6" "38.6" "39.2" "27.7"
##  [41] "31.4" "36.5" "33.2" "32.8" "40.4" "25.3" "30.2" "47.5" "20.3" "30"
##  [51] "28.9" "28.1" "31.1" "21.7" "27"   "24.1" "45.9" "44.1" "22.9" "29.1"
##  [61] "32.3" "41.1" "25.6" "29.8" "26.3" "26.2" "29.4" "24.4" "28"   "28.8"
##  [71] "34.6" "19.4" "30.3" "41.5" "22.6" "56.6" "27.1" "31.3" "31"   "31.7"
##  [81] "35.8" "28.4" "20.1" "26.7" "38.7" "34.9" "25"   "23.8" "21.8" "27.5"
##  [91] "24.6" "32.9" "26.1" "31.9" "34.1" "36.9" "37.3" "45.7" "34.2" "23.6"
## [101] "22.3" "37.1" "45"   "25.5" "30.8" "37.4" "34.5" "27.9" "29.5" "46"
## [111] "42.5" "35.5" "26.9" "45.5" "31.5" "33"   "23.4" "30.7" "20.5" "21.5"
## [121] "40"   "28.6" "42.2" "29.6" "35.4" "16.9" "26.8" "39.3" "32.6" "35.9"
## [131] "21.2" "42.4" "40.5" "36.7" "29.3" "19.6" "18"   "17.6" "19.1" "50.1"
## [141] "17.7" "54.6" "35"   "22"   "39.4" "19.7" "22.5" "25.2" "41.8" "60.9"
## [151] "23.7" "24.5" "31.2" "16"   "31.6" "25.1" "24.8" "18.3" "20"   "19.5"
## [161] "36"   "35.3" "40.1" "43.1" "21.4" "34.3" "27.6" "16.5" "24.3" "25.7"
## [171] "21.9" "38.4" "25.9" "54.7" "18.6" "24.9" "48.2" "20.7" "39.5" "23.3"
## [181] "64.8" "35.1" "43.6" "21"   "47.3" "16.6" "21.6" "15.5" "35.6" "16.7"
## [191] "41.9" "16.4" "17.1" "29.2" "37.9" "44.6" "39.6" "40.3" "41.6" "39"
## [201] "23.2" "18.9" "36.1" "36.3" "46.5" "16.8" "46.6" "35.2" "20.9" "13.8"
## [211] "31.8" "15.3" "38.2" "45.2" "17"   "49.8" "27.8" "60.2" "23"   "22.1"
## [221] "26"   "44.3" "51"   "39.7" "34.7" "21.3" "41.2" "34.8" "19.2" "35.7"
## [231] "40.8" "24.7" "19"   "32.4" "34"   "28.7" "32.1" "51.5" "20.4" "30.6"
## [241] "71.9" "19.3" "40.9" "17.2" "16.1" "16.2" "40.6" "18.4" "21.1" "42.3"
## [251] "32.2" "50.2" "17.5" "18.7" "42.1" "47.8" "20.8" "30.1" "17.3" "36.4"
```

```
## [261] "12"   "36.2" "55.7" "14.4" "43"   "41.7" "33.8" "43.9" "22.7" "57.5"
## [271] "37"   "38.5" "16.3" "44"   "32.7" "54.2" "40.2" "33.3" "17.4" "41.3"
## [281] "52.3" "14.6" "17.8" "46.1" "33.1" "18.1" "43.8" "50.3" "38.9" "43.7"
## [291] "39.9" "15.9" "19.8" "12.3" "78"   "38.3" "41"   "42.6" "43.4" "15.1"
## [301] "20.6" "33.5" "43.2" "30.4" "38"   "33.4" "44.9" "44.7" "37.6" "39.8"
## [311] "53.4" "55.2" "42"   "37.2" "42.8" "18.8" "42.9" "14.3" "37.7" "48.4"
## [321] "50.6" "46.2" "49.5" "43.3" "33.9" "18.5" "44.5" "45.4" "55"   "54.8"
## [331] "19.9" "17.9" "15.6" "52.8" "15.2" "66.8" "55.1" "18.2" "48.5" "55.9"
## [341] "57.3" "10.3" "14.1" "15.7" "56"   "44.8" "13.4" "51.8" "38.1" "57.7"
## [351] "44.4" "38.8" "49.3" "39.1" "54"   "56.1" "97.6" "53.9" "13.7" "11.5"
## [361] "41.4" "14.2" "49.4" "15.4" "45.1" "49.2" "48.7" "53.8" "42.7" "48.8"
## [371] "52.7" "53.5" "50.5" "15.8" "45.3" "14.8" "51.9" "63.3" "40.7" "61.2"
## [381] "48"   "46.8" "48.3" "58.1" "50.4" "11.3" "12.8" "13.5" "14.5" "15"
## [391] "59.7" "47.4" "52.5" "13.2" "52.9" "61.6" "49.9" "54.3" "47.9" "13"
## [401] "13.9" "50.9" "57.2" "64.4" "92"   "50.8" "57.9" "45.8" "47.6" "14"
## [411] "46.4" "46.9" "47.1" "13.3" "48.1" "51.7" "46.3" "54.1" "14.9"
##
## $smoking_status
## [1] "formerly smoked" "never smoked"    "smokes"          "Unknown"
```

2. Cleaning the Data

To construct the models to make predictions, first the data needs to be cleaned according to the model requirement. By looking at the data it can be said that most of the data are already cleaned. There are few data which needs to be transformed to check the correlations.

```
table(factor(data1$gender))
```

2.1 Gender

```
##
## Female   Male  Other
##   2994   2115      1
```

```
cat("\nTable of Strokes corresponding to the gender")
```

```
##
## Table of Strokes corresponding to the gender
```

```
table(data1$gender, data1$stroke)
```

```
##
##             0    1
##   Female 2853  141
##   Male   2007  108
##   Other     1    0
```

```
cat("\nTransforming gender information into the numeric")
```

```
##
## Transforming gender information into the numeric
```

```
data1$gender <- as.character(data1$gender)
for (i in 1:length(data1$gender)) {
  if (data1$gender[i] == "Male") {
    data1$gender[i] <- 1
  }
  else if (data1$gender[i] == "Female") {
    data1$gender[i] <- 0
  }
  else {
    data1$gender[i] <- 0.5
  }
}
data1$gender <- as.numeric(data1$gender)
table(data1$gender)
```

```
##
##    0  0.5    1
## 2994    1 2115
```

```
cat("\nCorrelation between gender and stroke\n")
```

```
##
## Correlation between gender and stroke
```

```
with (data1,cor(gender,stroke))
```

```
## [1] 0.009072791
```

Here, it can be said that the data related to gender has been transformed into three category in respect of numerical values. It can also be said that some people may be identified as **other**. So, the data corresponds to other has not been merged with other two distinctive category and it is also not being omitted. Final observation from this part of the data is, gender doesn't have significant influence over the probability to have a stroke.

