

# **HarvardX PH125.9x Data Science: Capstone Stroke Prediction System**

*Robert Walscheid*

*July 28, 2021*

# Contents

<b>1. INTRODUCTION</b>	<b>3</b>
<b>2. PREREQUISITES</b>	<b>4</b>
2.1 Required Packages . . . . .	4
2.2 Required Data . . . . .	5
<b>3. STROKE-DATA.CSV DATA FILE</b>	<b>6</b>
<b>4. DATA PREPARATION &amp; ANALYSIS</b>	<b>8</b>
4.1 rawdata Dataset . . . . .	8
4.2 stroke_data Dataset . . . . .	9
4.2.1 stroke_data_num Dataset . . . . .	13
4.3 stroke_train and stroke_test Modeling Datasets . . . . .	14
4.4 Dataset Analysis . . . . .	16
<b>5. MODELING</b>	<b>23</b>
Model 1: Logistic Regression (GLM) . . . . .	23
Model 2: Naive Bayes . . . . .	24
Model 3: Linear Discriminant Analysis (LDA) . . . . .	25
Model 4: Classification and Regression Trees Model (CART) . . . . .	26
Model 5: Random Forest . . . . .	28
Model 6: k-Nearest Neighbor (kNN) . . . . .	29
<b>RESULTS</b>	<b>32</b>
<b>CONCLUSION</b>	<b>32</b>
<b>REFERENCES</b>	<b>33</b>

# 1. INTRODUCTION

In a healthy individual, our veins and arteries carry blood and nutrients throughout our body, including to our brain. A stroke is a medical emergency that occurs when this blood supply to the brain is interrupted or stops moving through the vascular system to the brain. According to the National Institute of Neurological Disorders and Stroke, the United States sees more than 795,000 stroke cases annually, which is the fifth leading cause of death in the country.<sup>1</sup> The CDC stated that in 2018, 1 in every 6 deaths from cardiovascular disease was due to stroke. Furthermore, every 40 seconds, someone in the United States has a stroke, and every 4 minutes, someone dies of stroke.<sup>2</sup>

The two main types of stroke are **ischemic**, where blood clots or fatty deposits called plaque block the blood vessels, and **hemorrhagic**, where blood vessels burst in the brain causing blood to pool up and damage surrounding tissue. Many factors increase the risk of having one of these two types of stroke, including being overweight (or obese), being physically inactive, heavy (or binge) drinking, use of illegal drugs, high blood pressure, smoking, high cholesterol, diabetes, obstructive sleep apnea, cardiovascular disease, family history of stroke or heart attack, and COVID-19 infection.<sup>3</sup>

This project focuses on creating a stroke prediction system using a synthetic dataset<sup>4</sup> uploaded by user Dhiren Dommeti to the well-known public code & data repository website, Kaggle. This is to fulfill the the “IDV Learners” portion of the **HarvardX: PH125.9x Data Science: Capstone** course.

This project report will step through the process of programmatically obtaining and analyzing the stroke data file, creating and analyzing the necessary datasets, building and testing multiple machine learning algorithms, and will conclude with commenting on the final model and its results. Using supervised machine learning methodologies, coded in the R programming language<sup>5</sup>, the following algorithms will be used to predict the likelihood of a stroke based on variables in the dataset:

1. Generalized Linear Model (GLM)
2. Naive Bayes
3. Linear Discriminant Analysis (LDA)
4. Classification and Regression Trees (CART)
5. Random Forest
6. k-Nearest Neighbor (KNN)

The resultant highest accuracy score was found with the Random Forest model at 0.998.

---

<sup>1</sup><https://www.stroke.nih.gov/materials/needtoknow.htm>

<sup>2</sup><https://www.cdc.gov/stroke/facts.htm>

<sup>3</sup><https://www.mayoclinic.org/diseases-conditions/stroke/symptoms-causes/syc-20350113>

<sup>4</sup><https://www.kaggle.com/dhirendommeti/stroke>

<sup>5</sup><https://www.r-project.org/>

## 2. PREREQUISITES

This project and its code has minimum requirements in order to duplicate the environment necessary to successfully run in R Studio. It will be assumed that all required R packages and applications are current as of the date of this report (found on the cover page), and that the computer has Internet access.

### Current RStudio Version Output:

```
##  
## platform      x86_64-w64-mingw32  
## arch          x86_64  
## os            mingw32  
## system        x86_64, mingw32  
## status  
## major         4  
## minor         1.0  
## year          2021  
## month         05  
## day           18  
## svn rev       80317  
## language      R  
## version.string R version 4.1.0 (2021-05-18)  
## nickname      Camp Pontanezen
```

### 2.1 Required Packages

In order for the R code to run properly, the below packages are required. In the event that they are not available at runtime, the R code will download, install, and load them first.

### Package Loading Code Snippet:

```
# Download and install the necessary packages for this project.  
if(!require(tidyverse)) +  
  install.packages("tidyverse", repos = "http://cran.us.r-project.org")  
if(!require(caret)) +  
  install.packages("caret", repos = "http://cran.us.r-project.org")  
if(!require(data.table)) +  
  install.packages("data.table", repos = "http://cran.us.r-project.org")  
if(!require(scales)) +  
  install.packages("scales", repos = "http://cran.us.r-project.org")  
if(!require(corrplot)) +  
  install.packages("corrplot", repos = "http://cran.us.r-project.org")  
if(!require(dplyr)) +  
  install.packages("dplyr", repos = "http://cran.us.r-project.org")  
if(!require(grid)) +  
  install.packages("gridExtra", repos = "http://cran.us.r-project.org")  
if(!require(grid)) +  
  install.packages("gridExtra", repos = "http://cran.us.r-project.org")
```

```

if(!require(kableExtra)) +
  install.packages("kableExtra", repos = "http://cran.us.r-project.org")
if(!require(stats))
  install.packages("stats", repos = "http://cran.us.r-project.org")

# Load the necessary libraries for this project.
library(tidyverse)
library(caret)
library(data.table)
library(scales)
library(corrplot)
library(dplyr)
library(grid)
library(gridExtra)
library(kableExtra)
library(stats)

```

## 2.2 Required Data

The stroke dataset used for this project is originally from the Kaggle website. Since Kaggle requires each user to log into their site before being able to download any data files, the 314KB **Stroke-Data.csv** file was previously downloaded and pushed to the GitHub repository this RStudio project was created in.

### Dataset Download Code Snippet:

```

# Download the Stroke-Data.csv file from GitHub, create the data frame "rawdata",
# and fill it with the 12 columns of data that were in the file, labeling them
# id, gender, age, hypertension, heart_disease, ever_married, work_type,
# Residence_type, bmi, avg_glucose_level, smoking_status, and stroke,
# respectively.
rawdata <- read.csv("https://raw.githubusercontent.com/rwalscheid/HarvardX-Capstone-CYO/main/Stroke-Data.csv",
  quote = "",
  row.names = NULL,
  stringsAsFactors = FALSE)

```

The .csv file will automatically be downloaded to your computer's temporary directory and read into a data frame, which contains the necessary records for this project.

### 3. STROKE-DATA.CSV DATA FILE

With the required R packages loaded and the `Stroke-Data.csv` data file downloaded and ingested into a data frame, reviewing this file within an appropriate editor (e.g. Microsoft Excel, OpenOffice Calc, Google Drive, etc.) would be beneficial in understanding its full contents for this project. Only a few of the predictors found in this dataset are directly-related health-based factors that impact a person's risk of stroke, while others are surrounding environmental factors.

1. **id** (class=integer): A unique identification number for each person.
2. **gender** (class=character): The person's gender, as male or female.
3. **age** (class=numeric): The person's age in years.
4. **hypertension** (class=integer): Yes/No (as 1/0) entry for whether the person has hypertension.
5. **heart\_disease** (class=integer): Yes/No (as 1/0) entry for whether the person has heart disease.
6. **ever\_married** (class=character): Yes/No entry for whether the person has ever been married.
7. **work\_type** (class=character): Category of work the person is in (or whether they are a child).
8. **Residence\_type** (class=character): Rural/Urban entry for where the person lives.
9. **bmi** (class=character): The person's body mass index (BMI).
10. **avg\_glucose\_level** (class=numeric): The person's average blood sugar level.
11. **smoking\_status** (class=character): Category for whether the person is/was a smoker.
12. **stroke** (class=integer): Yes/No (as 1/0) entry for whether the person has had a stroke.

