Stroke Prediction Project

Bernardo Centeno

2023-08-03

Summary

Stroke is a leading cause of death and disability worldwide (1). Projections indicate that its prevalence will escalate in the coming years, placing considerable stress on healthcare systems and imposing substantial economic implications (2). Hence, the paramount significance of identifying individuals susceptible to stroke, benefiting both the affected persons and the broader community. The primary goal of this project was to construct a machine learning model capable of discerning individuals who have experienced a stroke based on diverse features. To accomplish this, the project employed the publicly accessible dataset titled 'Stroke Prediction Dataset,' accessible at https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset?resource=download. This dataset encompasses a total of 5110 instances and 12 variables, namely: id, gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg_glucose_level, bmi (body mass index), smoking_status, and Stroke. To fulfill the outlined objective, the project encompassed the subsequent stages: 1) Data exploration, cleaning, and visualization, 2) Division of the data into subsets, 3) Training and evaluation of machine learning models, and 4) Model validation. The ultimate model, based on the random forest algorithm, achieved an accuracy level of 95%.

Method

1) Data exploration, cleaning, and visualization

We first started by loading the necessary libraries and the dataset:

```
library(readxl)
library(tidyverse)
```

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

```
## v ggplot2 3.4.0     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

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

```
library(dplyr)
library(caret)
```

```
## Warning: package ’caret’ was built under R version 4.2.2

## Loading required package: lattice

##
## Attaching package: ’caret’

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

```
urlfile <- 'https://raw.githubusercontent.com/bcenteno76/StrokeProject/main/strokedata.csv'
stroke_df <- read_csv(url(urlfile))
```

```
## Rows: 5110 Columns: 12

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): gender, ever_married, work_type, Residence_type, bmi, smoking_status
## dbl (6): id, age, hypertension, heart_disease, avg_glucose_level, stroke
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

Then, we explored the structure of the dataset, and checked for the presence of missing values (NAs) and/or duplicates:

```
str(stroke_df)
```

```
## spc_tbl_ [5,110 x 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ id               : num [1:5110] 9046 51676 31112 60182 1665 ...
##  $ gender           : chr [1:5110] "Male" "Female" "Male" "Female" ...
##  $ age              : num [1:5110] 67 61 80 49 79 81 74 69 59 78 ...
##  $ hypertension     : num [1:5110] 0 0 0 0 1 0 1 0 0 0 ...
##  $ heart_disease    : num [1:5110] 1 0 1 0 0 0 1 0 0 0 ...
##  $ ever_married     : chr [1:5110] "Yes" "Yes" "Yes" "Yes" ...
##  $ work_type        : chr [1:5110] "Private" "Self-employed" "Private" "Private" ...
##  $ Residence_type   : chr [1:5110] "Urban" "Rural" "Rural" "Urban" ...
##  $ avg_glucose_level: num [1:5110] 229 202 106 171 174 ...
##  $ bmi              : chr [1:5110] "36.6" "N/A" "32.5" "34.4" ...
##  $ smoking_status   : chr [1:5110] "formerly smoked" "never smoked" "never smoked" "smokes" ...
##  $ stroke           : num [1:5110] 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   id = col_double(),
##   ..   gender = col_character(),
##   ..   age = col_double(),
##   ..   hypertension = col_double(),
##   ..   heart_disease = col_double(),
##   ..   ever_married = col_character(),
##   ..   work_type = col_character(),
##   ..   Residence_type = col_character(),
##   ..   avg_glucose_level = col_double(),
##   ..   bmi = col_character(),
##   ..   smoking_status = col_character(),
##   ..   stroke = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
any(is.na(stroke_df))
```

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

```
any(duplicated(stroke_df))
```

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

The dataset was composed of 5110 observations and 12 variables,without any NAs or duplicates. The variables were: id, gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg_glucose_level, bmi (body mass index), smoking_status, and Stroke. As the variable "id" was not a predictor, it was removed from the dataset. We noticed that the class of the variables gender, hypertension, heart_disease, ever_married, work_type, Residence_type, bmi, smoking_status and stroke were either 'Character' or 'Interger', when in fact, they should be all 'Factor' variables, with the exception of 'bmi' which should be 'numeric'. Therefore, the necessary adjustments to the original data frame were implemented:

```
stroke_df <- stroke_df %>% select(- id)%>% mutate(stroke = as.factor(stroke),
                          bmi = as.numeric(bmi),
                          gender = as.factor(gender),
                          ever_married = as.factor(ever_married),
                          smoking_status = as.factor(smoking_status),
                          hypertension = as.factor(hypertension),
                          heart_disease = as.factor(heart_disease),
                          work_type = as.factor(work_type),
                          Residence_type = as.factor(Residence_type))
```

```
## Warning in mask$eval_all_mutate(quo): NAs introducidos por coerción
```

When 'bmi' was parsed from 'character' to 'numeric', NAs were introduced by coercion, meaning that some values were not numbers; so, we decided to imput those NAs. We used the Predictive Mean Matching (PMM) method for handling missing values from the mice package. The argument 'm = 5' indicates that five imputed datasets will be generated

```
library(mice)
```

```
## Warning: package 'mice' was built under R version 4.2.3
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