```
table(factor(data1$ever_married))
```

2.2 Marital Status

```
##
##   No  Yes
## 1757 3353
```

```
cat("\nTable of Strokes corresponding to the Marital status")
```

```
##
## Table of Strokes corresponding to the Marital status
```

```
table(data1$ever_married, data1$stroke)
```

```
##
##          0    1
##   No  1728   29
##   Yes 3133  220
```

```
cat("\nTransforming Ever_married information to numerical")
```

```
##
## Transforming Ever_married information to numerical
```

```
data1$ever_married <- as.numeric(factor(data1$ever_married)) - 1
table(data1$ever_married)
```

```
##
##    0    1
## 1757 3353
```

```
cat("\nCorrelation between Marital Status and stroke\n")
```

```
##
## Correlation between Marital Status and stroke
```

```
with (data1,cor(ever_married,stroke))
```

```
## [1] 0.1083397
```

First observation from this part of the data is, people who are married are expected to have higher probability of having stroke than the unmarried people. One theory can be stated that usually people who are married are usually belong to generation whose age are between 18 and respected life expectancy. So, there may be an explanation which later can be confirmed that people above 18 years are in the risk of having stroke. Another thing can be noticed that marital status and stroke are moderately correlated.

```
table(factor(data1$work_type))
```

2.3 Work Type or Status

```
##
##      children      Govt_job  Never_worked       Private Self-employed
##           687           657            22          2925           819
```

```
cat("\nTable of Strokes corresponding to the Work type")
```

```
##
## Table of Strokes corresponding to the Work type
```

```
table(data1$work_type, data1$stroke)
```

```
##
##                    0    1
##   children       685    2
##   Govt_job       624   33
##   Never_worked    22    0
##   Private       2776  149
##   Self-employed  754   65
```

```
data1$work_type <- as.character(data1$work_type)
for (i in 1:length(data1$id)) {
  if (data1$work_type[i] == "children" || data1$work_type[i] == "Never_worked" ) {
    data1$young[i] <- 1
  }
  else {
    data1$young[i] <- 0
  }
}

for (i in 1:length(data1$id)) {
  if (data1$work_type[i] == "Govt_job" || data1$work_type[i] == "Private Self" ) {
    data1$work_others[i] <- 1
  }
  else {
    data1$work_others[i] <- 0
  }
}

for (i in 1:length(data1$id)) {
  if (data1$work_type[i] == "Self-employed" ) {
    data1$work_self[i] <- 1
  }
  else {
    data1$work_self[i] <- 0
  }
}

cat("\nCorrelation between  Work Status (young) and stroke\n")
```

```
##
## Correlation between  Work Status (young) and stroke
```

```
with (data1,cor(young,stroke))
```

```
## [1] -0.08558266
```

```
cat("\nCorrelation between  Work Status (Govtl. or private) and stroke\n")
```

```
##
## Correlation between  Work Status (Govtl. or private) and stroke
```

```
with (data1,cor(work_others,stroke))
```

```
## [1] 0.002676705
```

```
cat("\nCorrelation between  Work Status (Self employed) and stroke\n")
```

```
##
## Correlation between  Work Status (Self employed) and stroke
```

```
with (data1,cor(work_self,stroke))
```

```
## [1] 0.06216826
```

The data related to work type are very diverse staring from government job to children. So, it will be a wise idea to sub categorized these information which will be easy in the long run to transform the data into numerical. So, here three sub category has been created in order to accommodate these diverse column. The first category represents children and people who never worked. The second category shows the people who works for government or private corporation. And the last one represents people who are self employed or entrepreneur. By looking at the correlation value, it is clear that self employed people are more prone to have strokes. So, the focus should on this category.

```
table(factor(data1$Residence_type))
```

2.4 Residence Type

```
##
## Rural Urban
##  2514  2596
```

```
cat("\nTable of Strokes corresponding to the Residence type")
```

```
##
## Table of Strokes corresponding to the Residence type
```

```
table(data1$Residence_type, data$stroke)
```

```
##
##            0    1
##   Rural 2400  114
##   Urban 2461  135
```

```
cat("\nTransforming Residence_type information to numerical")
```

```
##
## Transforming Residence_type information to numerical
```

```
data1$Residence_type <- as.numeric(factor(data1$Residence_type)) - 1

cat("\n\nCorrelation between  Residence type and stroke\n")
```

```
##
##
## Correlation between  Residence type and stroke
```

```
with (data1,cor(Residence_type,stroke))
```

```
## [1] 0.01545797
```

It can be explicitly seen that, people are evenly distributed among urban area and rural area. It can also said that Residence type and stroke are poorly correlated.

```
cat("\nTransforming bmi data to numeric")
```

2.4 BMI

```
##
## Transforming bmi data to numeric
```

```
suppressWarnings(data1$bmi <-  as.numeric(data1$bmi))

cat("\nReplacing  with mean value")
```

```
##
## Replacing  with mean value
```

```
data1$bmi[is.na(data1$bmi)] <- mean(data1$bmi,na.rm=TRUE)

cat("\n\nCorrelation between  BMI and stroke\n")
```

```
##
##
## Correlation between  BMI and stroke
```

```
with (data1,cor(bmi,stroke))
```

```
## [1] 0.0389466
```

BMI is one of the most important factor in this data set. One thing can be easily be distinguished from this column is that some of the data are missing. The best approach to settle this issue to use mean value in the place of missing values. Another thing should be noted that the BMI and stroke are less correlated.

```
table(factor(data1$smoking_status))
```

2.5 Smoking Status

```
##
## formerly smoked    never smoked          smokes         Unknown
##             885            1892             789            1544
```

```
cat("\nTable of Strokes corresponding to the smoking status")
```

```
##
## Table of Strokes corresponding to the smoking status
```

```
table(data1$smoking_status, data1$stroke)
```

```
##
##                      0    1
##   formerly smoked  815   70
##   never smoked    1802   90
##   smokes           747   42
##   Unknown         1497   47
```

```
data1$smoking_status <- as.character(data1$smoking_status)
for (i in 1:length(data1$id)) {
  if (data1$smoking_status[i] == "Unknown") {
    data1$smoking_status[i] <- 10
  }

    else if (data1$smoking_status[i] == "never smoked") {
    data1$smoking_status[i] <- 0
  }

  else if (data1$smoking_status[i] == "formerly smoked") {
    data1$smoking_status[i] <- 20
  }

  else if (data1$smoking_status[i] == "smokes") {
    data1$smoking_status[i] <- 30
  }
}
data1$smoking_status <- as.numeric(data1$smoking_status)