A basic analysis of this dataset once created is shown below.

rawdata Object Summary:

```
# Summary information of the "rawdata" dataset
summary(rawdata)
```

```
##           id           gender           age           hypertension
## Min.      :   67  Length:5110  Min.      : 0.08  Min.      :0.00000
## 1st Qu.:11980  Class :character  1st Qu.:25.00  1st Qu.:0.00000
## Median :32702  Mode  :character  Median :45.00  Median :0.00000
## Mean    :33831                Mean    :43.23  Mean    :0.09746
## 3rd Qu.:52644                3rd Qu.:61.00  3rd Qu.:0.00000
## Max.     :72940                Max.     :82.00  Max.     :1.00000
## heart_disease  ever_married      work_type      Residence_type
## Min.          :0.00000  Length:5110  Length:5110  Length:5110
## 1st Qu.:0.00000  Class :character  Class :character  Class :character
## Median :0.00000  Mode  :character  Mode  :character  Mode  :character
## Mean        :0.05401
## 3rd Qu.:0.00000
## Max.        :1.00000
##           bmi      avg_glucose_level  smoking_status      stroke
## Length:5110  Min.      : 55.12  Length:5110  Min.      :0.00000
## Class :character  1st Qu.: 77.25  Class :character  1st Qu.:0.00000
## Mode  :character  Median : 91.89  Mode  :character  Median :0.00000
##                      Mean    :106.15                Mean    :0.04873
##                      3rd Qu.:114.09                3rd Qu.:0.00000
##                      Max.     :271.74                Max.     :1.00000
```

## rawdata Data Structure:

```
# Structure of the "rawdata" dataset
```

```
str(rawdata)
```

```
## 'data.frame':    5110 obs. of  12 variables:
## $ id            : int  9046 9047 9048 9049 9050 9051 9052 9053 9054 9055 ...
## $ 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" ...
## $ bmi           : chr   "36.6" "N/A" "32.5" "34.4" ...
## $ avg_glucose_level: num   229 202 106 171 174 ...
## $ smoking_status : chr   "formerly smoked" "never smoked" "never smoked" "smokes" ...
## $ stroke        : int    1 1 1 1 1 1 1 1 1 1 ...
```

## 4. DATA PREPARATION & ANALYSIS

### 4.1 rawdata Dataset

With the rawdata dataset now created, the previous basic analysis showed the data frame consists of 5,110 rows and 12 columns (id, gender, age, hypertension, heart\_disease, ever\_married, work\_type, Residence\_type, bmi, avg\_glucose\_level, smoking\_status, and stroke).

Each variable with multiple categories will need to be reviewed, as they will be changed to factors with levels (the variable's categories) for easier plotting and analysis throughout this project report. This will also help determine upfront as to whether there are outliers that may be discarded to minimize confounding.

```
# View the multiple categories for each applicable variable to determine factor levels  
# to convert to.
```

```
unique(rawdata$gender)
```

```
## [1] "Male" "Female" "Other"
```

```
unique(rawdata$ever_married)
```

```
## [1] "Yes" "No"
```

```
unique(rawdata$work_type)
```

```
## [1] "Private" "Self-employed" "Govt_job" "children"  
## [5] "Never_worked"
```

```
unique(rawdata$Residence_type)
```

```
## [1] "Urban" "Rural"
```

```
unique(rawdata$smoking_status)
```

```
## [1] "formerly smoked" "never smoked" "smokes" "Unknown"
```

### Variable Conversion to Factors with Levels

Out of the 5,110 rows in the dataset, there is a single non-female/male entry in the gender category labeled Other.

```
# View the gender count for the category "Other".  
rawdata %>% filter(gender %in% "Other") %>% group_by(gender) %>% summarize(n=n())
```

```
## # A tibble: 1 x 2  
##   gender      n  
##   <chr> <int>  
## 1 Other      1
```



The count of Other entries above shows only one person listed. This single entry will be removed during the training/testing data split for simplicity since 1 entry out of 5,110 in this particular dataset is not considered statistically significant. This will leave only two levels for the gender factor when converted: Male and Female.

To make plotting charts and performing summary analysis throughout this report easier to read and identify, the variables below will be converted to factors, then relabeled and re-ordered (if necessary) as follows:

Variables	Factor Levels
<b>gender</b>	Male
	Female
<b>hypertension</b>	wo_hypertension = 0
	w_hypertension = 1
<b>heart_disease</b>	wo_heart_disease = 0
	w_heart_disease = 1
<b>ever_married</b>	Yes
	No
<b>work_type</b>	Underage_child = children
	Never_worked
	Self-employed
	Private_company = Private
<b>Residence_type</b>	Govt_job
	Rural
<b>smoking_status</b>	Urban
	never smoked
	formerly smoked
	smokes
<b>stroke</b>	Unknown
	no_stroke = 0
	stroke = 1

When examining the structure of the rawdata data frame in Section 3, it was noted that the bmi vector is of class character. Since the body mass index (BMI) is numeric, this variable will need to be converted.

## 4.2 stroke\_data Dataset

With the variable categories laid out and the factor level order, labels, and levels determined, a new dataset named stroke\_data will be created with all of the changes listed in Section 4.2 above. This new data frame will contain 5,109 rows with 12 columns.

```
# Remove the single "Other" gender row to not skew results, convert the BMI to numeric,
# and convert "gender", "hypertension", "heart_disease", "ever_married", "work_type",
# "Residence_type", "smoking_status", and "stroke" all to factors with their respective
# levels.
stroke_data <- rawdata %>% filter(gender!="Other") %>%
  mutate(gender = factor(gender, levels = c("Male", "Female")),
         hypertension = factor(hypertension,
```

```

        labels = c("wo_hypertension", "w_hypertension"),
        levels = c("0", "1")),
heart_disease = factor(heart_disease,
        labels = c("wo_heart_disease", "w_heart_disease"),
        levels = c("0", "1")),
ever_married = factor(ever_married,
        labels = c("never_married", "is_or_was_married"),
        levels = c("No", "Yes")),
work_type = factor(work_type,
        labels = c("Underage_child",
                    "Never_worked",
                    "Self-employed",
                    "Private_company",
                    "Govt_job"),
        levels = c("children",
                    "Never_worked",
                    "Self-employed",
                    "Private",
                    "Govt_job")),
Residence_type = factor(Residence_type,
        levels = c("Rural", "Urban")),
bmi = as.numeric(bmi),
smoking_status = factor(smoking_status,
        levels = c("never smoked",
                    "formerly smoked",
                    "smokes",
                    "Unknown")),
stroke = factor(stroke,
        labels = c("no_stroke", "stroke"),
        levels = c("0", "1"))

```

A basic analysis of this dataset once created is shown below.

### stroke\_data Object Summary:

```

# Summary information of the "stroke_data" dataset
summary(stroke_data)

```

```

##          id          gender          age          hypertension
##  Min.   : 67   Male :2115   Min.   : 0.08   wo_hypertension:4611
##  1st Qu.:11973   Female:2994   1st Qu.:25.00   w_hypertension : 498
##  Median :32687                                Median :45.00
##  Mean   :33826                                Mean   :43.23
##  3rd Qu.:52619                                3rd Qu.:61.00
##  Max.   :72940                                Max.   :82.00
##
##          heart_disease          ever_married          work_type
##  wo_heart_disease:4833   never_married :1756   Underage_child : 687

```

```
## w_heart_disease : 276   is_or_was_married:3353   Never_worked   : 22
##                                                    Self-employed   : 819
##                                                    Private_company:2924
##                                                    Govt_job        : 657
##
##
## Residence_type      bmi      avg_glucose_level      smoking_status
## Rural:2513          Min.    :10.30      Min.    : 55.12      never smoked   :1892
## Urban:2596          1st Qu.:23.50      1st Qu.: 77.24      formerly smoked: 884
##                      Median :28.10      Median : 91.88      smokes        : 789
##                      Mean    :28.89      Mean    :106.14      Unknown       :1544
##                      3rd Qu.:33.10      3rd Qu.:114.09
##                      Max.    :97.60      Max.    :271.74
##                      NA's    :201
##
##      stroke
## no_stroke:4860
## stroke   : 249
##
##
##
##
##
```

### stroke\_data Data Structure:

```
# Structure of the "stroke_data" dataset
str(stroke_data)
```

```
## 'data.frame':    5109 obs. of  12 variables:
## $ id           : int  9046 9047 9048 9049 9050 9051 9052 9053 9054 9055 ...
## $ gender       : Factor w/ 2 levels "Male","Female": 1 2 1 2 2 1 1 2 2 2 ...
## $ age          : num  67 61 80 49 79 81 74 69 59 78 ...
## $ hypertension : Factor w/ 2 levels "wo_hypertension",...: 1 1 1 1 2 1 2 1 1 1 ...
## $ heart_disease : Factor w/ 2 levels "wo_heart_disease",...: 2 1 2 1 1 1 2 1 1 1 ...
## $ ever_married  : Factor w/ 2 levels "never_married",...: 2 2 2 2 2 2 2 1 2 2 ...
## $ work_type     : Factor w/ 5 levels "Underage_child",...: 4 3 4 4 3 4 4 4 4 4 ...
## $ Residence_type : Factor w/ 2 levels "Rural","Urban": 2 1 1 2 1 2 1 2 1 2 ...
## $ bmi           : num  36.6 NA 32.5 34.4 24 29 27.4 22.8 NA 24.2 ...
## $ avg_glucose_level: num  229 202 106 171 174 ...
## $ smoking_status : Factor w/ 4 levels "never smoked",...: 2 1 1 3 1 2 1 1 4 4 ...
## $ stroke        : Factor w/ 2 levels "no_stroke","stroke": 2 2 2 2 2 2 2 2 2 2 ...
```

In the rawdata structure output in Section 3, it can be seen that the system generating the Stroke-Data.csv data replaced any null (empty) bmi fields with the characters “N/A”. When the stroke\_data dataset converted those “N/A” character entries to numeric format, NA’s were introduced in their place by coercion.

