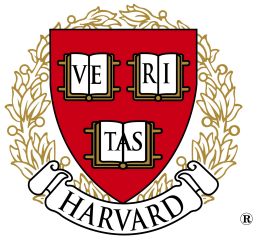


Stroke Report

Ing. José Ramón Riesgo Escovar

May 2020

INTRODUCTION TO STROKE

Stroke is one of the leading causes of death globally. Sometimes it is also called a brain attack. A stroke happens when something is blocking the blood supply to a part of the brain. Also a Stroke happens when a vessel in the brain bursts. In either case, this stops the flow of blood and due to this, part of the brain becomes damage or dies. Unfortunately when this happens it can cause lasting brain damage, long-term disability and even in some cases the death. Our brain controls our movements, stores our memories, and it is the source of our thoughts, emotions and our language. Besides these functions also the brain controls functions like breathing and controls our digestion.

Our brain uses 20% of the oxygen we breathe and the arteries deliver our oxigen-rich blood to all the sections of our brain.

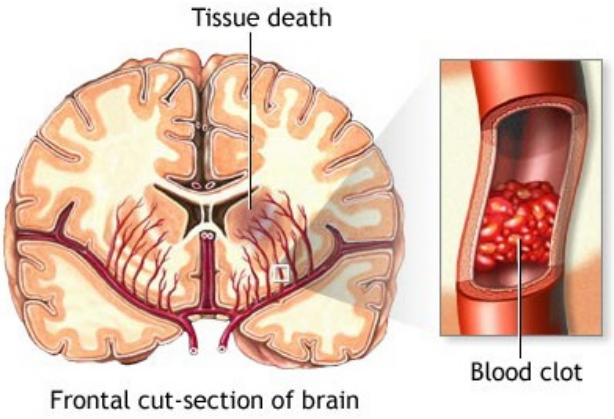


Image taken from CDC.gob

A stroke happens when a blood clot blocks blood flow to the brain or when there is a bursts that prevents the flow of blood

The stroke is the third major cause of disability. Long term disability affects people severely, in terms of their productive life. The aim of this report is to identify the risk factors and with them be able to predict if someone has high risk of having a Stroke.

The patient dataset was obtain from **Kaggle** the process and methods to ascertain whether a variable is a risk factor will be evaluated and described. We will visualized and discovered insights of the dataset, ending with a conclusion and some ideas and suggestions for future work on the reserach for early symptons that could help prevent an actual Stroke.

From Kaggle we get the description of the columns in our dataset:

1) id: unique identifier
2) gender: Male, Female or Other
3) age: age of the patient
4) hypertension: 0 if the patient doesn't have hypertension, 1 if the patient has hypertension
5) heart_disease: 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease
6) ever_married: "No" or "Yes"
7) work_type: "children", "Govt_jov", "Never_worked", "Private" or "Self-employed"
8) Residence_type: "Rural" or "Urban"
9) avg_glucose_level: average glucose level in blood
10) bmi: body mass index
11) smoking_status: "formerly smoked", "never smoked", "smokes" or "Unknown"*
12) stroke: 1 if the patient had a stroke or 0 if not

*Note: "Unknown" in smoking_status means that the information is unavailable for this patient

In the next section we will first do a data preprocessing and then we will execute the exploratory data analysis and then work on the models.

DATA PREPROCESSING ANALYSIS:

The Dataset consists of:

```
stroke_data <- read.csv("stroke.csv", header = TRUE)
dim(stroke_data)
```

```
## [1] 5110   12
```

We have 5110 records and 12 columns, 11 potential predictors and the column that indicates if the patient got actually a stroke or not.

Now we will be exploring one by one the potential predictors:

1) ID : Unique Identifier

This is a number to identify the patient but it is irrelevant because does not provide any meaningful information for our future models so we will delete this column from our datset by applying this code:

```
stroke_data <- subset( stroke_data, select = -id )
```

2) Gender: The specific gender of the patients

With the following will generate the actual distribution of the gender within the dataset:

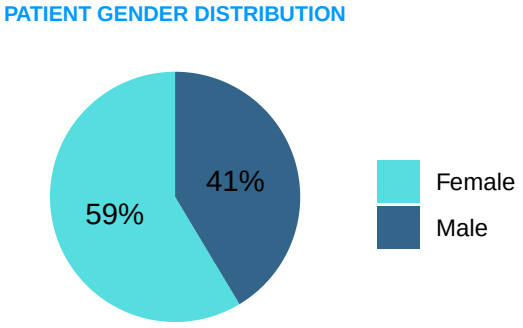
```
stroke_data %>%
  group_by(gender) %>%
  summarise(total = n())
```

```
## # A tibble: 3 x 2
##   gender total
##   <chr>  <int>
## 1 Female  2994
## 2 Male    2115
## 3 Other      1
```