cat("\n\nCorrelation between  smoking status and stroke\n")
```

```
##
##
## Correlation between  smoking status and stroke
```

```
with (data1,cor(smoking_status,stroke))
```

```
## [1] 0.03068196
```

In this part of data cleaning, another issue appears to us. Here, a distinctive value which is **Unknown** means the patients are either feeling not to share the information or they genuinely don't know whether they have ever smoked before. By considering their response, this unknown shouldn't be merged or distributed among other sub category. To consider all the smoking related information, some discrete value has been assigned to all the sub holders. Unknown here comes between never smoked and formerly smoked.

2.6 Deleting Unrelated Column

By analyzing the available data, it can be decided that the column representing ID doesn't have any influence over the stroke. Additionally the work type column also has been transformed into three more column. As a result, column 7 can be easily removed.

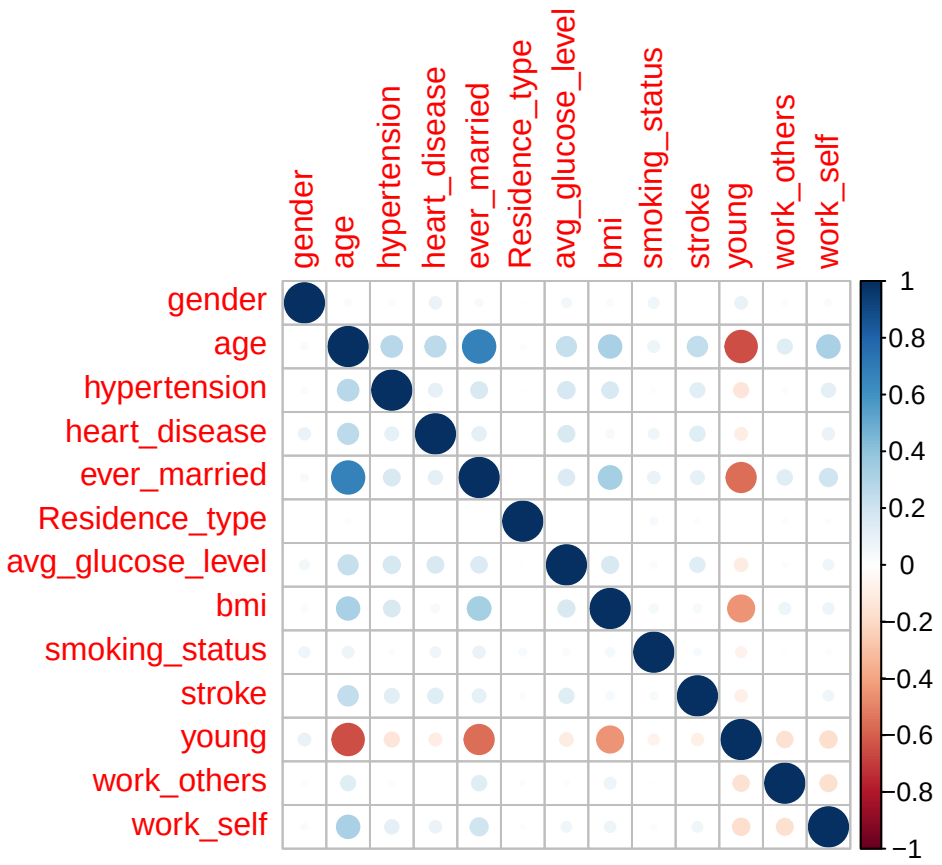
```
cat("Unused columns has been dropped")
```

```
## Unused columns has been dropped
```

```
data1 <- data1[ -c(1,7) ]
```

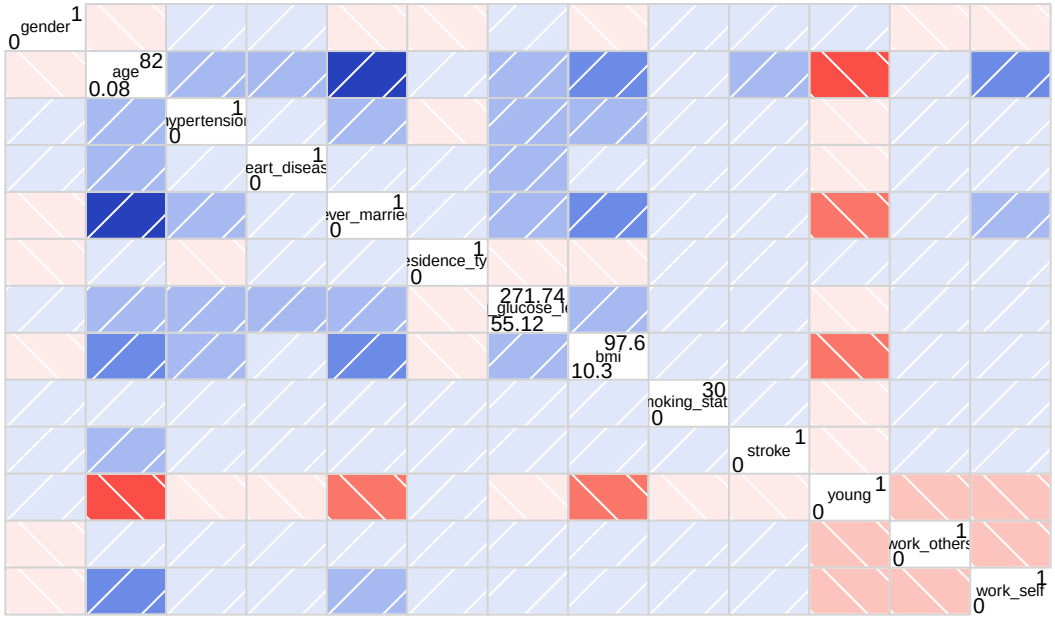
3. Correlation Graph

```
correlations <- cor(data1, method = "pearson", use = "complete.obs")
corrplot(correlations, method="circle")
```




```
corrplot(cor(data1), method = "number")
```


```
corrgram(data1, order=NULL, panel=panel.shade, text.panel=panel.txt,
         diag.panel=panel.minmax, main="Correlogram of Stroke Dataset")
```