## bmi “NA” Quantity & Percentage Calculations:

```
# Calculate the number of bmi N/A entries as a percentage of the total.
bmi_na_num <- sum(is.na(stroke_data$bmi))
bmi_na_num
```

```
## [1] 201
```

```
bmi_na_pct_tot <- (bmi_na_num/NROW(stroke_data$bmi))*100
bmi_na_pct_tot
```

```
## [1] 3.934234
```

With the resulting quantity of bmi NA's being 201 (accounting for almost 4% of the entire dataset), the number of entries justifies looking into the statistical impact against the overall outcome. With the outcome being stratified into no\_stroke or stroke results, determining the stroke-negative and stroke-positive percentages these NA's have will show whether they can be excluded from further calculations.

## bmi “NA” Stroke/No-Stroke Percentage Calculations:

```
# Calculate statistical significance the stroke-positive vs. stroke-negative NA's
# would have on the predictability.
bmi_na_pct_stroke <-
  (length(which(is.na(stroke_data$bmi) & stroke_data$stroke == "stroke"))/
    NROW(stroke_data$bmi))*100
bmi_na_pct_stroke #Stroke-Positive Percentage
```

```
## [1] 0.7829321
```

```
bmi_na_pct_nostroke <-
  (length(which(is.na(stroke_data$bmi) & stroke_data$stroke == "no_stroke"))/
    NROW(stroke_data$bmi))*100
bmi_na_pct_nostroke #Stroke-Negative Percentage
```

```
## [1] 3.151302
```

As shown above, the NA entries in the bmi variable have a 0.7829% and 3.1513% impact to the predictability of a stroke vs. no\_stroke result, respectively. If this difference had been closer to a 50%/50% split, we could possibly omit the NA bmi entries since their presence would not impact the overall correlation of bmi value to a stroke/no\_stroke outcome. The NA's instead will be replaced with the mean bmi values for stroke/no\_stroke results, accordingly.

## bmi “NA” Replacement:

```

# Calculate the mean bmi in the dataset for all non-"NA" bmi entries that were also
# stroke-negative.
ns_bmi_mu <- as.numeric(stroke_data %>% filter(!is.na(bmi) & stroke=="no_stroke") %>%
  summarize(bmi_mu=mean(bmi)))

# Calculate the mean bmi in the dataset for all non-"NA" bmi entries that were also
# stroke-positive.
s_bmi_mu <- as.numeric(stroke_data %>% filter(!is.na(bmi) & stroke=="stroke") %>%
  summarize(bmi_mu=mean(bmi)))

# Replace all bmi "NA" entries with the stroke-positive or stroke-negative mean bmi
# values, accordingly.
stroke_data <- stroke_data %>%
  mutate(bmi = case_when(is.na(bmi) & stroke == "stroke" ~ s_bmi_mu,
    is.na(bmi) & stroke == "no_stroke" ~ ns_bmi_mu,
    TRUE ~ bmi))

```

#### 4.2.1 stroke\_data\_num Dataset

While the factor variables in the `stroke_data` dataset allow for easier graphical analysis, correlation coefficient functions like `cor()`, `cor.test()`, and `corrplot()` will be used in later sections for analyzing variable importance. This requires that the factors be converted into numeric format. The `stroke_data` dataset will be copied into a new dataset called `stroke_data_num` with the exception of the `id` column, since that variable is not necessary for this purpose, and all necessary variables will be converted accordingly.

```

# Create the fully-numeric "stroke_data_num" dataset, copying the original
# "stroke_data" dataset and convert all factor variables.
stroke_data_num <- stroke_data %>% mutate(gender=as.numeric(gender),
  hypertension=as.numeric(hypertension),
  heart_disease=as.numeric(heart_disease),
  ever_married=as.numeric(ever_married),
  work_type=as.numeric(work_type),
  Residence_type=as.numeric(Residence_type),
  smoking_status=as.numeric(smoking_status),
  stroke=as.numeric(stroke)) %>%
  select(-id) #Leave out the "id" column as it is not needed for correlation analysis

```

#### stroke\_data\_num Data Structure:

```

# Structure of the "stroke_data_num" dataset
str(stroke_data_num)

## 'data.frame':    5109 obs. of  11 variables:
## $ gender          : num  1 2 1 2 2 1 1 2 2 2 ...
## $ age             : num  67 61 80 49 79 81 74 69 59 78 ...
## $ hypertension    : num  1 1 1 1 2 1 2 1 1 1 ...
## $ heart_disease    : num  2 1 2 1 1 1 2 1 1 1 ...

```

```
## $ ever_married      : num  2 2 2 2 2 2 2 1 2 2 ...
## $ work_type         : num  4 3 4 4 3 4 4 4 4 4 ...
## $ Residence_type    : num  2 1 1 2 1 2 1 2 1 2 ...
## $ bmi               : num  36.6 30.5 32.5 34.4 24 ...
## $ avg_glucose_level: num  229 202 106 171 174 ...
## $ smoking_status     : num  2 1 1 3 1 2 1 1 4 4 ...
## $ stroke            : num  2 2 2 2 2 2 2 2 2 2 ...
```

### 4.3 stroke\_train and stroke\_test Modeling Datasets

Now that the data is ready for model development, the `stroke_data` data frame will be split with a 80%/20% ratio into a training subset called `stroke_train` (4,087 rows) and a test subset called `stroke_test` (1,022 rows), respectively.

#### stroke\_train and stroke\_test Subset Creation Code Snippet:

```
# The training dataset, "stroke_train", and validation test dataset,
# "stroke_test", will be created from an 80%/20% split of the "stroke_data"
# data frame, respectively.
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = stroke_data$stroke, times = 1, p = 0.2, list = FALSE)
stroke_train <- stroke_data[-test_index,]
stroke_test <- stroke_data[test_index,]
```

#### stroke\_train and stroke\_test Object Summary:

```
# Summary information for the "stroke_train" dataset.
summary(stroke_train)
```

```
##           id           gender           age           hypertension
## Min.      : 67      Male :1685      Min.      : 0.08      wo_hypertension:3699
## 1st Qu.:12102      Female:2402      1st Qu.:25.00      w_hypertension : 388
## Median :32840                                     Median :44.00
## Mean      :33777                                     Mean      :42.94
## 3rd Qu.:52540                                     3rd Qu.:60.00
## Max.      :72940                                     Max.      :82.00
##           heart_disease           ever_married           work_type
## wo_heart_disease:3871      never_married      :1421      Underage_child : 560
## w_heart_disease : 216      is_or_was_married:2666      Never_worked   : 15
##                                     Self-employed   : 649
##                                     Private_company:2330
##                                     Govt_job        : 533
##
## Residence_type      bmi           avg_glucose_level           smoking_status
## Rural:2023          Min.      :10.30      Min.      : 55.22      never smoked    :1494
## Urban:2064          1st Qu.:23.80      1st Qu.: 77.46      formerly smoked: 714
##                                     Median :28.50      Median : 92.23      smokes          : 640
##                                     Mean      :28.92      Mean      :106.40      Unknown         :1239
```

```
##          3rd Qu.:32.80    3rd Qu.:114.09
##          Max.    :97.60    Max.    :267.76
##          stroke
## no_stroke:3888
## stroke   : 199
##
##
##
##
```

```
# Summary information for the "stroke_test" dataset.
summary(stroke_test)
```

```
##          id          gender          age          hypertension
## Min.    :   91    Male   :430    Min.    : 0.16    wo_hypertension:912
## 1st Qu.:11776    Female:592    1st Qu.:26.00    w_hypertension :110
## Median :32480                                Median :46.00
## Mean   :34023                                Mean   :44.41
## 3rd Qu.:53333                                3rd Qu.:63.00
## Max.   :72867                                Max.   :82.00
##          heart_disease          ever_married          work_type
## wo_heart_disease:962    never_married    :335    Underage_child :127
## w_heart_disease : 60    is_or_was_married:687    Never_worked   : 7
##                                     Self-employed   :170
##                                     Private_company:594
##                                     Govt_job       :124
##
## Residence_type          bmi          avg_glucose_level          smoking_status
## Rural:490    Min.    :13.00    Min.    : 55.12    never smoked    :398
## Urban:532    1st Qu.:23.50    1st Qu.: 76.20    formerly smoked:170
##                                     Median :28.10    Median : 90.28    smokes         :149
##                                     Mean   :28.83    Mean   :105.10    Unknown       :305
##                                     3rd Qu.:33.08    3rd Qu.:113.93
##                                     Max.   :78.00    Max.   :271.74
##          stroke
## no_stroke:972
## stroke   : 50
##
##
##
##
```

The analysis and charting in the next section (Section 4.4) will use the larger `stroke_data` dataset, while the modeling section (Section 5) will use the smaller `stroke_train/stroke_test` datasets for model development.