We see that there is just one record of an "Other" gender, having just one record is insignificant to the dataset and the future predicion models so we will eliminate it from the dataset using this code:

```
stroke_data <- subset (stroke_data, stroke_data$gender !="Other")
```

Now the dataset has the following distribution of Gender of Patients:



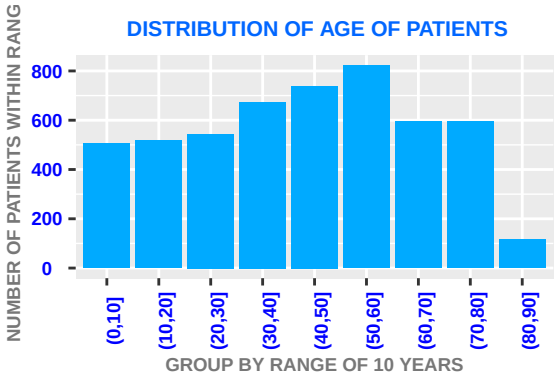
3) Age: Distribution of the age of the patients

Generating the summary of how the age is distributed within our patients

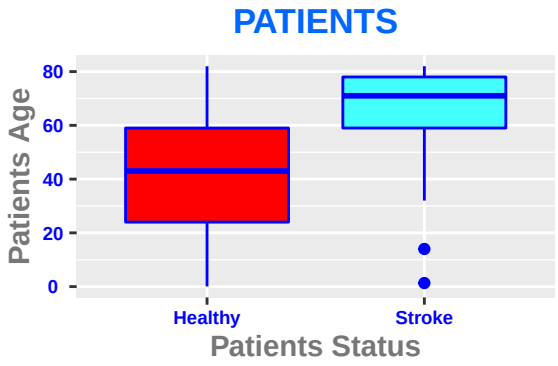
```
summary(stroke_data$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.08   25.00   45.00   43.23   61.00   82.00
```

The next chart shows how the patients are distribute by grouping them in ranges of 10 years of age



With the following BoxPlot we can see very clear that there is a tendency for older patient to have a Stroke:



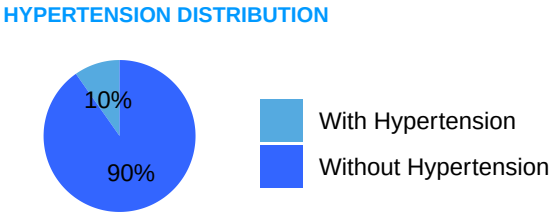
4) Hypertension: Information if the patient had hypertension or not

With the following code we visualized the total patients that have hypertension and the ones without hypertension

```
stroke_data %>%
  mutate(text = ifelse(hypertension==0,"Without Hypertension","With Hypertension")) %>%
  group_by(text) %>%
  summarise(total = n())
```

```
## # A tibble: 2 x 2
##   text                total
##   <chr>               <int>
## 1 With Hypertension     498
## 2 Without Hypertension  4611
```

Showing in a pie chart the hypertension distribution within our set:



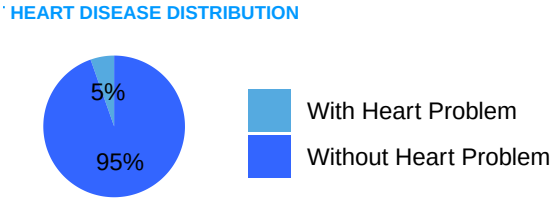
5) Heart Disease: Information if the patient had a problem in his heart or not

With the following code we visualized the total patients that have heart problems and the ones without any problem in their heart

```
stroke_data %>%
  mutate(text = ifelse(heart_disease==0,"Without Heart Problem","With Heart Problem")) %>%
  group_by(text) %>%
  summarise(total = n())
```

```
## # A tibble: 2 x 2
##   text                  total
##   <chr>                 <int>
## 1 With Heart Problem      276
## 2 Without Heart Problem  4833
```

The following pie chart shows the distribution of the patients with problems in their heart:



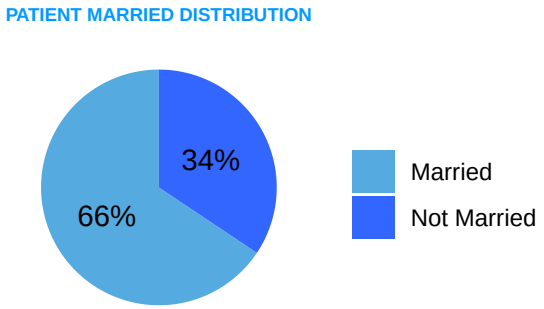
6) ever_married: Information if the patient had been married or not