```
round(cor(subset(data1, select=c(age,hypertension, heart_disease,avg_glucose_level, bmi, stroke))),2)
```

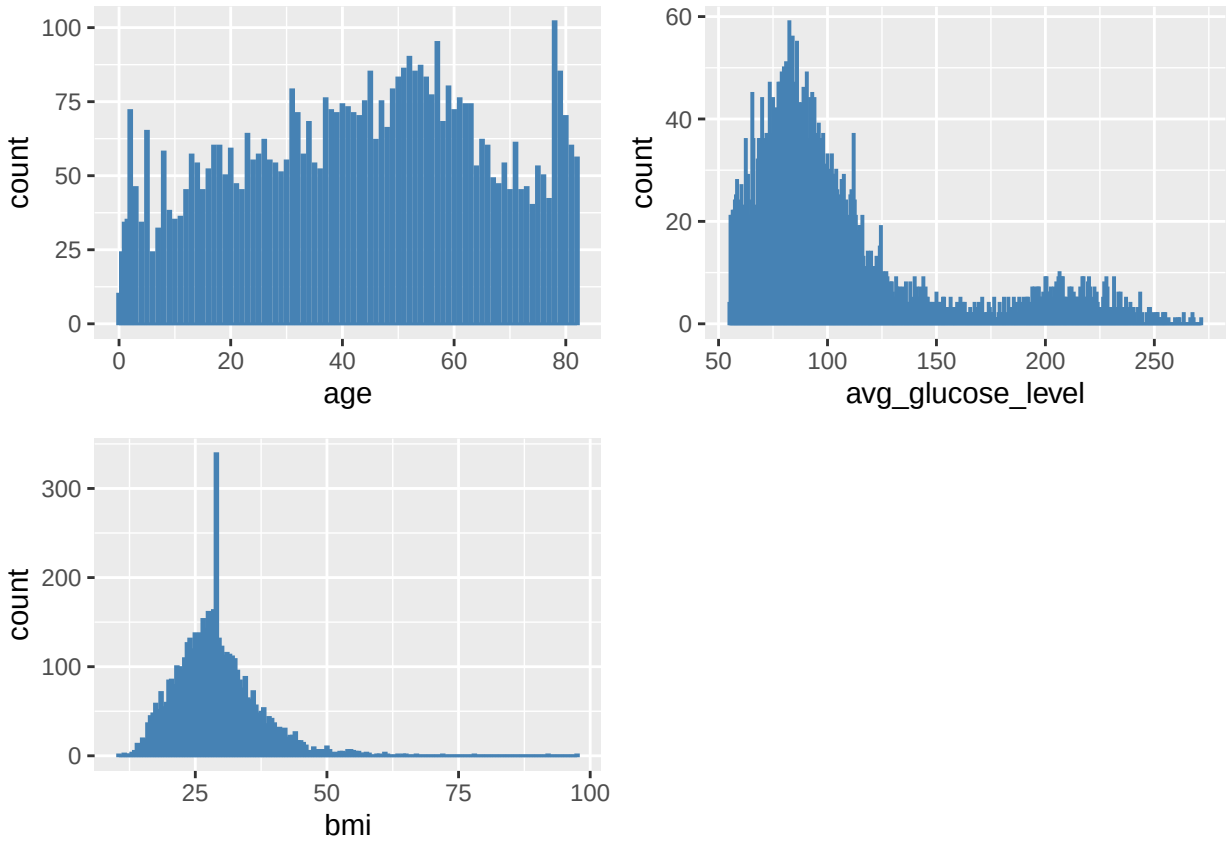
```
##                     age hypertension heart_disease avg_glucose_level  bmi stroke
## age                1.00         0.28          0.26              0.24 0.33   0.25
## hypertension       0.28         1.00          0.11              0.17 0.16   0.13
## heart_disease      0.26         0.11          1.00              0.16 0.04   0.13
## avg_glucose_level  0.24         0.17          0.16              1.00 0.17   0.13
## bmi                0.33         0.16          0.04              0.17 1.00   0.04
## stroke             0.25         0.13          0.13              0.13 0.04   1.00
```

From the correlation diagram and chart, it appears that data related to **age, hypertension, heart disease, average glucose level** are the main correlated factor for being vulnerable to have a stroke. So, these four features are being considered for the rest of the process.

One thing needs to be realized that correlation works for measurable data with meaningful numbers, often quantities of some kind. It cannot be utilized for solely categorical data, such as gender, purchased brands, or favorite color [2].

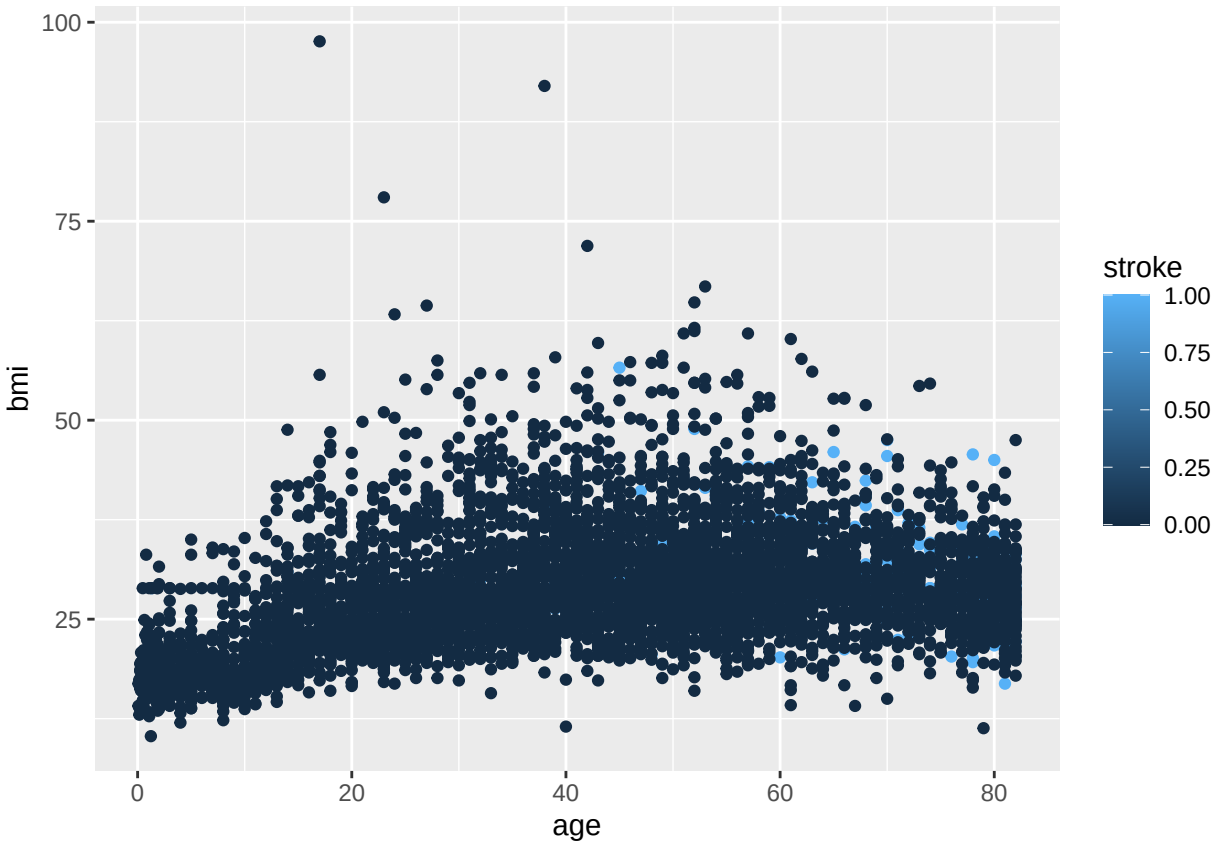
4. Exploratory Data Analysis