## 4.4 Dataset Analysis

Reviewing the datasets in raw format, along with creating correlative visual aids is necessary when comparing and contrasting the data. Basic analysis of each dataset has been done following their creation in previous sections. This section will focus on studying the data itself and any correlations that can aid in model selection.

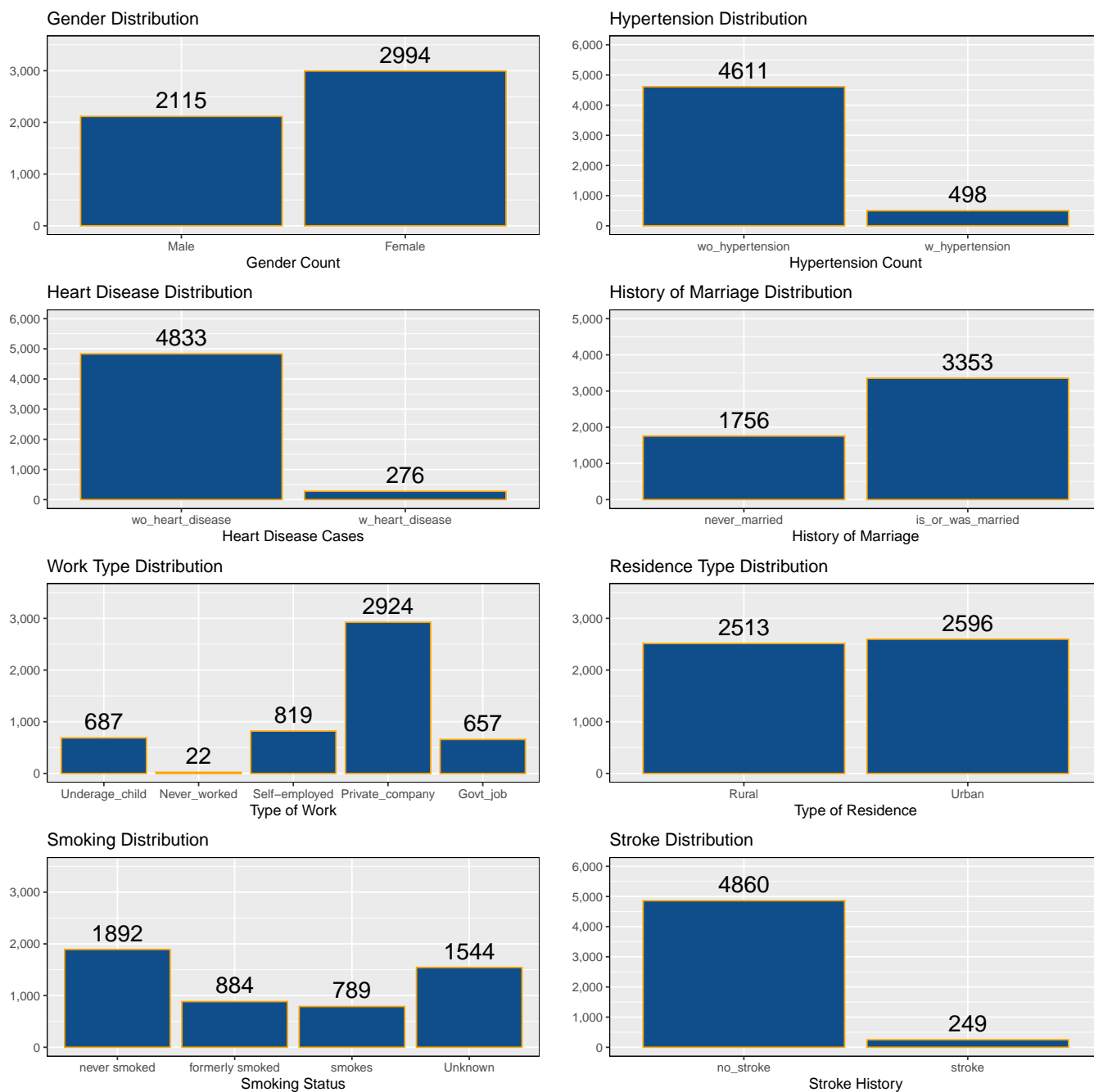
### stroke\_data Object Glimpse:

```
# Get a quick glimpse of the data in stroke_data.  
glimpse(stroke_data, width=80)
```

```
## Rows: 5,109  
## Columns: 12  
## $ id          <int> 9046, 9047, 9048, 9049, 9050, 9051, 9052, 9053, 9054~  
## $ gender      <fct> Male, Female, Male, Female, Female, Male, Male, Fema~  
## $ age         <dbl> 67, 61, 80, 49, 79, 81, 74, 69, 59, 78, 81, 61, 54, ~  
## $ hypertension <fct> wo_hypertension, wo_hypertension, wo_hypertension, w~  
## $ heart_disease <fct> w_heart_disease, wo_heart_disease, w_heart_disease, ~  
## $ ever_married <fct> is_or_was_married, is_or_was_married, is_or_was_marr~  
## $ work_type    <fct> Private_company, Self-employed, Private_company, Pri~  
## $ Residence_type <fct> Urban, Rural, Rural, Urban, Rural, Urban, Rural, Urb~  
## $ bmi         <dbl> 36.60000, 30.47129, 32.50000, 34.40000, 24.00000, 29~  
## $ avg_glucose_level <dbl> 228.69, 202.21, 105.92, 171.23, 174.12, 186.21, 70.0~  
## $ smoking_status <fct> formerly smoked, never smoked, never smoked, smokes,~  
## $ stroke       <fct> stroke, stroke, stroke, stroke, stroke, stroke, stro~
```



In **Figure 4.4.1** below, a grid of eight bar graphs shows a graphical breakdown of all factors and their individual category (level) quantities. The Male/Female ratio is split fairly close at 41% to 59%, respectively. The distribution of individuals diagnosed with hypertension, heart\_disease, and/or stroke is very imbalanced, which will require further correlative review among the other predictors to determine any statistical significance. The smoking\_status factor has a fairly large quantity of individuals in the Unknown category (where their status was unavailable at time of entry), which could skew the correlation coefficient between it and stroke. The residence type is split almost 50/50 between Rural and Urban, which means that predictor will have little to no impact on the prediction model.

