Politecnico di Torino

Assignemnt 8

Stroke epidemiology

BioQuants

2023/2024

Professor: Benso Alfredo

Riccardo Racca

February 2024

Chapter 1

EDA - Exploratory Data Analysis

1.1 Data Description

According to the World Health Organization (WHO), stroke is the 2nd leading cause of death globally, responsible for approximately 11% of total deaths. This dataset is used to predict whether a patient is likely to get a stroke based on input parameters like gender, age, various diseases, and smoking status. Each row in the data provides relevant information about the patient. It has a total of 5109 patients and 12 columns, it is registered an imbalance in the target variables since the people who suffered from stroke are roughly 5 percentage of the total.

1.1.1 Attribute Information

1. **id:** Unique identifier [integer]
2. **gender:** "Male", "Female", or "Other" [character]
3. **age:** Age of the patient [numeric]
4. **hypertension:** 0 if the patient doesn't have hypertension, 1 if the patient has hypertension [integer]
5. **heart_disease:** 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease [integer]
6. **ever_married:** "No" or "Yes" [character]
7. **work_type:** "Children", "Govt_job", "Never_worked", "Private", or "Self-employed" [character]
8. **Residence_type:** "Rural" or "Urban" [character]
9. **avg_glucose_level:** Average glucose level in blood [numeric]
10. **bmi:** Body mass index [character]
11. **smoking_status:** "Formerly smoked", "Never smoked", "Smokes", or "Unknown"* [character]
12. **stroke:** 1 if the patient had a stroke or 0 if not [integer]

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

1.2 Data Cleaning

Now there will be handling missing values, outliers, and inconsistencies in the data. The first thing to notice, even though "bmi" was stored as "character", is more appropriate to change to "integer". Consequently, missing values were highlighted, showing presence in "bmi" as well as in "smoking_status". Lastly, in only one instance occurred "Other" as gender, thus it was simply discarded.
To handle the missing values that occurred in "bmi", it was first checked if they were many, and if they were largely present in the sub-population with target variable positive. Unfortunately, too many missing values are present to discard the rows. Therefore, a "filling" strategy was chosen, with the only intent of keeping the statistical nature of the data. Thus, initially, it was tried to check whether the distribution of "bmi", pruned from outliers, was normal. The histogram follows.

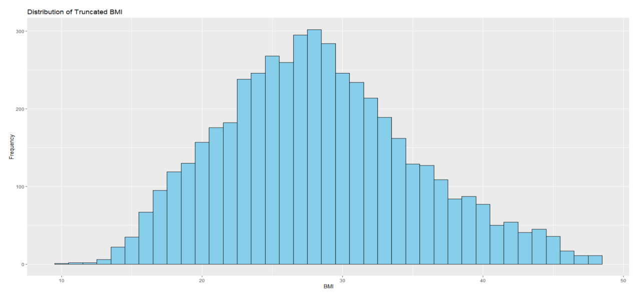

Figure 1.1: Distribution of bmi pruned from outliers

Even though it might seem that the assumption of normality is correct, the Shapiro-Wilk test, specifically designed to determine if a distribution can be assumed normal, proved the opposite. Below will be found the R-code and its output (this will be a frequent framework in this paper)

```r
# Computing mean and std
mu <- mean(truncated_bmi)  # Example mean
sigma <- sd(truncated_bmi)  # Example standard deviation

# Standardize the data
standardized_data <- (truncated_bmi - mu) / sigma

# Perform Shapiro-Wilk test for normality on the standardized data
shapiro_test_result <- shapiro.test(standardized_data)
```

Listing 1.1: R-Code: Shapiro-Wilk test

Output:

```
Shapiro-Wilk normality test

data:  standardized_data
W = 0.98867, p-value < 2.2e-16
```

Listing 1.2: Output: Shapiro-Wilk test

Consequently, the features "smoking_status" was analyzed, and in the same fashion as "bmi", the missing values, stored as "Unknown" class, were changed with samples, directly from the discrete sample distribution built on the data.
Further analyses were made also in the other columns, without any anomalies registered.
Finally, as usual, the column related to "id" was discarded, as it does not seem to have any statistical relevance.