```
c1 <- ggplot(data = data1) + geom_histogram(mapping = aes(x = age), binwidth = 0.5, col = 'steelblue')
c2 <- ggplot(data = data1) + geom_histogram(mapping = aes(x = avg_glucose_level), binwidth = 0.5, col =
c3 <- ggplot(data = data1) + geom_histogram(mapping = aes(x = bmi), binwidth = 0.5, col = 'steelblue')
grid.arrange(c1,c2,c3, ncol= 2)
```



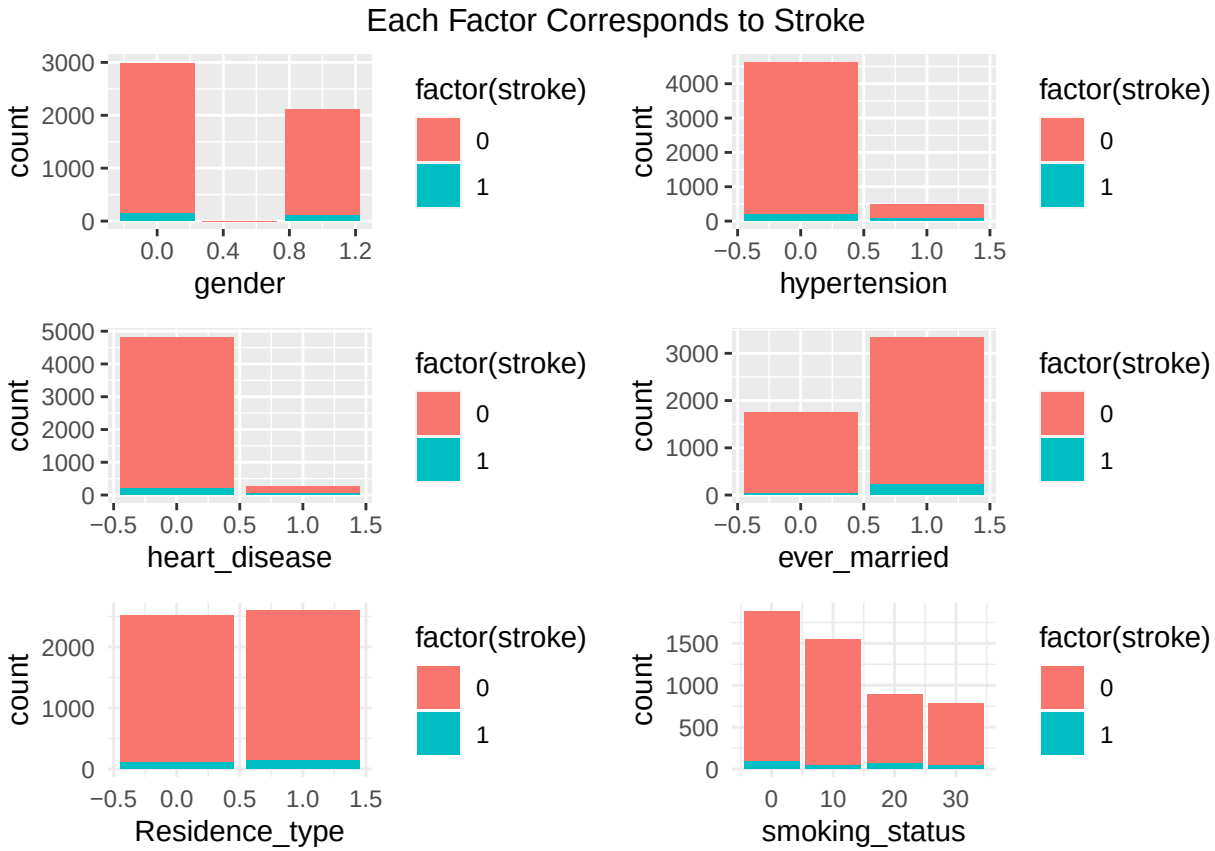
These graphs show that age is somewhat biased to the right, whereas glucose level and bmi are skewed to the left. The pike in the bmi plot is the consequence of substituting mean for NA.

```
cont.plot <- ggplot(data = data1, aes(x= age, y = bmi, color = stroke))+geom_point()
cont.plot
```



From this scatter plot, it can be seen that older people are likely to suffer from stroke regardless of their BMI level.

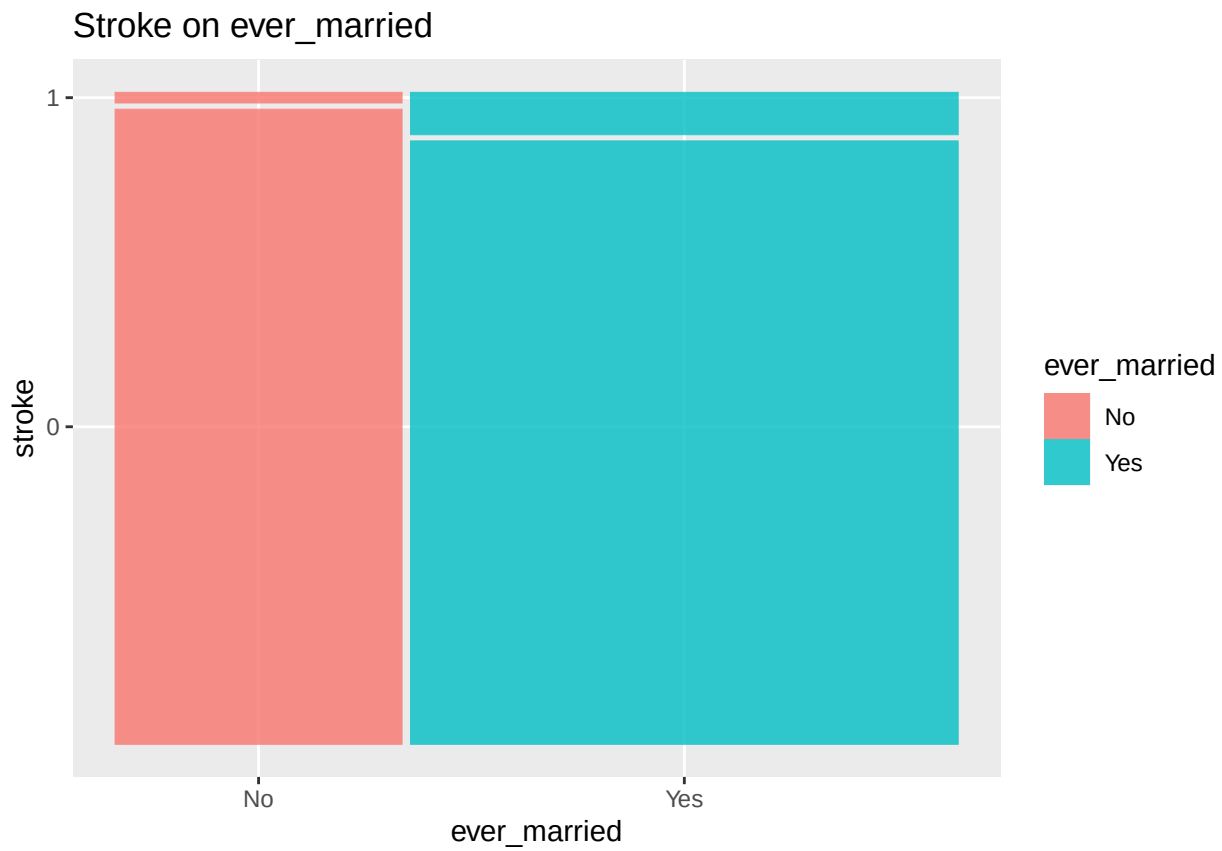
```
p1 <- ggplot(data = data1) +geom_bar(mapping = aes(x = gender, fill=factor(stroke)))
p2 <-ggplot(data = data1) +geom_bar(mapping = aes(x = hypertension, fill=factor(stroke)))
p3 <-ggplot(data = data1) +geom_bar(mapping = aes(x = heart_disease, fill=factor(stroke)))
p4 <-ggplot(data = data1) +geom_bar(mapping = aes(x = ever_married, fill=factor(stroke)))
p5 <-ggplot(data = data1) +geom_bar(mapping = aes(x = Residence_type, fill=factor(stroke)))+theme_minima
p6 <-ggplot(data = data1) +geom_bar(mapping = aes(x = smoking_status, fill=factor(stroke)))+theme_minima
grid.arrange(p1,p2,p3,p4,p5,p6, ncol= 2,top = "Each Factor Corresponds to Stroke")
```



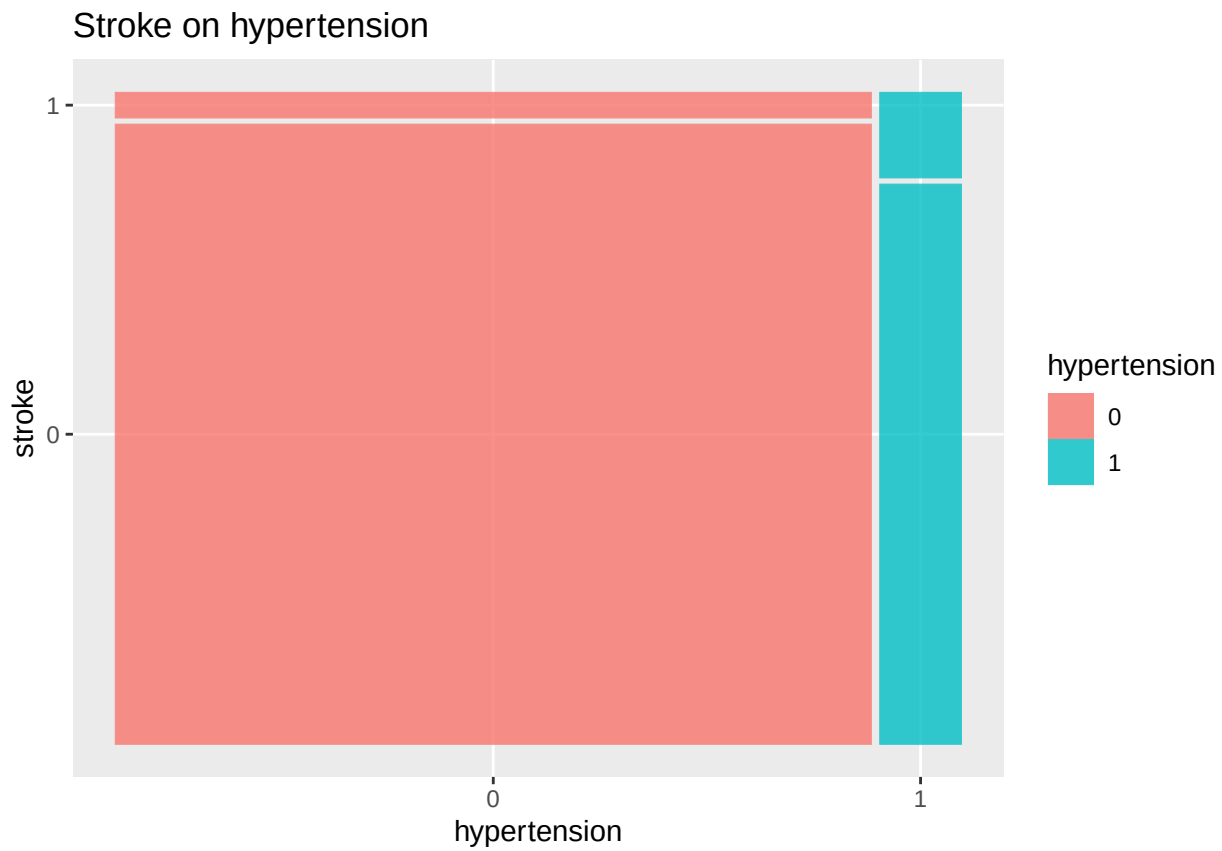
The amount of those who have had a stroke is a small portion of the population. This follows for the populations of different factors. Proportions of those who had a stroke in each factor has been compared.