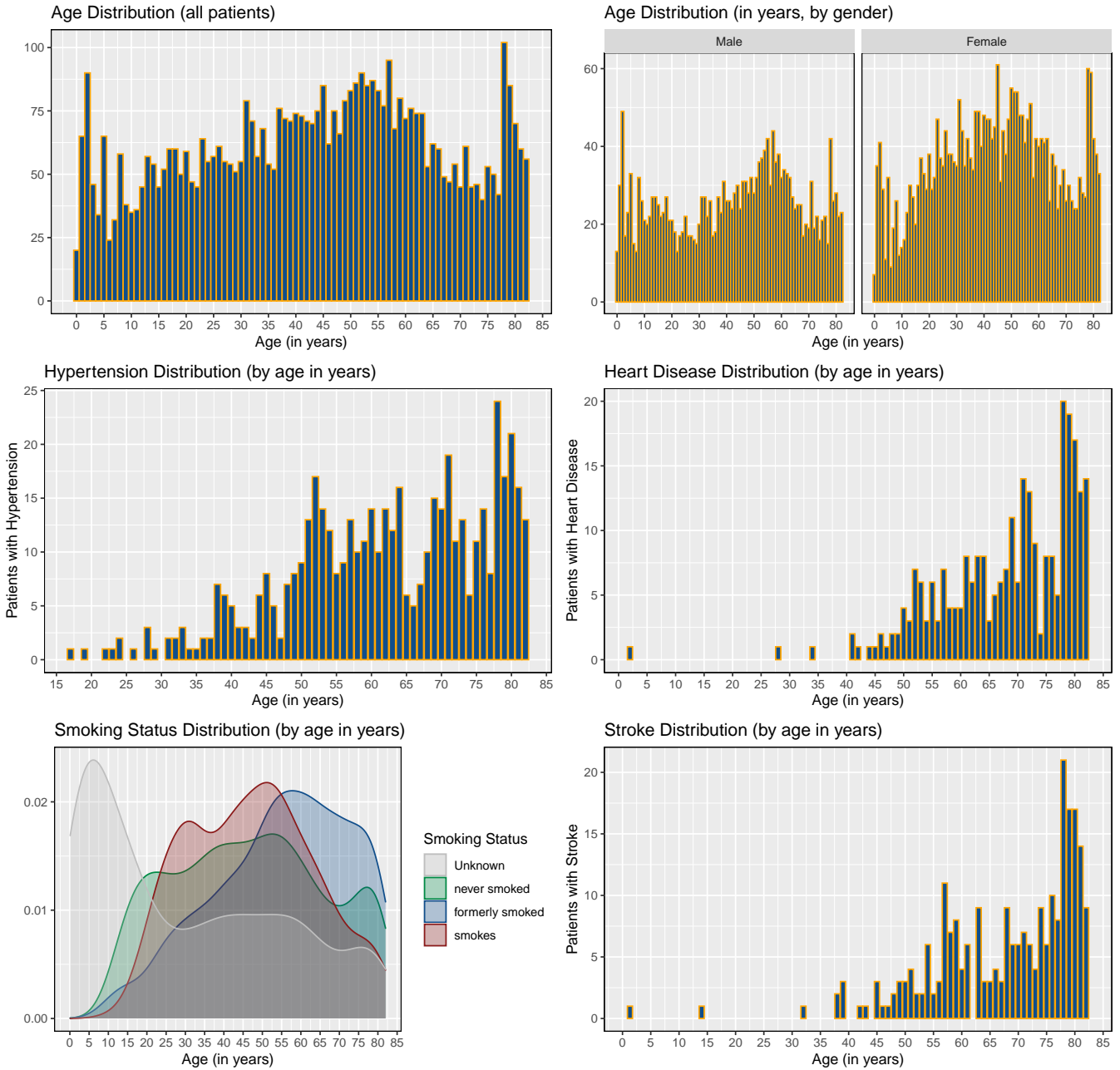


Source Data: stroke\_data  
Figure 4.4.1

In **Figure 4.4.2** below, a grid of six charts shows a graphical comparison of various predictors as they relate to patient age. When reviewing the charts showing hypertension, heart disease, and history of stroke as it relates to patient age, those predictors have little prevalence in patients under 35 years old.

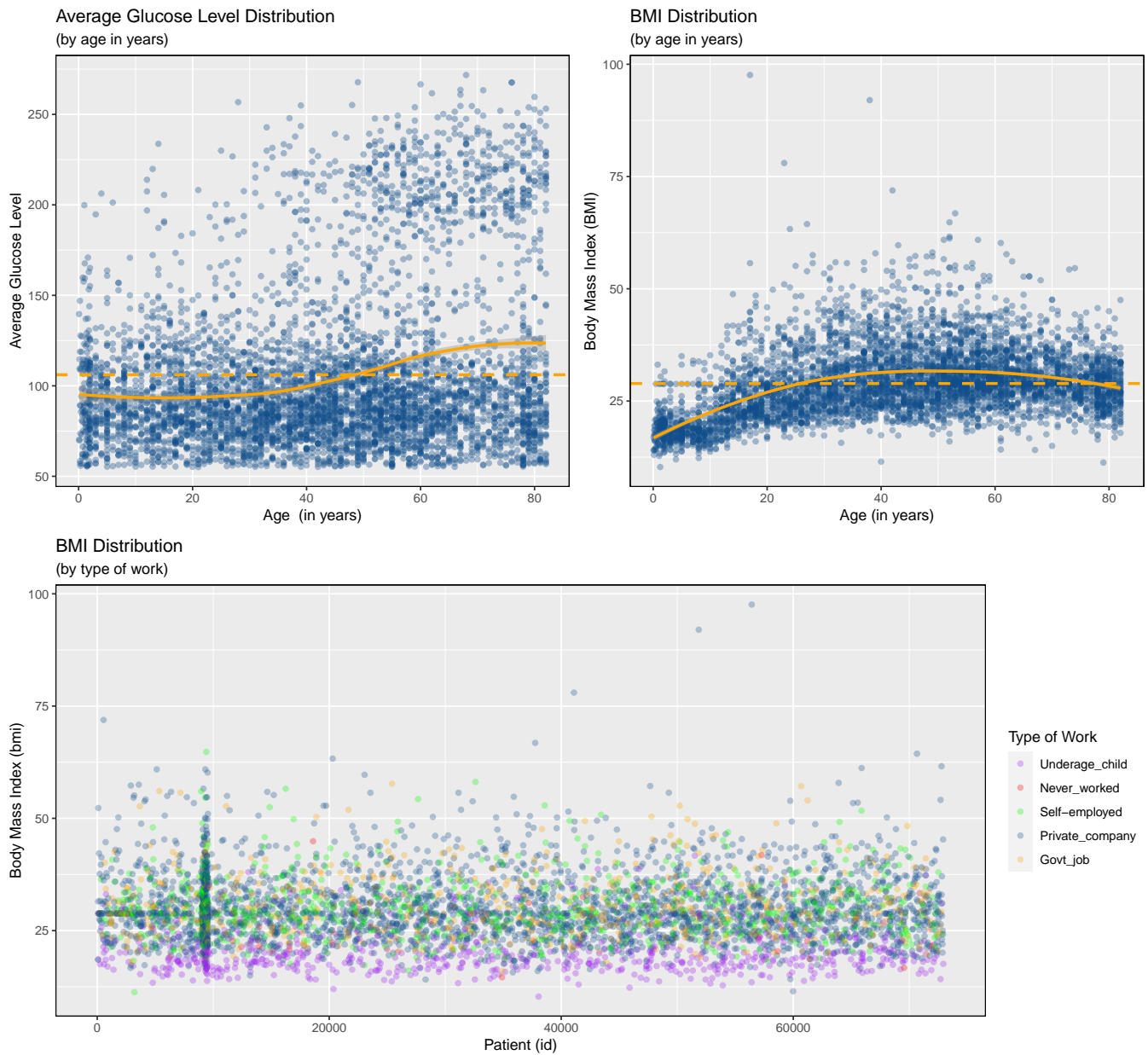
The Unknown smoking status category, when stratified by age, shows a trend in minors that can be explained by the parent/guardian of the patient viewing the smoking section of the doctor’s/hospital admission forms as “not applicable”. The relevant age of the `smoking_status` predictor, unlike hypertension and heart disease, starts around age 15 as that is when the `smokes` category increases rapidly.

The age distribution by gender shows a similar curve between Male/Female from ages 30 and up, but there are more Male patients that are minors.