1.3 Encoding

The Dummy encoding was chosen, and together with the addition of the intercept feature, i.e. a column full of 1's, the coefficients' matrix for building the Generalized Linear Model was created.

```
# Perform dummy encoding
encoded_df <- cbind(patients_df, model.matrix(~ gender + ever_married + Residence_
    type + work_type + smoking_status, data = patients_df))

# Remove the columns encoded to avoid collinearity
encoded_df <- select(encoded_df, -ever_married)
encoded_df <- select(encoded_df, -Residence_type)
encoded_df <- select(encoded_df, -work_type)
encoded_df <- select(encoded_df, -smoking_status)
encoded_df <- select(encoded_df, -gender)
```

Listing 1.3: R-Code: Dummy Encoding

To avoid listing again all variables, they can be read in the following section by looking at the heat map.

1.4 Correlation analysis

The next stage will be focused on the linear correlation analysis between features and target and within features. This is crucial for two reasons: firstly, understanding which features are the largest correlated with the target variable. Secondly, it is a way to check whether there is collinearity between features. The second aspect listed must be kept into account when building a (generalized) linear model. Indeed, if the collinearity is preserved, then the conclusion might be biased. It follows the code to build the Heat Map, and the Heat Map itself.

```
# Compute correlation matrix
correlation_matrix <- cor(encoded_df)

# Convert correlation matrix to long format
correlation_df <- melt(correlation_matrix)

# Plot heatmap using ggplot2
ggplot(correlation_df, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0) +
  labs(x = NULL, y = NULL, title = "Heatmap of Correlation Matrix") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.title = element_blank(),
        panel.grid = element_blank()) +
  coord_fixed()  # Fix aspect ratio
```

Listing 1.4: R-Code: Heat Map

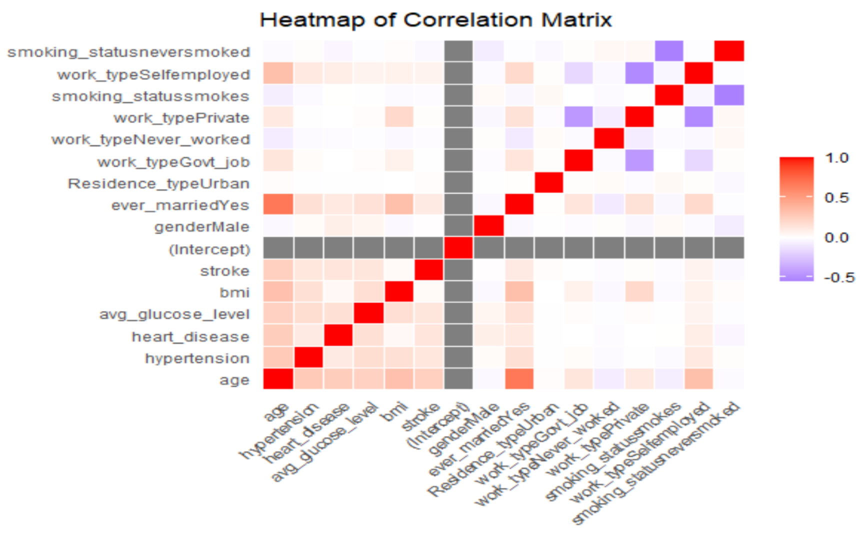

Figure 1.2: Heat Map of the encoded features and the target variable

After having set a heuristic threshold equal to 0.1, the features showing **absolute** correlation larger than the threshold with the target variable are: **"age"**, **"hypertension"**, **"heart_disease"**, **"avg_glucose_level"**, **"ever_marriedYes"**

Chapter 2

Biological Questions:

After having prepared the dataset, a series of requests related to biological aspects, of stroke epidemiology, will be tackled The following questions will contribute to giving further knowledge of the features involved, as well as highlighting their relationship with the target variable, covering some aspects that may be left out in EDA. Without further do, here are the questions:

1. What seem to be the highest risk factors in causing strokes?
2. Are there any demographic factors (such as age, gender, or marital status) associated with strokes?
3. How does the prevalence of strokes vary by age group, gender, and marital status?
4. What is the prevalence of hypertension and heart disease in the population?
5. Show some statistics to produce better conclusions.
6. Are certain occupational or work-related factors linked with a higher risk of stroke?
7. Is there a relationship between residence type (urban or rural) and the prevalence of strokes?
8. What is the average glucose level distribution in the population, and how does this affect the stroke tendency?
9. Can we predict the likelihood of strokes based on demographic factors and lifestyle choices?