```
data2 <- data
ggplot(data = data2) +geom_mosaic(aes(x = product(stroke,ever_married), fill=ever_married)) + labs(titl
```

```
## Warning: `unite_()` was deprecated in tidyr 1.2.0.
## Please use `unite()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```




```
data2$hypertension <- as.factor(data2$hypertension)
ggplot(data = data2) +geom_mosaic(aes(x = product(stroke,hypertension), fill=hypertension)) + labs(titl
```



The number of people who have suffered a stroke is likely to be married and experience hypertension.

```
data2$stroke  <- ifelse(data2$stroke  == 0, "No", 'Yes')
ggplot(data2, aes(x=stroke , y=age, fill = stroke)) +
  geom_violin(trim=FALSE, fill='#cacaca', color="darkred")+
  geom_boxplot(width=0.15) + theme_minimal()
```

From the violin plot, it is clear that patients who are above 25 are most likely to suffer from the stroke compared to young age people. The older the patient is, there is a higher likelihood to be diagnosed with stroke.

5. Training and testing data sets

Here, it can be seen that only 4 features has been considered as they are more correlated to stroke. 70% of the data will be used to train the model and the rest 30% of the data will be utilized to evaluate its performance.

```
data3 <- data1[,c("age","hypertension","avg_glucose_level","heart_disease","stroke")]
data_scaling1 <- scale(data3[,c(-5)])
data_scaled <- cbind(data_scaling1, data$stroke)

data_scaled <- as.data.frame(data_scaled)
data_scaled <- rename(data_scaled,"stroke" =  V5)


cat("Train - Test split has been done\n")
```

```
## Train - Test split has been done
```

```
trainIndex <- createDataPartition(data_scaled$stroke, p = 0.7,
                                  list = FALSE,
```

```
                                    times = 1)
dataTrain0 <- data_scaled[ trainIndex,]
dataTest  <- data_scaled[-trainIndex,]
```

One of the key issues that emerged while developing the model was a significant disparity between people who had suffered a stroke and those who had not. Oversampling may assist to solve this issue in this scenario, but it does not remove it.

```
cat("Oversampling the training data from 5% do 30%\n")
```

```
## Oversampling the training data from 5% do 30%
```

```
dataTrain <- ovun.sample(as.factor(stroke)~.,data = dataTrain0, method = 'over',p = 0.3)$data
```