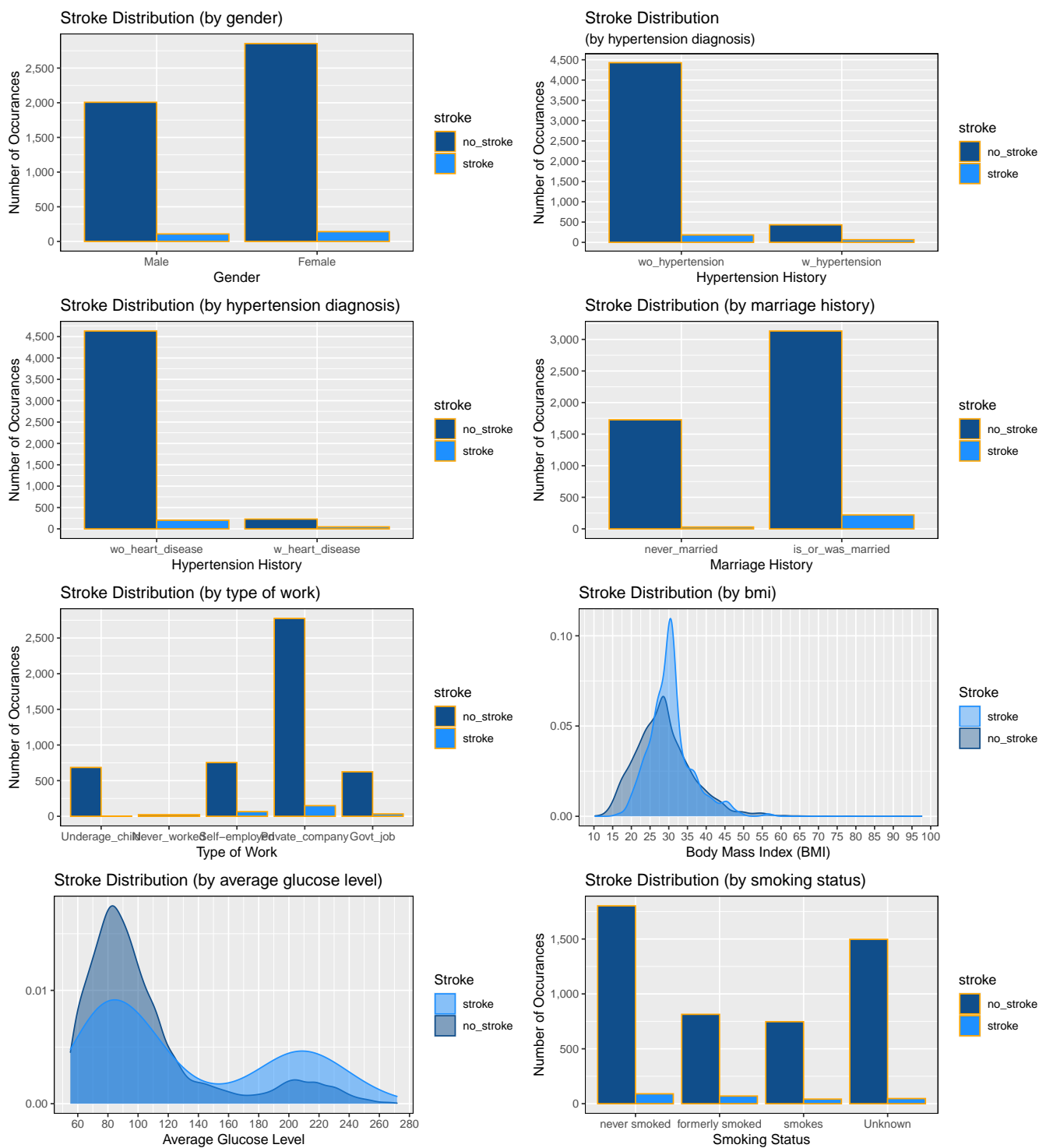
Source Data: `stroke_data`  
Figure 4.4.2

The average blood sugar (glucose) level and BMI distribution charts (**Figure 4.4.3 below**) both show anticipated trends (though both are fairly flat) where the body's ability to manage blood sugar lowers over time (causing the average levels to increase with age), and the BMI increases as you reach middle-age, then lowers a little in the very late stages of life.



Source Data: *stroke\_data*  
Figure 4.4.3

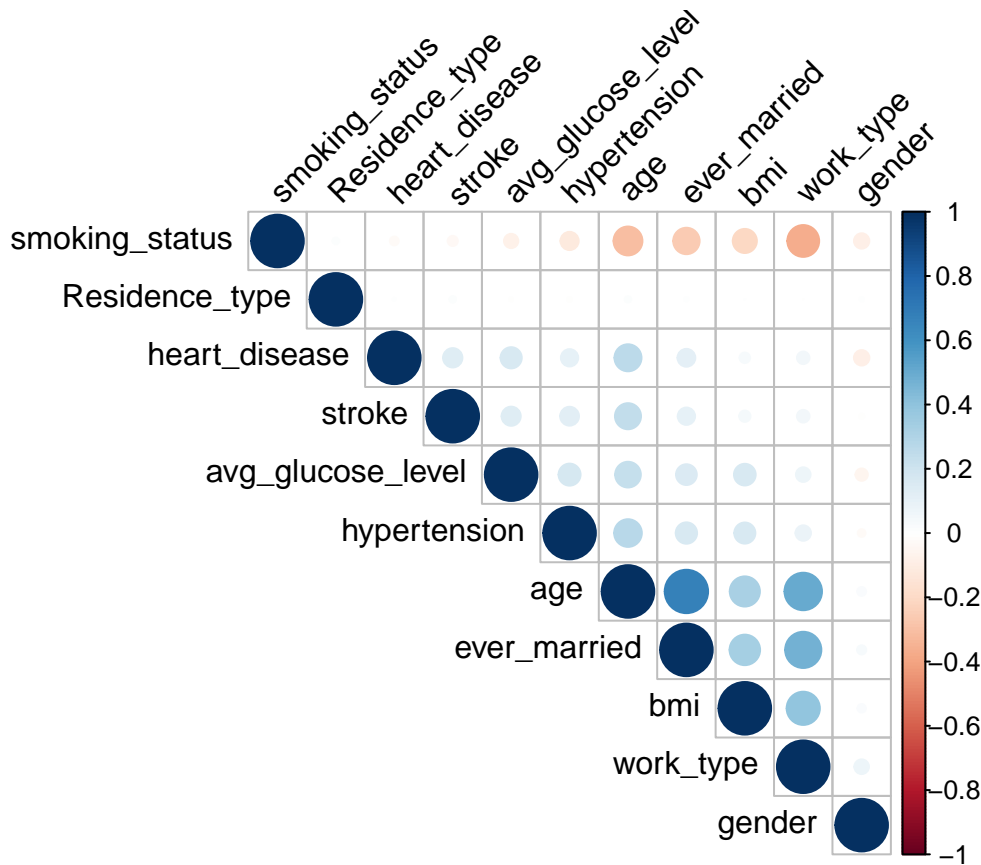
In **Figure 4.4.4** below, a grid of eight charts shows a graphical comparison of various predictors as they relate to stroke occurrence. Since **Figure 4.4.1** showed a stroke count of only 249, out of a possible 5109, it is unsurprising that they appear highly imbalanced. All charts show an evenly-scaled distribution of stroke/no\_stroke occurrences among each predictor (not previously charted) or its categories.



Source Data: stroke\_data  
Figure 4.4.4

While the charts allow the ability to correlate the predictors to the outcome by sight, it is as (if not more) important to confirm statistically what can be surmised visually. Using the `corrplot` package, a visual representation of actual correlation coefficient data can be maintained:

### Correlation Coefficient Chart



```
# Create correlation variables for each correlation coefficient's output
cor_gender <- cor(stroke_data_num$gender, stroke_data_num$stroke)
cor_age <- cor(stroke_data_num$age, stroke_data_num$stroke)
cor_hypertension <- cor(stroke_data_num$hypertension, stroke_data_num$stroke)
cor_heart_disease <- cor(stroke_data_num$heart_disease, stroke_data_num$stroke)
cor_ever_married <- cor(stroke_data_num$ever_married, stroke_data_num$stroke)
cor_work_type <- cor(stroke_data_num$work_type, stroke_data_num$stroke)
cor_Residence_type <- cor(stroke_data_num$Residence_type, stroke_data_num$stroke)
cor_bmi <- cor(stroke_data_num$bmi, stroke_data_num$stroke)
cor_avg_glucose_level <- cor(stroke_data_num$avg_glucose_level, stroke_data_num$stroke)
cor_smoking_status <- cor(stroke_data_num$smoking_status, stroke_data_num$stroke)

# Create a table title variable with the list of dataset variable names
cor_table_titles <- c("gender", "age", "hypertension", "heart_disease", "ever_married",
                      "work_type", "Residence_type", "bmi", "avg_glucose_level",
                      "smoking_status")
```

```

# Create a variable with a list of dataset correlation coefficient results.
cor_table_coefs <- c(cor_gender,cor_age,cor_hypertension,cor_heart_disease,
                    cor_ever_married,cor_work_type,cor_Residence_type,cor_bmi,
                    cor_avg_glucose_level,cor_smoking_status)
# Create a table with the "cor_table_titles" contents in the left column, and
# the "cor_table_coefs" contents in the right column.
kable(tibble(cor_table_titles, cor_table_coefs),
      col.names = c("Variable Name", "Correlation Coefficient to Stroke")) %>%
  row_spec(0,background="#104E8B", color="white") %>%
  column_spec(2, bold=TRUE) %>%
  kable_styling(bootstrap_options="bordered",
                full_width=FALSE,
                position="center",
                latex_options="HOLD_position")

```

Variable Name	Correlation Coefficient to Stroke
gender	<b>-0.0090806</b>
age	<b>0.2452388</b>
hypertension	<b>0.1278913</b>
heart_disease	<b>0.1349048</b>
ever_married	<b>0.1082993</b>
work_type	<b>0.0573656</b>
Residence_type	<b>0.0154146</b>
bmi	<b>0.0460594</b>
avg_glucose_level	<b>0.1319912</b>
smoking_status	<b>-0.0366669</b>

The table above shows the correlation coefficient amounts for each applicable variable. The id column was removed during dataset creation since there's no direct correlation with that field, and running the `cor()` command against the stroke variable would always be 1, so it was not added to the table.