2.1 What seem to be the highest risk factors in causing strokes?

The features showing the largest absolute linear correlation with the target "stroke" are: "age", "hypertension", "heart_disease", "avg_glucose_level", and "ever_marriedYes". All show a positive linear correlation with the target.
They can therefore be, in the first stages, assumed to be candidate risk factors.

2.2 Are there any demographic factors (such as age, gender, or marital status) associated with strokes?

Yes, there are. Indeed from the correlation analysis emerged that "age" and marital status may be a cause of strokes.

2.3 How does the prevalence of strokes vary by age group, gender, and marital status?

The following will show three plots, representing, respectively, the distributions of age, gender and marital status of the stroke-affected people. The R-code will be reported just for the first one, as the others are analogously done.

```r
# Age

# Filter dataset to include only rows where stroke = 1
patients_stroke_df <- patients_df %>%
  filter(stroke == 1)

# Visualization of stroke prevalence by age group
ggplot(patients_stroke_df, aes(x = age)) +
  geom_bar(fill = "brown") +
  labs(title = "Stroke Prevalence by Age Group", y = "Count of Patients") +
  theme_minimal()
```

Listing 2.1: R-Code: Stroke Prevalence by Age Group (Plot)

Output:

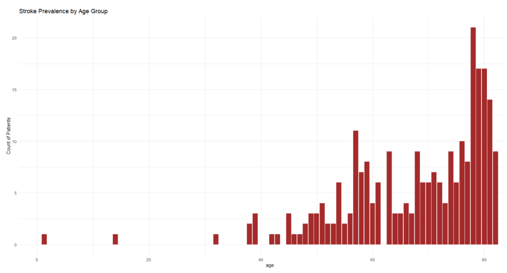

Figure 2.1: Stroke Prevalence by Age Group

Observations: Focusing on the people who suffered from stroke, in 2.1, can be seen a prevalence of old people, especially from mid-fifty on. The 2.2 shows an almost exact balance between genders, while 2.3 highlights a large prevalence of married people over the non-ones.

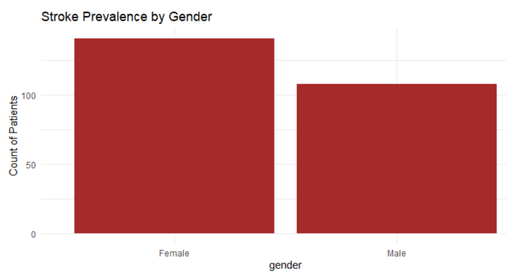

Figure 2.2: Stroke Prevalence by Gender

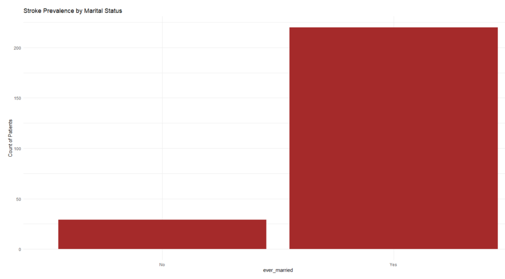

Figure 2.3: Stroke Prevalence by Marital Status

2.4 What is the prevalence of hypertension and heart disease in the population?

Now there will be listed the percentage of people with hypertension and heart disease. Firstly in the whole population, and then just focusing on the sub-population that have registered a stroke. To avoid unnecessary codes, it will be reported just one of 4 snipes of code, the others are analogue. The results obtained are:

- Percentage of people with hypertension: 9.75%
- Percentage of people with both hypertension and strokes: 26.51 %
- Percentage of people with heart disease: 5.4 %
- Percentage of people with both heart disease and strokes: 18.88 %

```
    # Count the number of individuals with hypertension
hypertension_count <- sum(patients_df$hypertension == 1)

# Calculate the total number of individuals in the dataset
total_count <- nrow(patients_df)

# Calculate the percentage of people with hypertension
percentage_hypertension <- (hypertension_count / total_count) * 100

# Print the percentage
print(paste("Percentage of people with hypertension:", round(percentage_
   hypertension, 2), "%"))
```

Listing 2.2: R-Code: Percentage of people with hypertension

Output:

```
"Percentage of people with hypertension: 9.75 \%"
```

Listing 2.3: Output: Percentage of people with hypertension

Both "hypertension" and "heart_disease" seem to be a relevant factor in causing stroke. Indeed hypertension is about three times larger in the "stroke" fraction of the population, while heart disease shows almost four times larger percentages. Further investigations will be done, to avoid any sort of bias due to the correlation of age and these two potential factors.

2.5 Show some statistics to produce better conclusion.

2.5.1 Brief theoretical introduction

The statistic test that will be proposed is called: **Test of Homogeneity**, a brief description follows:
Let's suppose to have two categorical variables A and C, for which is possible to compute the contingency table. Moreover let J and I be two natural numbers representing the size of the set of possible values of A and C respectively. Finally A_J and C_I are the sets of classes for the random variables.
For each fixed class of C, namely $c_i \in C_I$, it is counted the number of instances where a_j occurs together with c_i, this for every class $a_j \in A_J$. Now it can be questioned if A statistically affect C by checking whether the probability of c_i is different knowing that a certain a_j co-occurred.
This procedure can be re-iteratevly done with all $c_i \in C_I$.
More formally it is possible to set the following hypothesis test:
$H_0 : p_{c_j|a_1} = p_{c_j|a_2} = \ldots = p_{c_j|a_I} \; \forall \; c_j \in J$ and $H_1 = \bar{H_0}$
This leads to the following Homogeneity test, which will be computed for the couples:
hypertension-stroke, heart_disease-stroke, genderMale-stroke, everMarried-stroke.

2.5.2 Actual testing

```
# Define function for performing chi-square test
perform_chi_square_test <- function(data, var1, var2) {
  # Filter out missing values
  filtered_data <- na.omit(data[c(var1, var2)])

  # Create contingency table
  contingency_table <- table(filtered_data[[var1]], filtered_data[[var2]])

  # Perform chi-square test
  chi_square_test <- chisq.test(contingency_table)

  # Extract test statistics and p-value
  result <- list(
    ChiSquareStatistic = chi_square_test$statistic,
    PValue = chi_square_test$p.value
  )

  return(result)
}

# Perform chi-square test for each variable pair
homogeneity_test_results <- list()

# Chi-square test for "hypertension-stroke" pair
homogeneity_test_results[["hypertension-stroke"]] <- perform_chi_square_test(
    encoded_df, "hypertension", "stroke")
```

```r

# Chi-square test for "heart_disease-stroke" pair
homogeneity_test_results[["heart_disease-stroke"]] <- perform_chi_square_test(
    encoded_df, "heart_disease", "stroke")

# Chi-square test for "genderMale-stroke" pair
homogeneity_test_results[["genderMale-stroke"]] <- perform_chi_square_test(encoded
    _df, "genderMale", "stroke")

# Chi-square test for "ever_marriedYes-stroke" pair
homogeneity_test_results[["ever_marriedYes-stroke"]] <- perform_chi_square_test(
    encoded_df, "ever_marriedYes", "stroke")

# Print results
homogeneity_test_results)
```

Listing 2.4: R-Code: Homogeneity test

Output:

```
‘hypertension-stroke‘:
ChiSquareStatistic
X-squared
 81.57314
PValue
[1] 1.688936e-19


‘heart_disease-stroke‘:
ChiSquareStatistic
X-squared
 90.22944
PValue
[1] 2.120831e-21


‘genderMale-stroke‘:
ChiSquareStatistic
X-squared
0.3400025
PValue
[1] 0.5598278


‘ever_marriedYes-stroke‘:
ChiSquareStatistic
X-squared
 58.86781
 PValue
[1] 1.686286e-14
```

Listing 2.5: R-Code: Homogeneity test

Observations: Everything up to now shows that the information carried by: hypertension, heart disease, age and marital status are separately* statistically linked to strokes. Instead, concerned with ”genderMale”, the p-value is way over the threshold of 0.05, therefore it does not give any evidence of relevance in studying the stroke occurrences. This can be easily seen by checking the p-values, which in those cases are extremely low, way below 0.05 (heuristic but common threshold set in these analyses).