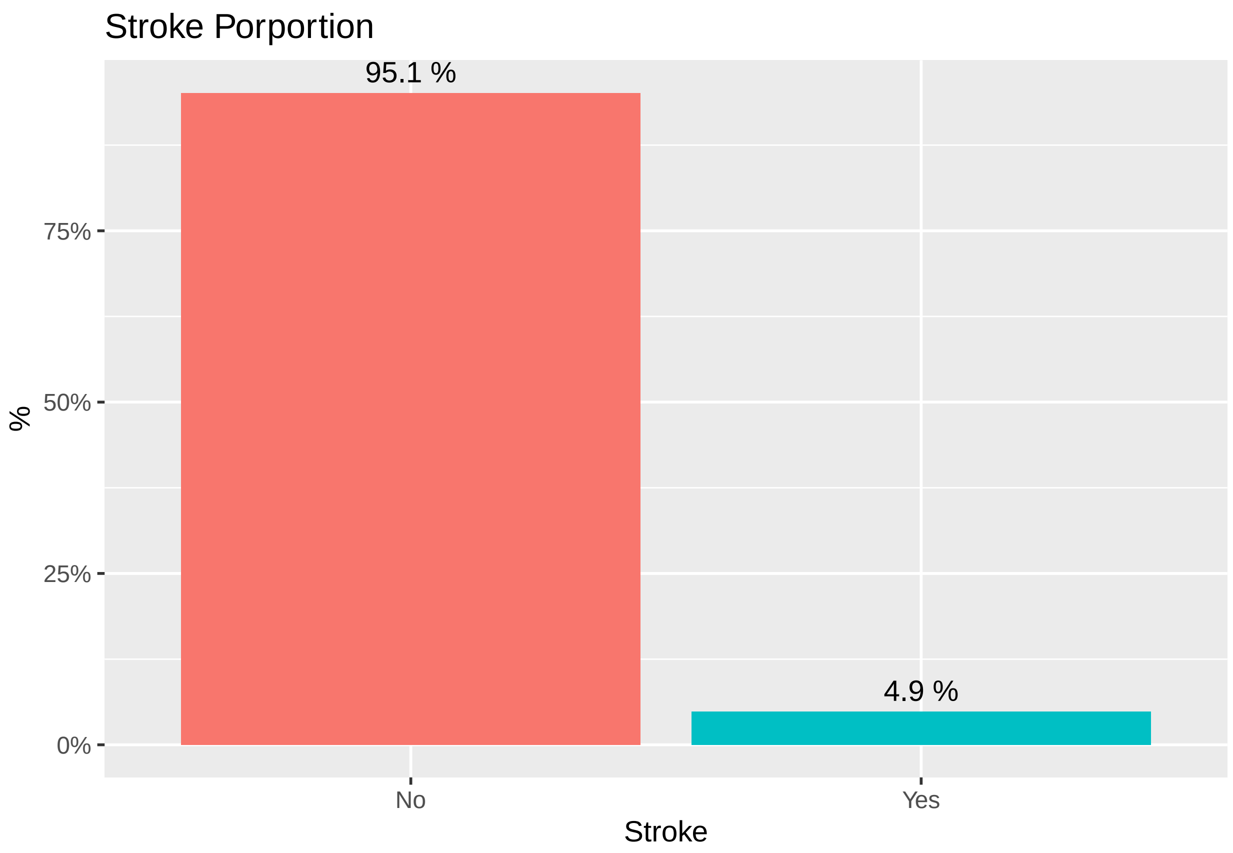
```
datos_imputados <- mice(stroke_df, method = "pmm", m = 5, seed = 123)
```

```
##
##  iter imp variable
##   1   1  bmi
##   1   2  bmi
##   1   3  bmi
##   1   4  bmi
##   1   5  bmi
##   2   1  bmi
##   2   2  bmi
##   2   3  bmi
##   2   4  bmi
##   2   5  bmi
##   3   1  bmi
##   3   2  bmi
##   3   3  bmi
##   3   4  bmi
##   3   5  bmi
##   4   1  bmi
##   4   2  bmi
##   4   3  bmi
##   4   4  bmi
##   4   5  bmi
##   5   1  bmi
##   5   2  bmi
##   5   3  bmi
##   5   4  bmi
##   5   5  bmi
```

```
stroke_df <- complete(datos_imputados)
```

After adjusting the dataset, we studied the outcome (stroke) and each predictor:

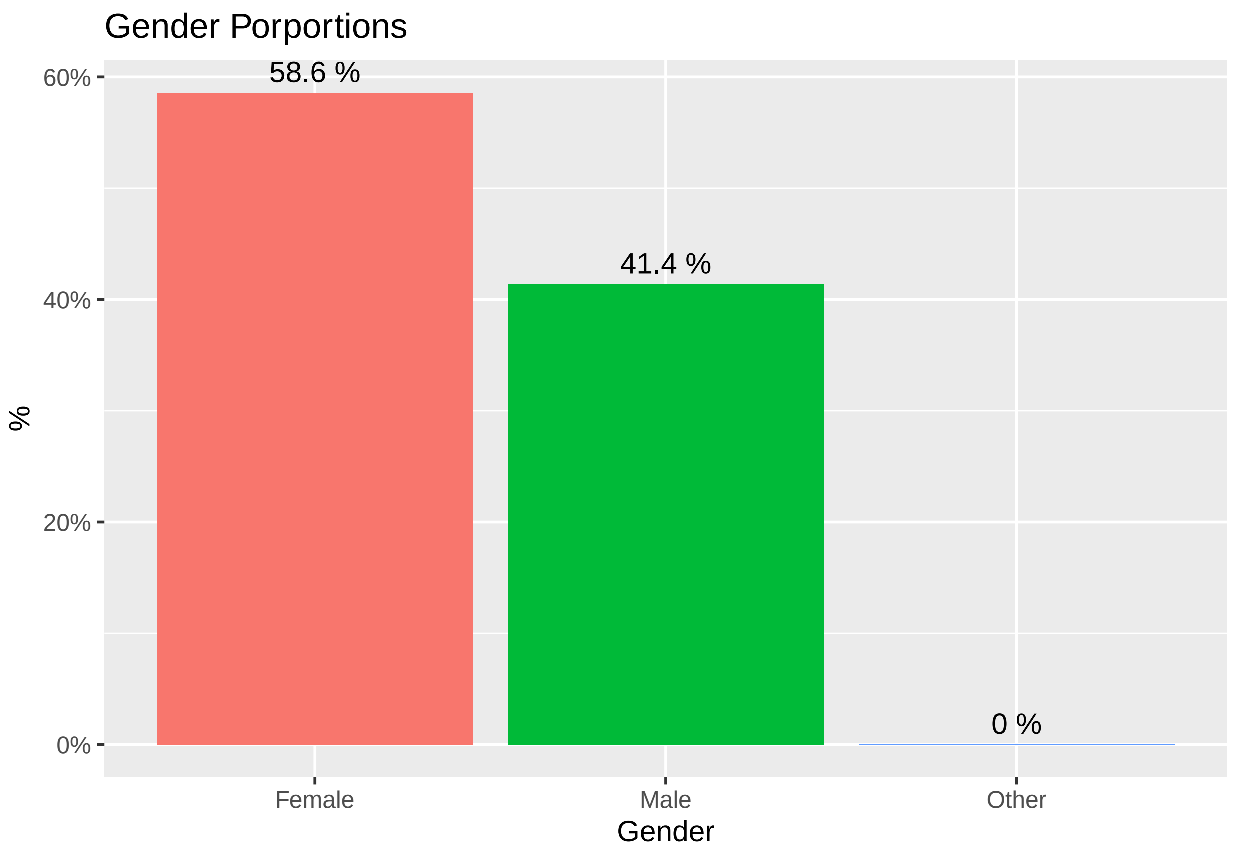
a) Stroke



The prevalence of stroke is 4.9%, which results in an imbalanced dataset. This imbalance can interfere with the performance of machine learning algorithms. For this reason, oversampling will be performed later as a method to address this issue.

b) Gender

Gender was a categorical (factor) variable. 59% of the sample was female, 41% male and 0% other.



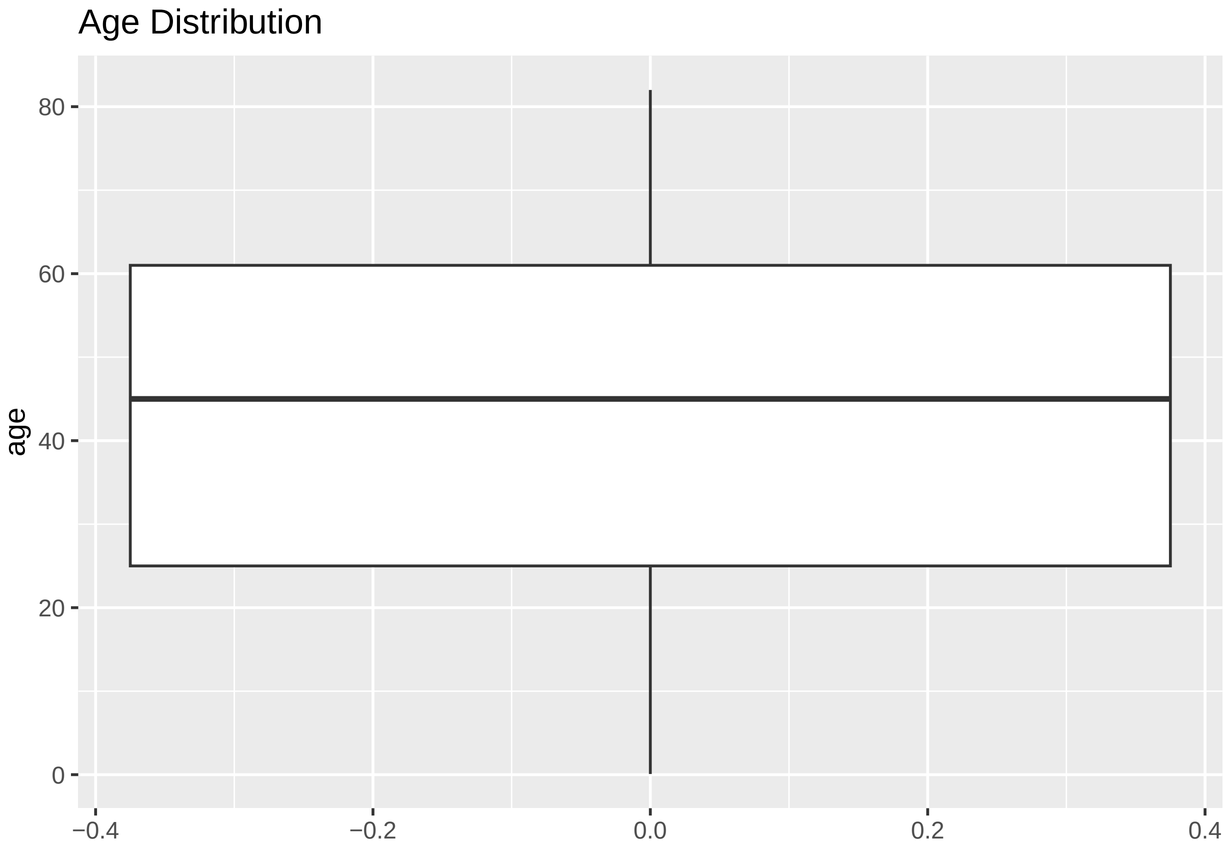
The proportion of strokes in each stratum of the gender variable was:

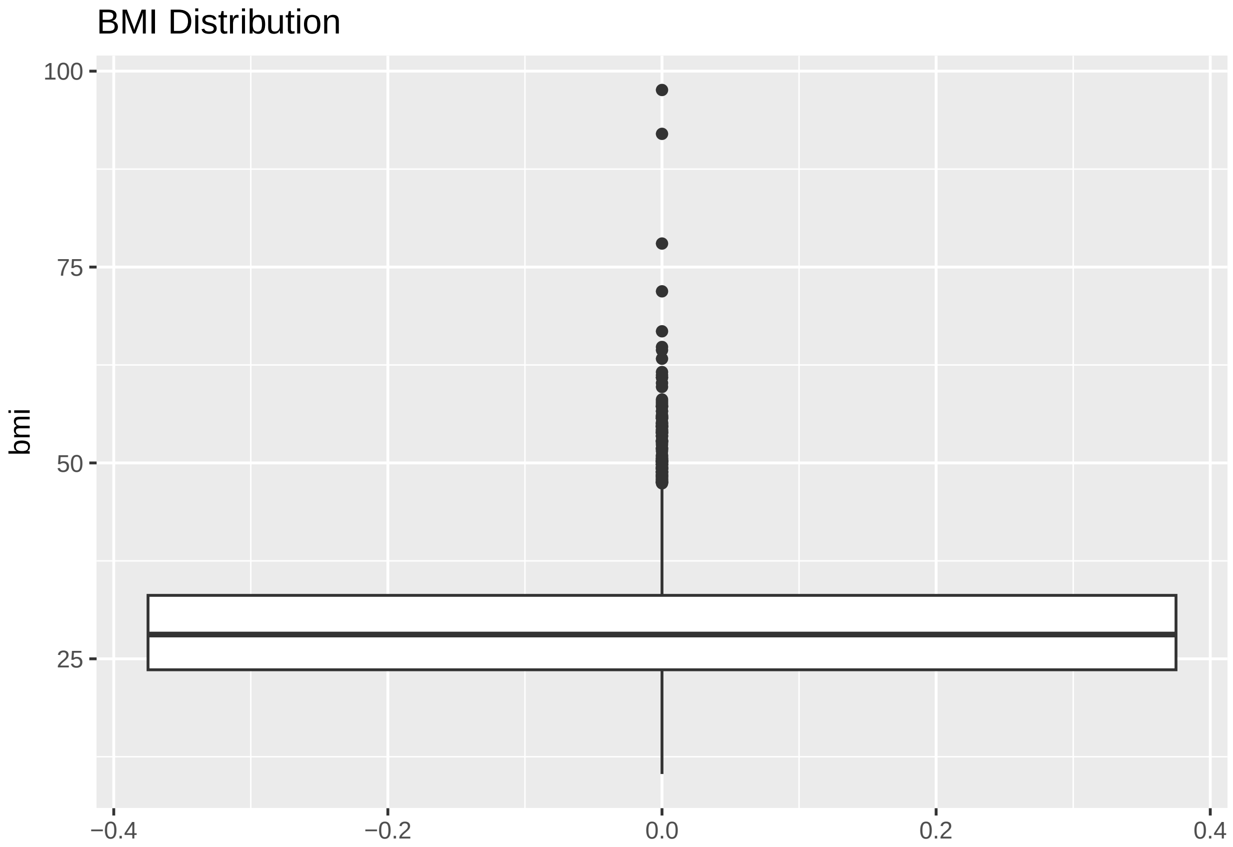
```
##       gender
## stroke Female  Male Other
##    no    95.3  94.9 100.0
##    yes    4.7   5.1   0.0
```