The data shows that the age, heart\_disease, avg\_glucose\_level, and hypertension are the most relevant predictors to stroke.

## 5. MODELING

Throughout the data analysis performed above, some data point comparisons show the possibility of higher correlations while others appear they would have little impact to the overall predictability, and thus the model. To determine the impact of each predictor, various models will be tested, starting with logistic regression and then including Naive Bayes, linear discriminant analysis, classification and regression trees, random forest, and k-Nearest Neighbor. Each resultant accuracy calculation will be summarized in a list with previous results as to make it easy to compare all results and determine which model performs the best (having the highest accuracy score).

### Cross-Validation Training Options Code Snippet:

```
# Model cross-validation training options variable creation.
fitControl <- trainControl(method = "cv", #cross-validation
                           number = 10) #10-fold cross-validation
```

### Model 1: Logistic Regression (GLM)

The logistic regression is a very basic and frequently used machine learning model where independent variables determine a binary outcome, such as the stroke prediction in this dataset. The conditional probability can be modeled as:

$$g\{Pr(Y = 1|X = x)\} = \beta_0 + \beta_1$$

```
set.seed(1, sample.kind = "Rounding")
# Train the logistic regression model with the generalized linear model (GLM)
# method, using the fitControl tuning parameters.
model1_fit <- train(stroke ~ ., data = stroke_train,
                   method = "glm",
                   preProcess=c("center", "scale"),
                   trControl = fitControl)
model1_fit
```

```
## Generalized Linear Model
##
## 4087 samples
## 11 predictor
## 2 classes: 'no_stroke', 'stroke'
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 3678, 3678, 3678, 3678, 3678, 3678, ...
## Resampling results:
##
## Accuracy Kappa
## 0.9451945 0.1148134
```

```
# Predict the outcome from the stroke_test dataset
model1_preds <- predict(model1_fit, stroke_test)

# Calculate the accuracy of the model against the stroke_test dataset.
model1_accuracy <- confusionMatrix(model1_preds, stroke_test$stroke)$overall["Accuracy"]
model1_accuracy
```

```
## Accuracy
## 0.9432485
```

Using the `stroke_test` data, the accuracy for this model is calculated to be 0.9432485.

Model	Accuracy
Model 1: Logistic Regression (GLM)	<b>0.9432485</b>

## Model 2: Naive Bayes

The Naive Bayes model, based on Bayes Theorem<sup>6</sup>, estimates the outcome by estimating the conditional distribution of the predictors, which also assumes independence among the predictors. The naive model can be written as:

$$p(x) = Pr(Y = 1|X = x) = \frac{f_{X|Y=1}(X)Pr(Y = 1)}{f_{X|Y=0}(X)Pr(Y = 0) + f_{X|Y=1}(X)Pr(Y = 1)}$$

With  $f_{X|Y=1}$  and  $f_{X|Y=0}$  representing the distribution functions of the predictor  $X$  for the two classes  $Y = 1$  and  $Y = 0$ .

```
set.seed(1, sample.kind = "Rounding")
# Train the Naive Bayes model using the fitControl tuning parameters.
model2_fit <- train(stroke ~ ., data = stroke_train,
                    method = "naive_bayes",
                    preProcess=c("center", "scale"),
                    trControl = fitControl)
model2_fit
```

```
## Naive Bayes
##
## 4087 samples
## 11 predictor
## 2 classes: 'no_stroke', 'stroke'
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 3678, 3678, 3678, 3678, 3678, 3678, ...
```

---

<sup>6</sup><https://rafalab.github.io/dsbook/>



```
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy  Kappa
##   FALSE      0.8277512  0.3047345
##   TRUE       0.9625683  0.3691158
##
## Tuning parameter 'laplace' was held constant at a value of 0
## Tuning
## parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were laplace = 0, usekernel = TRUE
## and adjust = 1.
```

```
# Predict the outcome from the stroke_test dataset
model2_preds <- predict(model2_fit, stroke_test)

# Calculate the accuracy of the model against the stroke_test dataset.
model2_accuracy <- confusionMatrix(model2_preds, stroke_test$stroke)$overall["Accuracy"]
model2_accuracy
```

```
## Accuracy
## 0.9657534
```

Using the stroke\_test data, the accuracy for this model is calculated to be 0.9657534.

Model	Accuracy
Model 1: Logistic Regression (GLM)	<b>0.9432485</b>
Model 2: Naive Bayes	<b>0.9657534</b>

### Model 3: Linear Discriminant Analysis (LDA)

The linear discriminant analysis model improved upon Naive Bayes by using a dimensionality reduction technique that estimates the probability that a new set of inputs belongs to every class. The resultant output class is the one that has the highest probability (the prediction).

```
set.seed(1, sample.kind = "Rounding")
# Train the LDA model using the fitControl tuning parameters.
model3_fit <- train(stroke ~ ., data = stroke_train,
                    method = "lda",
                    preProcess=c("center", "scale"),
                    trControl = fitControl)
model3_fit
```

```
## Linear Discriminant Analysis
##
```

```
## 4087 samples
## 11 predictor
## 2 classes: 'no_stroke', 'stroke'
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 3678, 3678, 3678, 3678, 3678, 3678, ...
## Resampling results:
##
## Accuracy Kappa
## 0.9469078 0.1480155
```

```
# Predict the outcome from the stroke_test dataset
model3_preds <- predict(model3_fit, stroke_test)

# Calculate the accuracy of the model against the stroke_test dataset.
model3_accuracy <- confusionMatrix(model3_preds, stroke_test$stroke)$overall["Accuracy"]
model3_accuracy
```

```
## Accuracy
## 0.9510763
```

Using the `stroke_test` data, the accuracy for this model is calculated to be 0.9510763.

Model	Accuracy
Model 1: Logistic Regression (GLM)	<b>0.9432485</b>
Model 2: Naive Bayes	<b>0.9657534</b>
Model 3: Linear Discriminant Analysis (LDA)	<b>0.9510763</b>

## Model 4: Classification and Regression Trees Model (CART)

Classification (decision) trees build up a set of decision rules that “branch out” to form a tree structure, similar to a flow chart. This branching helps predict an outcome based on the input data.

The Gini Index:

$$\text{Gini}(j) = \sum_{k=1}^K \hat{p}_{j,k}(1 - \hat{p}_{j,k})$$

or Entropy (below) can be used as a metric for classifiers in the classification tree method.

$$\text{entropy}(j) = - \sum_{k=1}^K \hat{p}_{j,k} \log(\hat{p}_{j,k}), \text{ with } 0 \times \log(0) \text{ defined as } 0$$

```

set.seed(1, sample.kind = "Rounding")
# Train the Decision Tree model, setting the tuning parameters.
model4_fit <- train(stroke ~ ., data = stroke_train,
                    method = "rpart",
                    preProcess=c("center", "scale"),
                    tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 25)))
model4_fit

```

```

## CART
##
## 4087 samples
## 11 predictor
## 2 classes: 'no_stroke', 'stroke'
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 4087, 4087, 4087, 4087, 4087, 4087, ...
## Resampling results across tuning parameters:
##
##  cp            Accuracy  Kappa
##  0.000000000    0.9971477  0.9693575
##  0.004166667    0.9971477  0.9693575
##  0.008333333    0.9970670  0.9683095
##  0.012500000    0.9970135  0.9678374
##  0.016666667    0.9968814  0.9664043
##  0.020833333    0.9968036  0.9656196
##  0.025000000    0.9965630  0.9632909
##  0.029166667    0.9965364  0.9630492
##  0.033333333    0.9966426  0.9642094
##  0.037500000    0.9965629  0.9633829
##  0.041666667    0.9965625  0.9637581
##  0.045833333    0.9964822  0.9629492
##  0.050000000    0.9964288  0.9624573
##  0.054166667    0.9964022  0.9621480
##  0.058333333    0.9964022  0.9621480
##  0.062500000    0.9964289  0.9624559
##  0.066666667    0.9964289  0.9624559
##  0.070833333    0.9965363  0.9635234
##  0.075000000    0.9965363  0.9635234
##  0.079166667    0.9965363  0.9635234
##  0.083333333    0.9965363  0.9635234
##  0.087500000    0.9965363  0.9635234
##  0.091666667    0.9965363  0.9635234
##  0.095833333    0.9965363  0.9635234
##  0.100000000    0.9965363  0.9635234
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.004166667.

```

```
# Predict the outcome from the stroke_test dataset
model4_preds <- predict(model4_fit, stroke_test)

# Calculate the accuracy of the model against the stroke_test dataset.
model4_accuracy <- confusionMatrix(model4_preds, stroke_test$stroke)$overall["Accuracy"]
model4_accuracy
```

```
## Accuracy
## 0.9970646
```

Using the `stroke_test` data, the accuracy for this model is calculated to be 0.9970646.