6. Logistic Regression Model

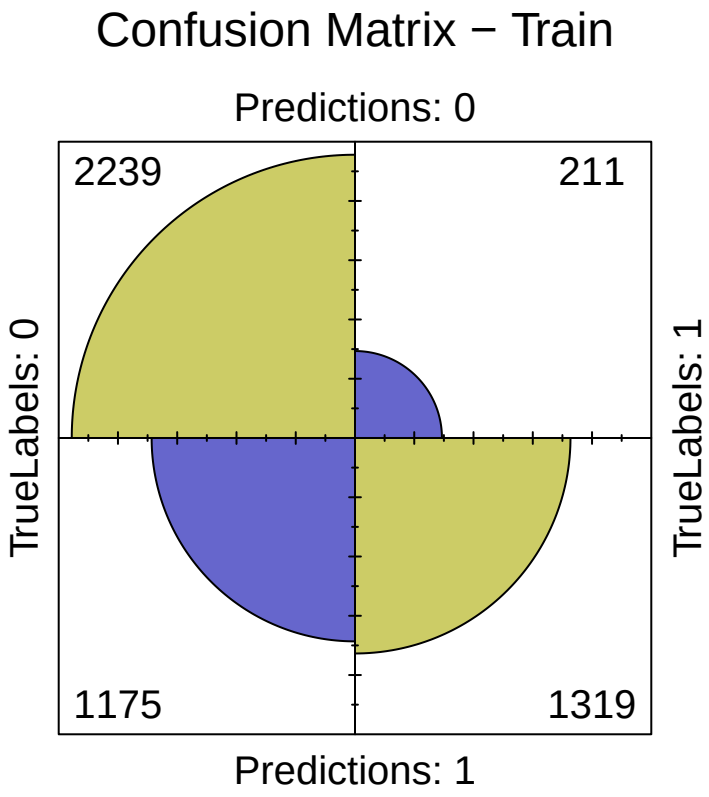
```
logit_2 <- glm(stroke~., family = binomial,data = dataTrain)

pred_train0 <- predict(logit_2, type = "response")
pred_train <- ifelse(pred_train0 >0.25,1,0)


pred_test0 <- predict(logit_2, newdata = dataTest, type = "response")
pred_test <- ifelse(pred_test0 >0.25,1,0)



a1 <- table(Predictions = pred_train, TrueLabels = dataTrain$stroke)
fourfoldplot(a1, color = c("#6666cc", "#cccc66"),
             conf.level = 0, margin = 1, main = "Confusion Matrix - Train")
```




```
acc1 <- ( a1[2,2] + a1[1,1] ) / (length(dataTrain$stroke))
cat("Accuracy = ",acc1 )
```

```
## Accuracy =  0.7196602
```

```
Precision1 =  a1[2,2] / ( a1[2,2] + a1[2,1] )
cat("\nPrecision = ",Precision1 )
```

```
##
## Precision =  0.5288693
```

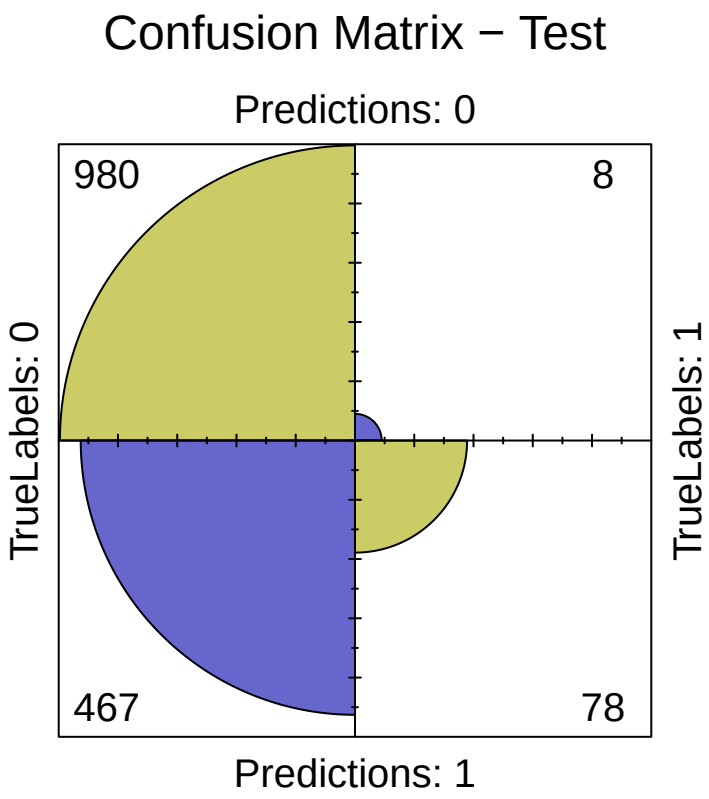
```
Recall1 =  a1[2,2] / ( a1[2,2] + a1[1,2] )
cat("\nRecall = ",Recall1 )
```

```
##
## Recall =  0.8620915
```

```
F1score1  = 2*(Recall1 * Precision1) / (Recall1 + Precision1)
cat("\nF1 score = ",F1score1 )
```

```
##
## F1 score =  0.6555666
```

```
b1 <- table(Predictions = pred_test, TrueLabels = dataTest$stroke)
fourfoldplot(b1, color = c("#6666cc", "#ccccc66"),
             conf.level = 0, margin = 1, main = "Confusion Matrix - Test")
```




```
acc1 <- ( b1[2,2] + b1[1,1] ) / (length(dataTest$stroke))
cat("Accuracy = ",acc1 )
```

```
## Accuracy =  0.69015
```

```
Precision1 =  b1[2,2] / ( b1[2,2] + b1[2,1] )
cat("\nPrecision = ",Precision1 )
```

```
##
## Precision =  0.1431193
```

```
Recall1 =  b1[2,2] / ( b1[2,2] + b1[1,2] )
cat("\nRecall = ",Recall1 )
```

```
##
## Recall =  0.9069767
```

```
F1score1  = 2*(Recall1 * Precision1) / (Recall1 + Precision1)
cat("\nF1 score = ",F1score1 )
```

```
##
## F1 score =  0.2472266
```

The model performed with poor accuracy, precision and F1 score. But it has better recall. In this part, we should consider recall mainly as we do not want to inform a patient who is vulnerable to have stroke that he doesn't have a risk to have stroke. So, out of all the metric we should emphasis on the recall mainly.

7. Random forest Model

A random forest model has been developed using the over sampled data. This model uses 400 trees in order to make predictions. One of the main reason to use random forest model over other model is to increase accuracy. But is has some drawback. For example, it takes longer time to calculate results.

Random forest transform data back to categorical. It needs to be dealt carefully.

```
data3$hypertension <- as.factor(data3$hypertension)
data3$heart_disease <- as.factor(data3$heart_disease)
data3$stroke <- as.factor(data3$stroke)


trainIndex2 <- createDataPartition(data3$stroke, p = 0.7,
                                   list = FALSE,
                                   times = 1)
dataTrain02 <- data3[ trainIndex2,]
dataTest2  <- data3[-trainIndex2,]
```

```
cat("Balance (train) data by oversampling. From 5% do 30%\n")
```

```
## Balance (train) data by oversampling. From 5% do 30%
```