With the following code we are able to visualize how many patients were married and how many were never married:

```
stroke_data %>%
  mutate(text = ifelse(ever_married=="Yes","Married","Not Married")) %>%
  group_by(text) %>%
  summarise(total = n())
```

```
## # A tibble: 2 x 2
##   text        total
##   <chr>       <int>
## 1 Married      3353
## 2 Not Married  1756
```

The following pie chart shows the distribution of the Patients that were Married against the ones that were never married:



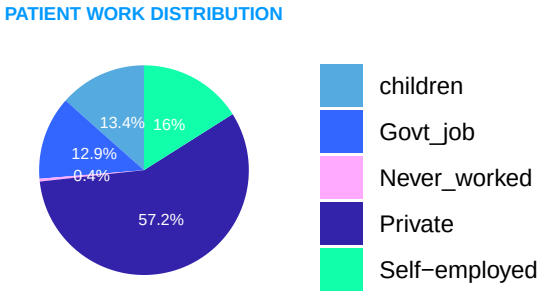
7) work_type: We will show the information if the patient have work and what type of work

With the following code shows the amount of the patients by their work situation unless they are children:

```
stroke_data %>%
  group_by(work_type) %>%
  summarise(total = n())
```

```
## # A tibble: 5 x 2
##   work_type     total
##   <chr>         <int>
## 1 children        687
## 2 Govt_job        657
## 3 Never_worked     22
## 4 Private        2924
## 5 Self-employed   819
```

With the following code we will show a pie chart with the work distribution of the patients:



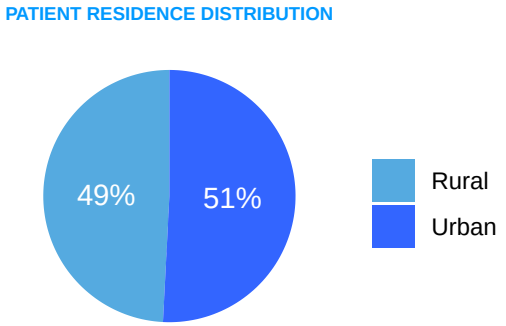
8) Residence_type: defines where the patients life in urban area or rural area

With the following code summarizes where patients residence is:

```
stroke_data %>%
  group_by(Residence_type) %>%
  summarise(total = n())
```

```
## # A tibble: 2 x 2
##   Residence_type total
##   <chr>          <int>
## 1 Rural           2513
## 2 Urban           2596
```

The following is a pie chart showing the distribution of the patients residence:



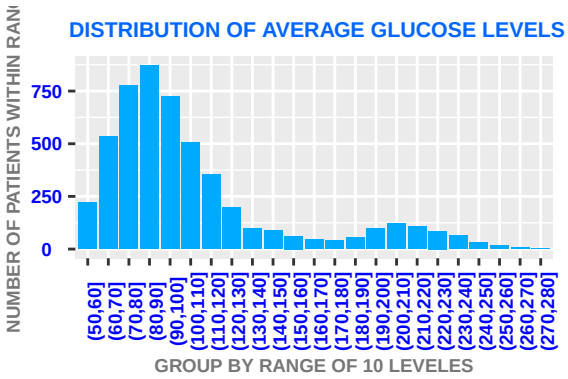
9) avg_glucose_level: defines the glucose levels of the patients in the set

With the following code we generate the summary of of the statistics of the average glucose level with the dataset:

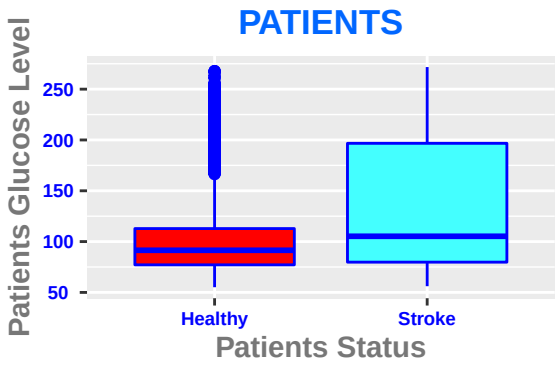
```
summary(stroke_data$avg_glucose_level)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   55.12   77.24   91.88  106.14  114.09  271.74
```

The following chart shows the distribution in ranges from the lower limit to the higher limit with increments of 10 units:



We show a box plot of the patients that had a stroke as well as the ones healthy:



The patients with high level of glucose tend to be more prone to have a Stroke.