Model	Accuracy
Model 1: Logistic Regression (GLM)	<b>0.9432485</b>
Model 2: Naive Bayes	<b>0.9657534</b>
Model 3: Linear Discriminant Analysis (LDA)	<b>0.9510763</b>
Model 4: Classification and Regression Trees Model (CART)	<b>0.9970646</b>

## Model 5: Random Forest

With the CART model being prone to over-fitting, the random forest model compensates for this by using decision trees to generate many predictors, and then averages them until a final prediction is made based on the averages. The trade-off for its performance gain is the loss of interpret ability of the data.

This model uses the `Rborist` method, which a high-performance implementation of random forest.

```
set.seed(1, sample.kind = "Rounding")
# Train the Random Forest model using the "Rborist" method. Set basic decision tree
# tuning parameters.
model5_fit <- train(stroke ~ ., data = stroke_train,
                    method = "Rborist",
                    preProcess=c("center", "scale"),
                    tuneGrid = expand.grid(predFixed = seq(1,4), minNode = 2))
model5_fit
```

```
## Random Forest
##
## 4087 samples
## 11 predictor
## 2 classes: 'no_stroke', 'stroke'
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 4087, 4087, 4087, 4087, 4087, 4087, ...
## Resampling results across tuning parameters:
```

```
##
##   predFixed Accuracy   Kappa
##   1         0.9515451 0.0000000
##   2         0.9667106 0.4684863
##   3         0.9938188 0.9304263
##   4         0.9958455 0.9549333
##
## Tuning parameter 'minNode' was held constant at a value of 2
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were predFixed = 4 and minNode = 2.
```

```
# Predict the outcome from the stroke_test dataset
model5_preds <- predict(model5_fit, stroke_test)

# Calculate the accuracy of the model against the stroke_test dataset.
model5_accuracy <- confusionMatrix(model5_preds, stroke_test$stroke)$overall["Accuracy"]
model5_accuracy
```

```
## Accuracy
## 0.9980431
```

Using the `stroke_test` data, the accuracy for this model is calculated to be 0.9980431.

Model	Accuracy
Model 1: Logistic Regression (GLM)	<b>0.9432485</b>
Model 2: Naive Bayes	<b>0.9657534</b>
Model 3: Linear Discriminant Analysis (LDA)	<b>0.9510763</b>
Model 4: Classification and Regression Trees Model (CART)	<b>0.9970646</b>
Model 5: Random Forest	<b>0.9980431</b>

## Model 6: k-Nearest Neighbor (kNN)

The k-nearest neighbor model searches its neighboring data, assuming that similar data points exist close by. The most similar data points to the ones you have to predict are found by averaging the neighboring values, or by most frequent class. Decreasing the value of K yields less accurate predictions, and increasing it too high produces errors which also yields less accurate predictions.

```
set.seed(1, sample.kind = "Rounding")
# Train the k-Nearest Neighbors model, setting the k-value tuning parameters.
model6_fit <- train(stroke ~ ., data = stroke_train,
                    method = "knn",
                    preProcess=c("center", "scale"),
                    tuneGrid = data.frame(k = seq(1,100,5)))
model6_fit
```

```
## k-Nearest Neighbors
##
## 4087 samples
## 11 predictor
## 2 classes: 'no_stroke', 'stroke'
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 4087, 4087, 4087, 4087, 4087, 4087, ...
## Resampling results across tuning parameters:
##
## k Accuracy Kappa
## 1 0.9254074 1.663073e-01
## 6 0.9403141 1.005008e-01
## 11 0.9482324 4.213315e-02
## 16 0.9499454 2.295856e-02
## 21 0.9508775 1.004189e-02
## 26 0.9512245 5.762871e-03
## 31 0.9511992 1.440415e-03
## 36 0.9513853 5.359097e-04
## 41 0.9514645 -1.576254e-04
## 46 0.9514645 -1.576254e-04
## 51 0.9514910 -1.053922e-04
## 56 0.9515180 -5.334865e-05
## 61 0.9515451 0.000000e+00
## 66 0.9515451 0.000000e+00
## 71 0.9515451 0.000000e+00
## 76 0.9515451 0.000000e+00
## 81 0.9515451 0.000000e+00
## 86 0.9515451 0.000000e+00
## 91 0.9515451 0.000000e+00
## 96 0.9515451 0.000000e+00
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 96.
```

```
# Predict the outcome from the stroke_test dataset
model6_preds <- predict(model6_fit, stroke_test)

# Calculate the accuracy of the model against the stroke_test dataset.
model6_accuracy <- confusionMatrix(model6_preds, stroke_test$stroke)$overall["Accuracy"]
model6_accuracy
```

```
## Accuracy
## 0.9510763
```

Using the `stroke_test` data, the accuracy for this model is calculated to be 0.9510763.

Model	Accuracy
Model 1: Logistic Regression (GLM)	<b>0.9432485</b>
Model 2: Naive Bayes	<b>0.9657534</b>
Model 3: Linear Discriminant Analysis (LDA)	<b>0.9510763</b>
Model 4: Classification and Regression Trees Model (CART)	<b>0.9970646</b>
Model 5: Random Forest	<b>0.9980431</b>
Model 6: k-Nearest Neighbor (kNN)	<b>0.9510763</b>

## RESULTS

Having done all of the initial data analysis in section 4 and pinpointing which factors may contribute to (or at least warrant a model to test) an accurate stroke prediction system, stepping through six (6) different models yielded the most accurate results with the use of the Random Forest model.

While testing the various performance tuning options of each model a few models ended up having extended compute-intensive time frames (up to 2 hours on an 8-core hyperthreaded processor with 64GB RAM). The final tests with the parameters used in this report only took around 30 minutes to complete from start to finish.

The final accuracy results table below shows the summary list of all prediction models tested for a stroke prediction system using the `stroke_train` and `stroke_test` datasets.

Model	Accuracy
Model 1: Logistic Regression (GLM)	<b>0.9432485</b>
Model 2: Naive Bayes	<b>0.9657534</b>
Model 3: Linear Discriminant Analysis (LDA)	<b>0.9510763</b>
Model 4: Classification and Regression Trees Model (CART)	<b>0.9970646</b>
Model 5: Random Forest	<b>0.9980431</b>
<b>Model 6: k-Nearest Neighbor (kNN)</b>	<b>0.9510763</b>

## CONCLUSION

The purpose of this project was to create a stroke prediction system, using a public dataset from Kaggle, and building multiple machine learning models, selecting the one with the highest accuracy. From initial GitHub repository creation, to data wrangling, dataset analysis, data visualizations, and model development, the knowledge and techniques learned throughout the entire course series has been used.

Both the logistic regression, Naive Bayes, and linear discriminant models all performed their calculations very quickly, and all used the same tuning parameters with results that were all still very high (0.9432485, 0.9657534, and 0.9510763, respectively). The Random Forest and k-Nearest Neighbors tuning parameters were more difficult to test as the calculation times would get extended, and the CPU resources would be locked to the process until completed each time.

After testing 6 different predictive models, using their respective calculated accuracy scores as the success indicator for each, the Random Forest model yielded the highest score of 0.9510763. Comparing this accuracy score to the initial Logistic Regression model (0.9432485), there was an improvement of approximately 0.783%.

While the dataset contained many of the health factors that were considered risks by the Centers for Disease Control (CDC) and the National Institute of Neurological Disorders and Stroke (NINDS), as indicated in the Introduction, limitations to the model included several other missing predictors that could be added to future tests, such as Race, geographic location, alcohol drinking habits, drug use, cholesterol



level, etc. Being able to inform both the patient and their doctor of their risk of stroke could lead to proactive treatment, better life choices, and a prolonged life span. The data in this particular dataset showed age, heart\_disease, avg\_glucose\_level, and hypertension are the most relevant predictors to stroke.

Further improvements and future work include an expanded dataset with more samples, additional performance tuning on all of the models in the future (especially in the event that additional factors get added). Additional models, such as the ensemble method, could be developed and tuned to test whether further accuracy can be achieved.

## REFERENCES

1. Data Science textbook by Rafael Irizarry, <https://rafalab.github.io/dsbook>.
2. Kaggle, Stroke Dataset, <https://www.kaggle.com/dhirendommeti/stroke>
3. National Institute of Neurological Disorders and Stroke (NINDS), <https://www.stroke.nih.gov/materials/needtoknow.htm>
4. Centers for Disease Control (CDC), <https://www.cdc.gov/stroke/facts.htm>
5. Mayo Clinic, <https://www.mayoclinic.org/diseases-conditions/stroke/symptoms-causes/syc-20350113>