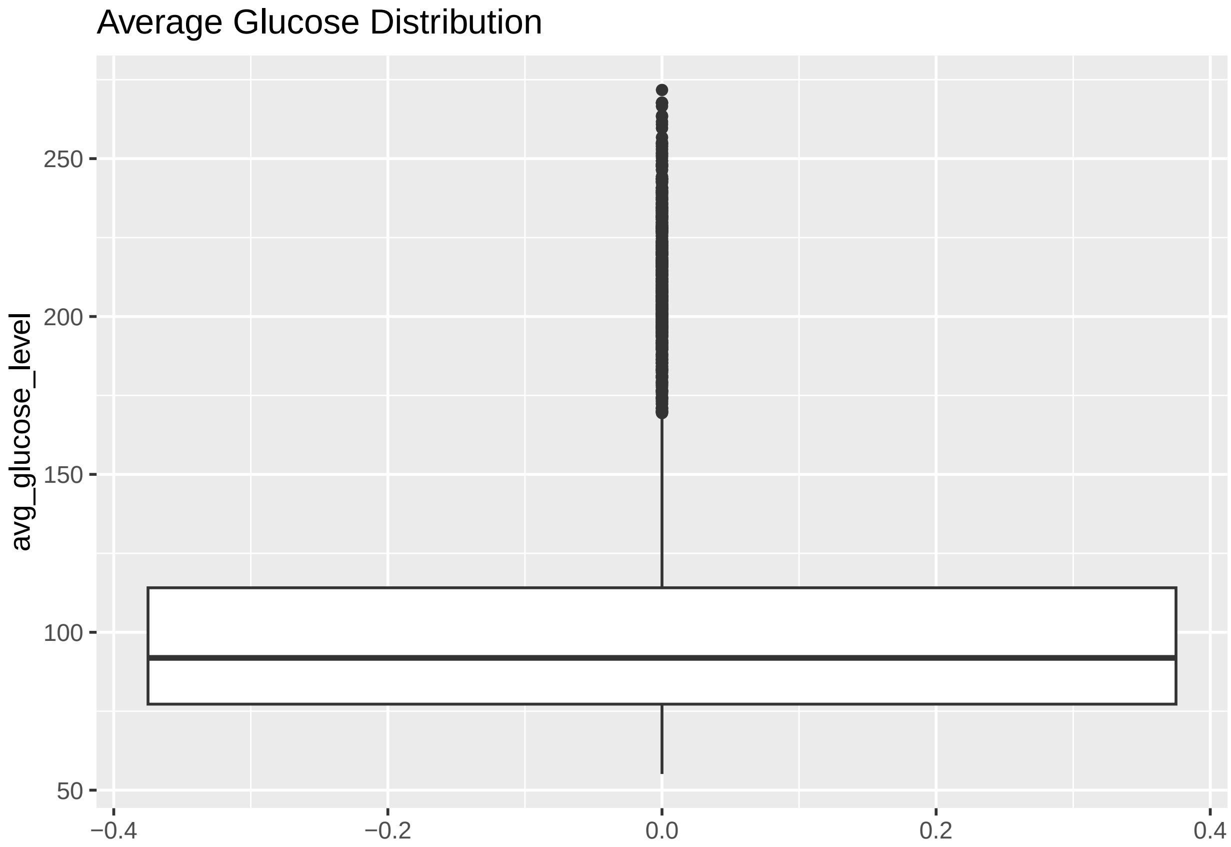
c) Age, bmi and average glucose level:

First we explored this continuous variables and made boxplots for studying their distribution

```
##       age           bmi      avg_glucose_level
##  Min.   : 0   Min.   :10   Min.   : 55
##  1st Qu.:25   1st Qu.:24   1st Qu.: 77
##  Median :45   Median :28   Median : 92
##  Mean   :43   Mean   :29   Mean   :106
##  3rd Qu.:61   3rd Qu.:33   3rd Qu.:114
##  Max.   :82   Max.   :98   Max.   :272
```