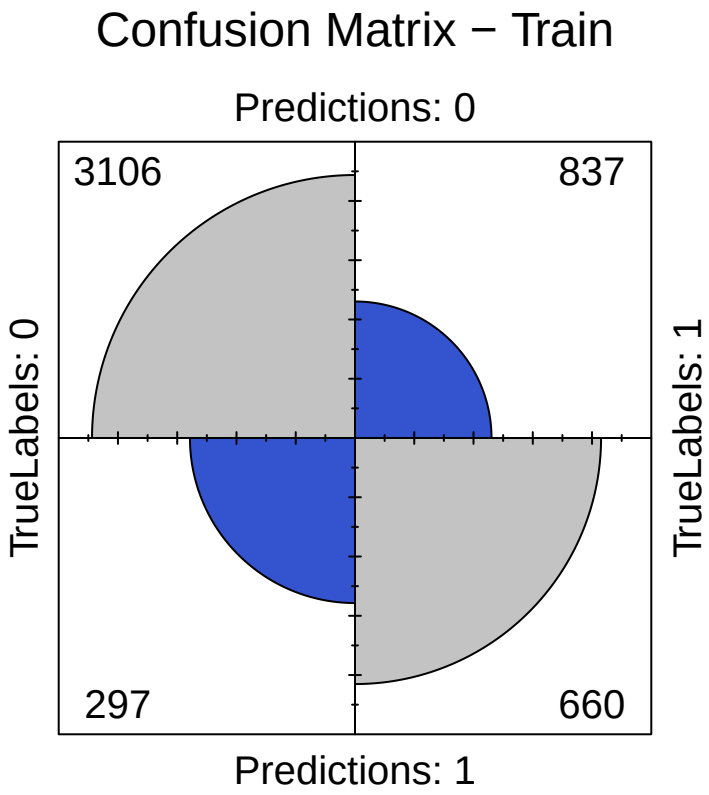
```
dataTrain2 <- ovun.sample(as.factor(stroke)~.,data = dataTrain02, method = 'over',p = 0.3)$data

rf <- randomForest(stroke ~., data = dataTrain2, ntree = 400, mtry = 1)
```

```
predRF_train <- predict(rf)

predRF_test <- predict(rf, newdata = dataTest2)
```

```
a2 <- table(Predictions = predRF_train, TrueLabels = dataTrain2$stroke)
fourfoldplot(a2, color = c("#3454cf", "#c3c3c3"),
             conf.level = 0, margin = 1, main = "Confusion Matrix - Train")
```




```
acc2 <- ( a2[2,2] + a2[1,1] ) / (length(dataTrain2$stroke))
cat("Accuracy = ",acc2 )
```

```
## Accuracy =  0.7685714
```

```
Precision2 =  a2[2,2] / ( a2[2,2] + a2[2,1] )
cat("\nPrecision = ",Precision2 )
```

```
##
## Precision =  0.6896552
```

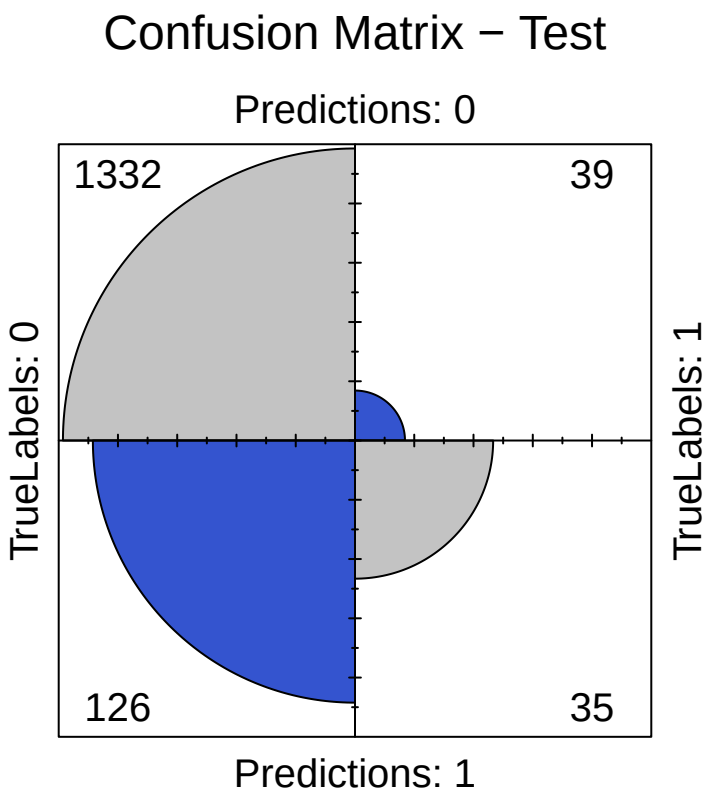
```
Recall2 =  a2[2,2] / ( a2[2,2] + a2[1,2] )
cat("\nRecall = ",Recall2 )
```

```
##
## Recall =  0.4408818
```

```
F1score2  = 2*(Recall2 * Precision2) / (Recall2 + Precision2)
cat("\nF1 score = ",F1score2 )
```

```
##
## F1 score =  0.5378973
```

```
b2 <- table(Predictions = predRF_test, TrueLabels = dataTest2$stroke)
fourfoldplot(b2, color = c("#3454cf", "#c3c3c3"),
             conf.level = 0, margin = 1, main = "Confusion Matrix - Test")
```




```
acc2 <- ( b2[2,2] + b2[1,1] ) / (length(dataTest2$stroke))
cat("Accuracy = ",acc2 )
```

```
## Accuracy =  0.8922977
```

```
Precision2 =  b2[2,2] / ( b2[2,2] + b2[2,1] )
cat("\nPrecision = ",Precision2 )
```

```
##
## Precision =  0.2173913
```

```
Recall2 =  b2[2,2] / ( b2[2,2] + b2[1,2] )
cat("\nRecall = ",Recall2 )
```

```
##
## Recall =  0.472973
```

```
F1score2  = 2*(Recall2 * Precision2) / (Recall2 + Precision2)
cat("\nF1 score = ",F1score2 )
```

```
##
## F1 score =  0.2978723
```

It can be seen that, Random forest model has better accuracy compared to Logistic Regression. But it has low recall.

8. Conclusion

Our goal is to minimize the false negatives. Because a person who has been falsely predicted to have stroke will be more careful to his/her diet, physical work and so on. On the other side, if the model predicts that a highly vulnerable person to the stroke would not have stroke, there is a high possibility that that person would not be careful towards his/her health. So, our priority should be minimizing the false negative while maintaining high accuracy. After fulfilling this criteria we should then focus on the precision and F1 score.

Overall this data set shows that people while aging are likely to be affected by the stroke. So, they should be careful about their physical and mental health. We should treat our body and soul equally. That is why it is recommended for moderately aged people to avoid eating unhealthy food, lead life with minimum mental pressure and continuously take part in physical exercise.

9. Reference

[1] Donkor E. S. (2018). Stroke in the 21st Century: A Snapshot of the Burden, Epidemiology, and Quality of Life. *Stroke research and treatment*, *2018*, 3238165. https://doi.org/10.1155/2018/3238165 [2] Dealing with Missing Values · UC Business Analytics R Programming Guide. (2022). Retrieved 25 March 2022, from https://uc-r.github.io/missing_values

[3] KUMAR KATURU, L. (2022). Stroke Prediction. Retrieved 25 March 2022, from https://www.kaggle.com/code/klprasanna1963/stroke-prediction-in-r-with-model-evaluation [4] ZEFERINO, J. (2022). Stroke-pred: LogReg vs RandomForest - R. Retrieved 25 March 2022, from https://www.kaggle.com/code/jzeferino/stroke-pred-logreg-vs-randomforest-r