10) bmi: defines the Body Mass Index of the patients in the set

Reviewing the information of the Body Mass Index (bmi) levels, there are patients without this information with an N/A in these row and also we identify that the values of the bmi are strings within the dataset instead of numbers, with the following code we are showing the summary of bmi

```
# To avoid warning due to N/A will disable them during this code execution
options(warn = -1) # To avoid Warning due to N/A
summary(as.numeric(stroke_data$bmi), na.rm=TRUE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   10.30   23.50   28.10   28.89   33.10   97.60     201
```

With the following code we will review what patients do not have bmi information and had not had any stroke:

```
# How many records we have without bmi information and that have not had any stroke
sum(stroke_data$bmi=="N/A" & stroke_data$stroke==0)
```

```
## [1] 161
```

With the following code we will review what patients that do not have bmi information and had a stroke:

```
# How many records we have without bmi information and that had a stroke
sum(stroke_data$bmi=="N/A" & stroke_data$stroke==1)
```

```
## [1] 40
```

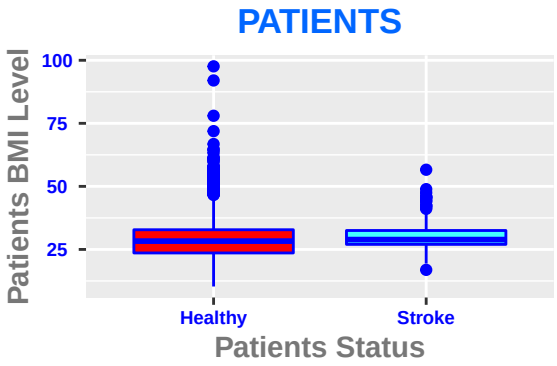
Due that these 201 registries (161 that have not suffer a stroke and 40 that suffer a stroke) we consider this relevant for our study and predictions of Stroke, so we will be calculate the average bmi of the dataset using this code:

```
# Calculate the mean of bmi
bmimean <- mean(as.numeric(stroke_data$bmi), na.rm = TRUE)
```

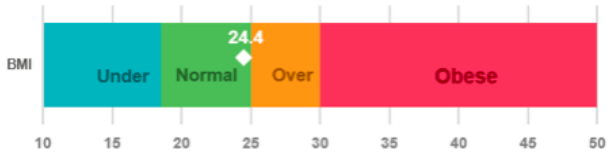
Adding a new column in our dataset with the patients bmi number, instead of the string and also for the patients without bmi information the mean of bmi will be assigned to them. The following code achieves this:

```
stroke_data <- stroke_data %>%
  mutate( bmi_num = ifelse(bmi=="N/A",bmimean,as.numeric(bmi)))
# Return to normal warnings
options(warn = 0L)
# Clearing the temporary variable bmimean to keep the environment clean
rm(bmimean)
```

The following boxplot shows the distribution of the patients that had a stroke and the healthy ones:



In a research of the BMI information we see the following pictures:

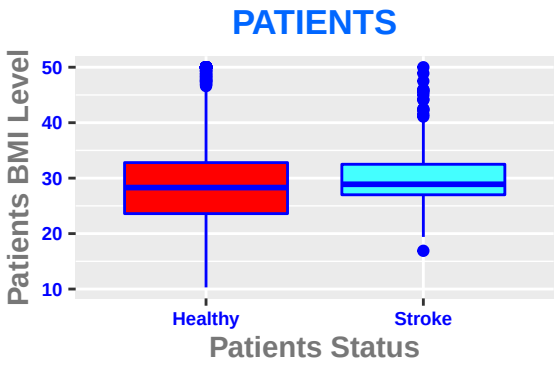


As seen here, anyone above 50 is extremely obese, so values on top of 56 of BMI seem out of range All the outliers of patients above 50 will be adjusted 56 to avoid distortion in the models

With the following code we will adjust this:

```
stroke_data <- stroke_data %>%
  mutate(bmi_num = ifelse(bmi_num >=50,50, bmi_num))
```

Showing the boxplot after the adjustment:



We do not see any specific trend with the BMI distribution between healthy patients and patients that had a stroke.