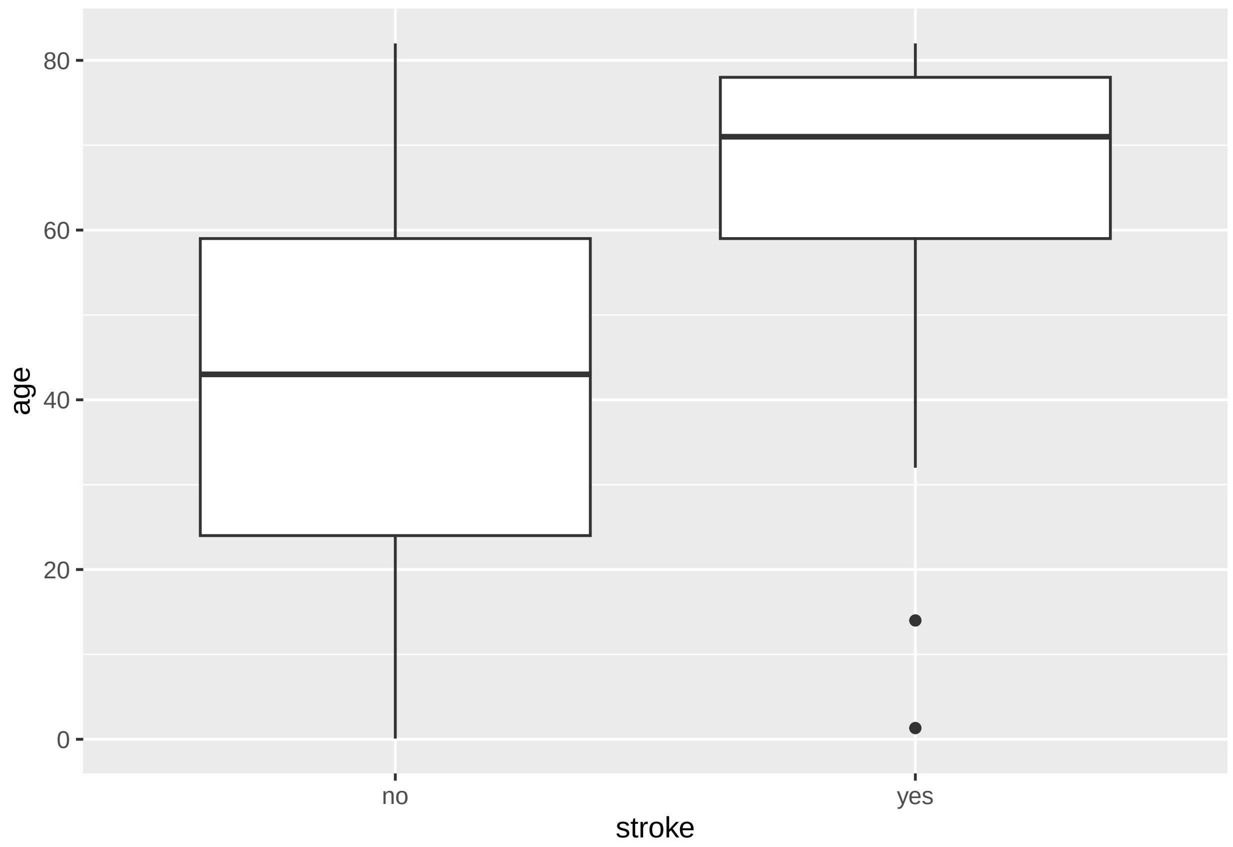

Age Distribution
age
80
60
40
20
0
−0.4
−0.2
0.0
0.2
0.4

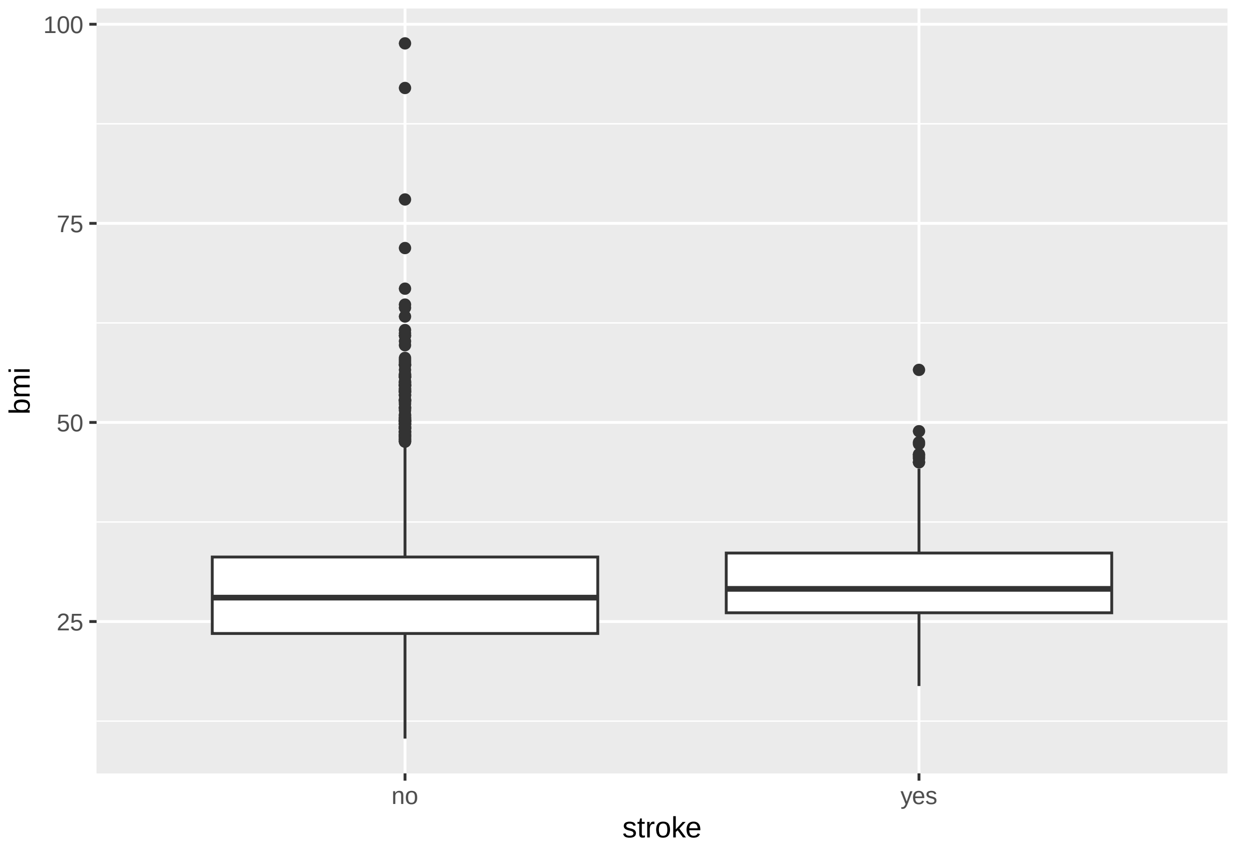

BMI Distribution
bmi
100
75
50
25
−0.4
−0.2
0.0
0.2
0.4

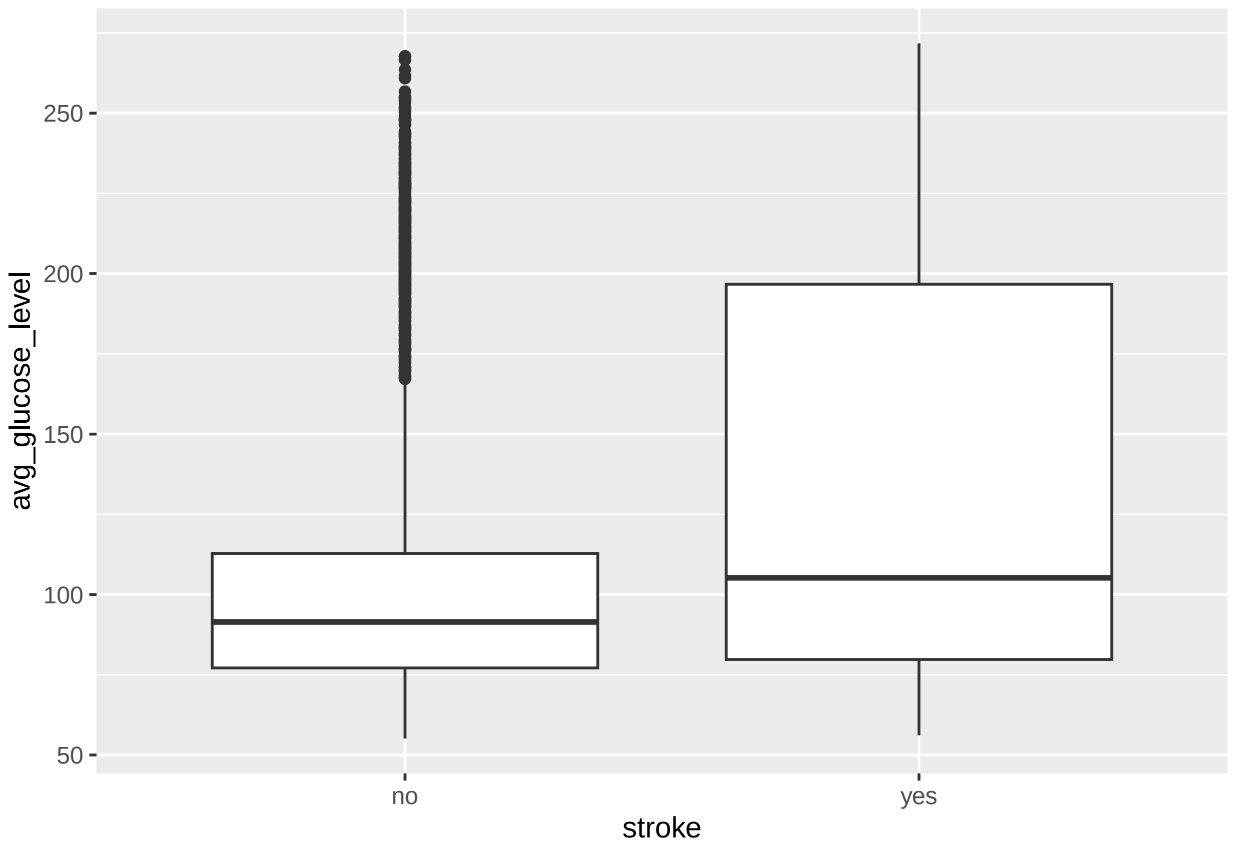


While there were statistical outliers in the variables bmi and avg_glucose_level, we chose not to remove them as they are biologically plausible despite being extreme values (3-4).

Then, we compared the distribution of each variable in patients with and without stroke


80
60
40
20
0
age
no
yes
stroke


100
75
50
25
bmi
no
yes
stroke



As evident from the analysis, individuals who experienced a stroke exhibited higher medians in terms of age, body mass index (bmi), and average glucose levels. This observation aligns with the literature, which has consistently demonstrated the association between these predictive factors and the incidence of stroke (5).

d) hypertension, heart_disease, smoking_status, work_Type, ever_married and Residence_Type

We explored the different proportions of stroke in each category of this predictors

```
## # A tibble: 2 x 2
##   Hypertension ‘Stroke %‘
##   <chr>             <dbl>
## 1 yes               13.3
## 2 no                 3.97
```

```
## # A tibble: 2 x 2
##   ‘Heart Disease‘ ‘Stroke %‘
##   <chr>                <dbl>
## 1 yes                  17.0
## 2 no                    4.18
```

```
## # A tibble: 4 x 2
##   ‘Smoking Status‘ ‘Stroke %‘
##   <fct>                 <dbl>
## 1 formerly smoked        7.91
## 2 smokes                 5.32
## 3 never smoked           4.76
## 4 Unknown                3.04
## # A tibble: 5 x 2
##   ‘Work Type‘   ‘Stroke %‘
##   <fct>              <dbl>
## 1 Self-employed      7.94
## 2 Private            5.09
## 3 Govt_job           5.02
## 4 children           0.291
## 5 Never_worked       0
```

```
## # A tibble: 2 x 2
##   ‘Ever Married‘ ‘Stroke %‘
##   <fct>               <dbl>
## 1 Yes                  6.56
## 2 No                   1.65
```

```
## # A tibble: 2 x 2
##   ‘Residence Type‘ ‘Stroke %‘
##   <fct>                 <dbl>
## 1 Urban                  5.20
## 2 Rural                  4.53
```

It is noticeable that the proportions of stroke are higher in individuals with hypertension, heart disease, current or past smoking history, married status, urban residence, and/or self-employment. These results are consistent with the literature regarding the variables of hypertension, heart disease, and smoking status (5). However, the evidence is inconclusive regarding the variables of married status, urban residence, and self-employment.

2) Division of the data into subsets

To have a set for training the model and a set to test it, the stroke_df dataset was partitioned into a train_set and a test_set respectively, using the createDataPartition() function from the caret package :

```r
set.seed(1)
test_index <- createDataPartition(stroke_df$stroke,times = 1, p = 0.2, list = FALSE)
train_set <- stroke_df[-test_index,]
test_set <- stroke_df[test_index,]
```

As derived from the code above, 80% of the observations were randomly assigned to train_set and 20% to test_set. This percentage distribution is very common among data science projects.

Then, to address the issue of imbalance data (explained above) by means of oversampling, we utilized the upSample() function (caret package):

```r
train_set <- upSample(x = train_set[, -11], y = train_set$stroke, yname = "stroke")
```

3) Training and evaluation of machine learning models

For training different models we used the train() function (caret package), which allows to train different algorithms using similar syntax. It’s important to note that the train() function automatically employs cross-validation, which by default is conducted using 25 bootstrap samples representing 25% of the observations.