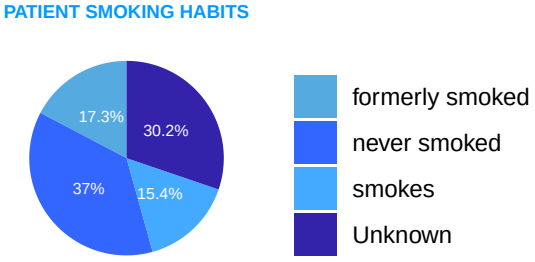
11) smoking_status: defines the patient relation to smoking

With the following code we show the distribution of the patients based on their smoking status:

```
stroke_data %>%
  group_by(smoking_status) %>%
  summarise(total = n())
```

```
## # A tibble: 4 x 2
##   smoking_status  total
##   <chr>           <int>
## 1 formerly smoked   884
## 2 never smoked     1892
## 3 smokes            789
## 4 Unknown          1544
```

In the following pie chart we observe the distribution of the smoking habits of the patients:



Note: "Unknown" in smoking_status means that the information is unavailable for this patient

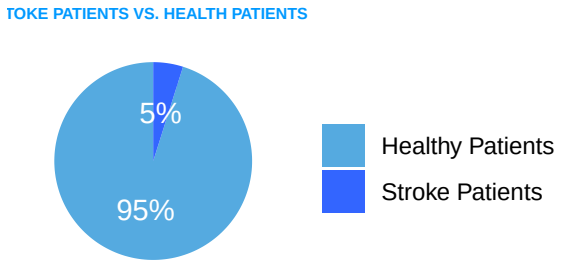
Stoke Patient in the Dataset:

With the following code we visualize the amount of stoke patients against the health patients in the dataset:

```
stroke_data %>%
  mutate(text = ifelse(stroke==0,"Healthy Patients","Stroke Patients")) %>%
  group_by(text) %>%
  summarise(total = n())
```

```
## # A tibble: 2 x 2
##   text             total
##   <chr>            <int>
## 1 Healthy Patients  4860
## 2 Stroke Patients    249
```

The following Pie chart show the distribution between Stroke Patients and Healthy Patients



General and preparation for models

For a correlation check it only accepts numerical variables and also for fitting models, so we are preprocessing all categorical variables to numbers, encoding them. Also we will scale age, avg_glucose_level and bmi because if we keep predictors that are measured at different scales they will not contribute equally to our fitting models and could create a bias. To deal with this possible problem we will standardized the age, avg_glucose_level and bmi to have a $(\mu = 0, \sigma = 1)$ before we start the fitting of the models

The following code achieves all the transformation:

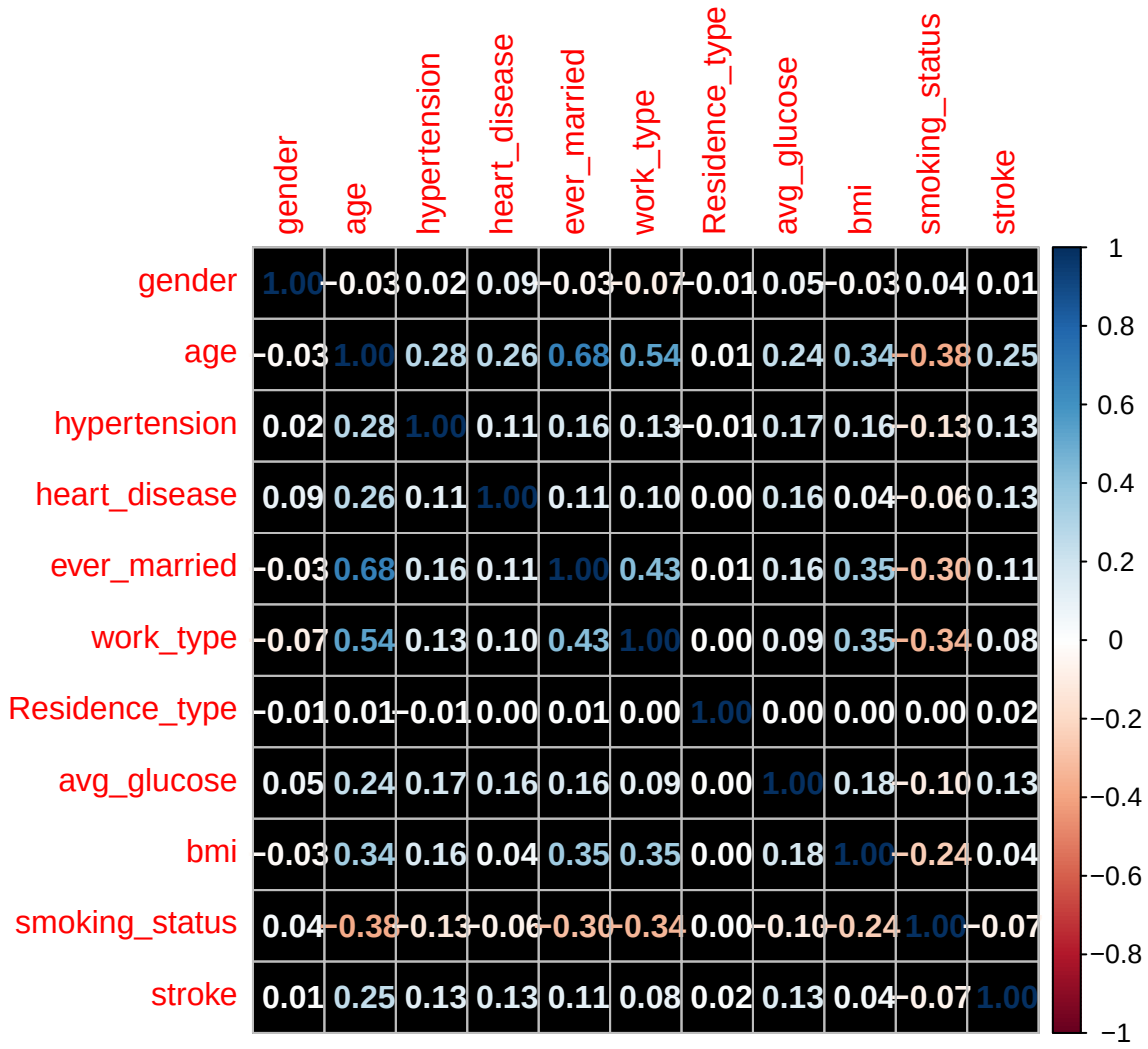
```
# mean of age
age_mean <- mean(stroke_data$age)
# sd of age
age_sd <- sd(stroke_data$age)
# mean of glucose
glucose_mean <- mean(stroke_data$avg_glucose_level)
# sd of glucose
glucose_sd <- sd(stroke_data$avg_glucose_level)
# mean of bmi
bmi_mean <- mean(stroke_data$bmi_num)
# sd of bmi
bmi_sd <- sd(stroke_data$bmi_num)

# We need to change our categorical variables to
stroke_data_num <- stroke_data %>%
  # Gender: Female 0, Male 1
  mutate(gender_num=ifelse(gender=="Female",0,1)) %>%
```

```
  # Married: Not_Married 0, Married 1
  mutate(married_num=ifelse(ever_married=="Yes",1,0)) %>%
  # In the following section we will be passing from text to numbers of work type
  # children 0, Govt_job 1, Never_worked2, Private 3, Self-employed 4
  mutate(work_type_num=sapply(work_type, function(x)
    switch(x,"children"= 0,"Govt_job"= 1,"Never_worked"= 2,"Private"= 3,
           "Self-employed"= 4))) %>%
  # Residence_type: Rural 0, Urban 1
  mutate(Residence_type_num=ifelse(Residence_type=="Urban",1,0)) %>%
  # In the following section we will be passing from text to numbers of smoking status
  # formerly smoked 0, never smoked 1, smokes 2, Unknown 3
  mutate(smoking_status_num=sapply(smoking_status, function(x)
    switch(x,"formerly smoked"= 0,"never smoked"= 1,"smokes"= 2,"Unknown"= 3))) %>%
  # Adjust/Fit the values of age
  mutate(age_fit=((age - age_mean)/age_sd)) %>%
  # Adjust/Fit the values of glucose
  mutate(glucose_fit=((avg_glucose_level - glucose_mean)/glucose_sd)) %>%
  # Adjust/Fit the values of bmi
  mutate(bmi_fit=((bmi_num - bmi_mean)/bmi_sd)) %>%
  dplyr::select(gender=gender_num, age=age_fit, hypertension,
         heart_disease, ever_married=married_num,
         work_type=work_type_num, Residence_type=Residence_type_num,
         avg_glucose= glucose_fit, bmi = bmi_fit,
         smoking_status=smoking_status_num, stroke )

# We remove the temporary values to keep as clean as possible the environment
rm(age_mean,age_sd,bmi_mean,bmi_sd,glucose_mean,glucose_sd)
```