We started with a model based on logistic regression

```
fit_glm <- train(stroke~., method= 'glm', data = train_set)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
y_hat <- predict(fit_glm,train_set)
result1 <- confusionMatrix(y_hat,train_set$stroke)$overall["Accuracy"]
result1
```

```
## Accuracy
##     0.78
```

obtaining an accuracy of 78%

Our second model was trained using the k-nearest neighbors (knn) method. This algorithm is used to predict the class of a data point based on its features. It works by calculating the distance between the observations (data points) in the feature space. Given a specific data point, the knn algorithm identifies the k nearest neighbors to that point based on the calculated distances. Subsequently, the algorithm assigns the class of the data point by considering the majority class among its k nearest neighbors. The value of k is a critical parameter in the knn algorithm, with small or large values potentially leading to over-training or over-smoothing respectively.

To set the k parameter, we used the tuneGrid argument, which expects a data frame with the parameter names as specified in the modelLookup output, so we defined a column named k

```
fit_knn <- train(stroke~.,method = 'knn', data = train_set,
                 tuneGrid = data.frame(k = seq(7, 51, 2)))
```

We tested this model on the train_set:

```
y_hat1 <- predict(fit_knn, train_set)
result2 <- confusionMatrix(y_hat1,train_set$stroke)$overall["Accuracy"]
result2
```

```
## Accuracy
##      0.9
```

and we got an accuracy of 90%.

Our third model was constructed using the Random Forest (rf) algorithm, which is employed to predict the class of a given data point based on its associated predictors. Unlike single decision trees prone to overfitting, rf leverages a collection of decision trees, each trained on different subsets of the data and predictors. When making a prediction, each tree in the forest independently classifies the data point, and the final class is determined through a majority vote among all the trees. By introducing randomness in both data sampling and feature selection, Random Forest mitigates overfitting and enhances generalization performance.

```
fit_rf <- train(stroke~.,method = 'rf', data = train_set)
```

We then proceed to test it on the train_set

```
y_hat2 <- predict(fit_rf, train_set)
result3 <- confusionMatrix(y_hat2,train_set$stroke)$overall["Accuracy"]
result3
```

```
## Accuracy
##        1
```

and we got a 100% accuracy.

4) Model Validation

We finally tested our rf-based model on the test_set:

```
y_hat_rf <- predict(fit_rf, test_set)
result4 <- confusionMatrix(y_hat_rf,test_set$stroke)$overall["Accuracy"]
result4
```

```
## Accuracy
##     0.95
```

and got an accuracy of 95%.

Results

Here we share a summary table showing the accuracy obtained for each model tested on the train_set, and for the final model on the test_set:

```
## # A tibble: 4 x 2
##   Method                       Accuracy
##   <chr>                           <dbl>
## 1 glm                             0.780
## 2 knn                             0.899
## 3 rf                              1
## 4 Final model validation (rf)     0.945
```

As shown above, during the validation phase, the accuracy obtained for the final model on the test_set was 95%.

Conclusion

In conclusion, this project aimed to construct a machine learning model capable of accurately identifying individuals at risk of experiencing a stroke based on various features. By utilizing the 'Stroke Prediction Dataset' and employing techniques such as data exploration, cleaning, oversampling, and model training, we achieved a significant breakthrough in stroke prediction accuracy.

However, it's important to acknowledge the limitations of this study. First, while the achieved accuracy is promising, further validation on larger and more diverse datasets is essential to ascertain the model's robustness. Additionally, our model is based on retrospective data, which may not fully capture the complexity of real-time clinical scenarios.

Furthermore, the dataset's imbalanced nature, with a higher proportion of non-stroke cases, could have introduced bias and affected the generalization of the model. Although oversampling was performed to mitigate this issue, the potential for overfitting and introducing noise should be considered.

In real-world application, the performance of the model would depend on the quality of input data and the prevalence of stroke in the population being studied. Clinical factors not included in the dataset could also impact the model's predictive ability.

In conclusion, while the achieved accuracy is promising, further research, refinement, and validation are necessary to ensure the model's clinical utility and generalizability.

References

1) Saini, V., Guada, L., & Yavagal, D. R. (2021). Global epidemiology of stroke and access to acute ischemic stroke interventions. Neurology, 97(20 Supplement 2), S6-S16.

2) Wafa, H. A., Wolfe, C. D., Emmett, E., Roth, G. A., Johnson, C. O., & Wang, Y. (2020). Burden of stroke in Europe: thirty-year projections of incidence, prevalence, deaths, and disability-adjusted life years. Stroke, 51(8), 2418-2427.

3) Penman, A. D., & Johnson, W. D. (2006). The changing shape of the body mass index distribution curve in the population: implications for public health policy to reduce the prevalence of adult obesity. Preventing chronic disease, 3(3).

4) Huang, W., Xu, W., Zhu, P., Yang, H., Su, L., Tang, H., & Liu, Y. (2017). Analysis of blood glucose distribution characteristics in a health examination population in Chengdu (2007–2015). Medicine, 96(49).

5) Feigin, V. L., Stark, B. A., Johnson, C. O., Roth, G. A., Bisignano, C., Abady, G. G., … & Hamidi, S. (2021). Global, regional, and national burden of stroke and its risk factors, 1990–2019: a systematic analysis for the Global Burden of Disease Study 2019. The Lancet Neurology, 20(10), 795-820.