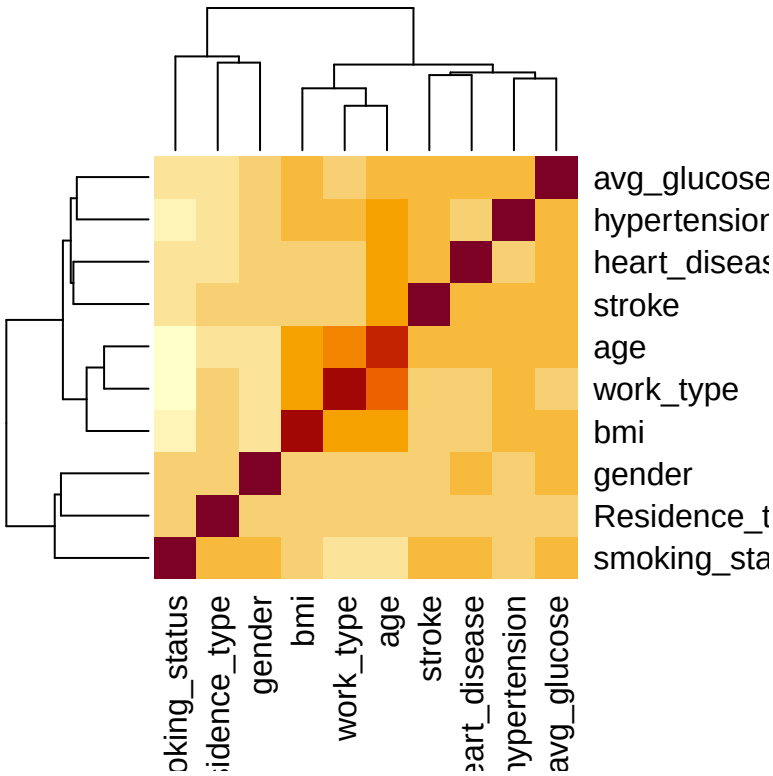
This is the correlation matrix:



There seems to be Multicollinearity between age and ever_married because we have a high correlation of 0.68 in principle age contains more information if a patient is susceptible to a stroke and we will discard ever_married from our predictors, the following code will execute this:

```
stroke_data_num <- subset( stroke_data_num, select = -ever_married )
```

The following is the Headmap of the predictors:



The full dataset will be split to 80% of a training set and 20% test sets by using the following code:

```
# Set seed to 5 to have always the same results and it generate
# and already review that there is a  balance betweeen the Stroke patients of
# the train and test set
set.seed(5, sample.kind = "Rounding")
#We need to first create a partition of the dataset for training 80% and 20% testing
test_index <- createDataPartition(stroke_data_num$stroke, times = 1, p = 0.2, list = FALSE)
# Generate the sets for the models:
# For convenience and more clarity we will split in "x" the predictors and
# "y" the actual value of stroke or not stoke
test_x <- stroke_data_num[test_index,1:9]
test_y <- stroke_data_num[test_index,10]
train_x <- stroke_data_num[-test_index,1:9]
train_y <- stroke_data_num[-test_index,10]
```

Assigning "S" to the patients with Stroke and and "H" to the Healthy patients, this will help in having a Categorical response instead of 0 and 1, so that the models generate "S" or "H" as the output of the models, this code will generate this changes to the dataset of Testing and Training.

```
train_y <- ifelse(train_y==1,"S","H")
test_y  <- ifelse(test_y==1,"S","H")
```

Verify the percentage of stroke patients in the training set and also the amount of stroke patients by executing this code:

```
sum(train_y=="S")/length(train_y)
```

```
## [1] 0.04869097
```

```
# How many stroke patients in Training set
sum(train_y=="S")
```

```
## [1] 199
```

Verify the percentage of stroke patients in the test set and also the amount of stroke patients by executing this code:

```
sum(test_y=="S")/length(test_y)
```

```
## [1] 0.04892368
```

```
# How many stroke patients in Test set
sum(test_y=="S")
```

```
## [1] 50
```

We have a very reasonable balance between both sets, almost the same percentage of stroke patients

MODELS WITH THE FULL SET

We will run the following models:

1) Generalized Linear Model:

The following code executes the Model and shows the accuracy and the actual predictions for stroke:

```
# We apply the method "glm" to the training set
generate_glm <- train(train_x, train_y, method = "glm")
# Then with the generated model we create the predictions for the test set
glm_predictions <- predict(generate_glm, test_x)
# We calculate the accuracy of the prediction
mean(glm_predictions == test_y)
```

```
## [1] 0.9510763
```

```
# How many patients calculate with stroke
sum(glm_predictions=="S")
```

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

2) Generalized Additive Model for LOESS:

The following code executes the Model and shows the accuracy and the actual predictions for stroke:

```
# We apply the method "gamLoess" to the training set
generate_gamloess <- train(train_x, train_y, method = "gamLoess")
# Then with the generated model we create the predictions for the test set
gamloess_predictions <- predict(generate_gamloess, test_x)
# We calculate the accuracy of the prediction
mean(gamloess_predictions == test_y)
```

```
## [1] 0.9510763
```

```
# How many patients calculate with Stroke
sum(gamloess_predictions=="S")
```

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

3) K-Nearest Neighbo (knn)r:

The following code executes the Model and shows the accuracy and the actual predictions for stroke:

```
# We apply the method knn to the training set
generate_knn <- train(train_x, train_y, method = "knn",
                      tuneGrid = data.frame(k = seq(1,40,2)))
# Then with the generated model we create the prediction for the test set
```

```r
knn_predictions <- predict(generate_knn, test_x)
# We calculate the accuracy of the prediction
mean(knn_predictions == test_y)
```

```
## [1] 0.9510763
```

```r
# How many patients calculate with Stroke
sum(knn_predictions=="S")
```

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

```r
#Cleaning all the used variables to keep the environment tidy
rm(generate_glm,generate_knn,generate_gamloess)
rm(gamloess_predictions,glm_predictions,knn_predictions)
```

Stopping the execution because we are not really predicting, we are just generating all predictions as Healthy, and the dataset has a bias, because there is a big distribution of 95% of the dataset as Healthy so really we are not generating any predictions for Stroke patients. So we are going to adjust the dataset to have close to 50% patients with stroke and 50% of aleatory healthy patients, so our models will predict the potential stroke and healthy patients based on the 9 predictors.

Adjusting the dataset to produce a more balance dataset between Stroke and Healthy Patients:

Passing all the stroke patients to a temporary dataset with the following code:

```r
# Pass all Stroke patients to a dataset subset
positive_stroke_patients <- stroke_data_num %>%
  filter(stroke==1)
```

Calculating the amount of patients with strokes with the following code:

```r
nrow(positive_stroke_patients)
```

```
## [1] 249
```

The set has 249 patients with Stroke

Passing all the Healthy patients to a temporary dataset with the following code:

```r
# Generate a subset of all helthy patients
health_stroke_patients <- stroke_data_num %>%
  filter(stroke==0)
```

Based on the full size of this healthy dataset we are going to partition 6% of the set and with that produce a similar size dataset to the stroke dataset with the following code:

```r
# Set the seed to 3
set.seed(3, sample.kind = "Rounding")
# Generate a partition of 6% of the healthy patients around 250 registries
health_index <-createDataPartition(health_stroke_patients$age, times = 1, p = 0.06, list = FALSE)
# Generate this new healthy patients subset
health_stroke_patients <- health_stroke_patients[health_index,1:10]
```

Calculating the amount of healthy patients with the following code:

```r
nrow(health_stroke_patients)
```

```
## [1] 294
```

The set has 294 healthy patients

Now with the following code we combine both sets to have a more balance dataset and without any bias for healthy patients and also calculate the count of patients:

```
# Combine the stroke patients with the generated subset
stroke_data_num_adj <- positive_stroke_patients %>%
  union(health_stroke_patients)
# Visualize the new size of the dataset
nrow(stroke_data_num_adj)
```

```
## [1] 543
```

```
# Cleaning the environment of the temporary objects
rm(health_stroke_patients,positive_stroke_patients, health_index)
```

With the following code we are going to split again but with this adjusted dataset 80% of records in the training set and 20% in the test set:

```
# Set seed to 5 to have always the same results and it generate
# and already review that there is a  balance betweeen the Stroke patients of
# the train and test set
set.seed(5, sample.kind = "Rounding")
#We need to first create a partition of the dataset for training 80% and 20% testing
test_index <- createDataPartition(stroke_data_num_adj$stroke, times = 1, p = 0.2, list = FALSE)
# Generate the sets for the models:
# For convenience and more clarity we will split in "x" the predictors and
# "y" the actual value of stroke or not stoke
test_x <- stroke_data_num_adj[test_index,1:9]
test_y <- stroke_data_num_adj[test_index,10]
train_x <- stroke_data_num_adj[-test_index,1:9]
train_y <- stroke_data_num_adj[-test_index,10]

# Cleaning the environment
rm(test_index, stroke_data_num_adj)
```

Applying the same logic we used before to adjust the output of the models to categorical values with thte following code and verifying that we have a reasonable balance between both set of the stroke patients as well as know the amount of stroke patients in each set:

```
train_y <- ifelse(train_y==1,"S","H")
test_y  <- ifelse(test_y==1,"S","H")
# Calculate the % of stoke patients in the train set
sum(train_y=="S")/length(train_y)
```

```
## [1] 0.4585253
```

```
# How many stroke patients in Training set
sum(train_y=="S")
```

```
## [1] 199
```

```
# Calculate the % of stoke patients in the test set
sum(test_y=="S")/length(test_y)
```

```
## [1] 0.4587156
```

```
# How many stroke patients in Test set
sum(test_y=="S")
```

```
## [1] 50
```

Looks a good balance between both sets, almost the same percentage of stroke patients so we will now perform the models again:

MODELS EXECUTED WITH ADJUSTED SET

CONCLUSION