* Cross analysis will follow to understand whether they are all relevant, or a sub-set of them sufficiently explains just as well as them all.

2.6 Are certain occupational or work-related factors linked with a higher risk of stroke?

The first step is to plot the actual distribution of the work feature in the whole population, after this, the focus will be shifted to the fraction of people who had stroke. As usual, just one code will be provided for brevity.

```r
# Calculate percentage of each work_type category
work_type_counts <- patients_df %>%
  group_by(work_type) %>%
  summarise(count = n()) %>%
  mutate(percentage = (count / sum(count)) * 100)

# Plot bar graph of work_type with percentage annotations
ggplot(work_type_counts, aes(x = work_type, y = count)) +
  geom_bar(stat = "identity", fill = "brown") +
  geom_text(aes(label = paste0(round(percentage), "%")), vjust = -0.5, color = "
      black") +
  labs(title = "Distribution of Work Type", x = "Work Type", y = "Count") +
  theme_minimal()
```

Listing 2.6: R-Code: Calculate percentage of each work type category

Output:

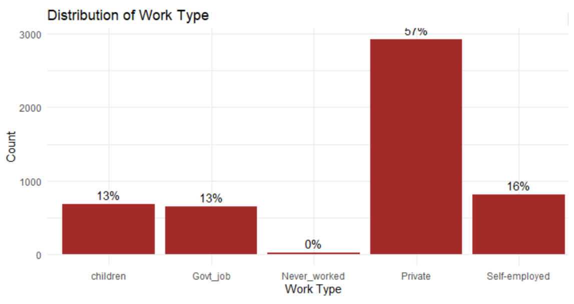

Figure 2.4: Distribution of work type

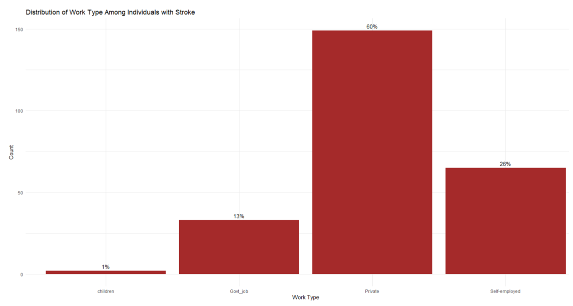

Figure 2.5: Distribution of work type in people who suffered stroke

Observations: From these bar-graphs it emerges that nobody who has "never-worked" suffered from stroke, which is why there is no bar in the 2.5. For classes Govt_job and Private, the percentages are almost the same in both pictures, while they differ significantly for children and self-employed. The former with much lower fraction than before and the latter approximately doubled w.r.t. the first graph.
From this analysis seems that self-employed might be the class with the highest risk of stroke. To consolidate the results, it is plotted below the chi-squared test of these classes, where indeed, just Self-employed has a significantly low p-value, encouraging the rejection of the null hypothesis.

```
'work_typePrivate-stroke'
ChiSquareStatistic
X-squared
0.6191734
PValue
```

```
[1] 0.4313546


‘work_typeGovt_job-stroke‘
$ChiSquareStatistic
  X-squared
0.008660334
PValue
[1] 0.9258552


’work_typeSelfemployed-stroke‘
ChiSquareStatistic
X-squared
 18.95549
PValue
[1] 1.33804e-05
```

Listing 2.7: R-Code: Homogeneity test for work type-stroke

2.7 Is there a relationship between residence type (urban or rural) and the prevalence of strokes?

The same framework just set will be proposed again below, the first block will represent the distribution for the whole dataset for the Residence type, while the second one will be focused on the stroke sub-dataset.

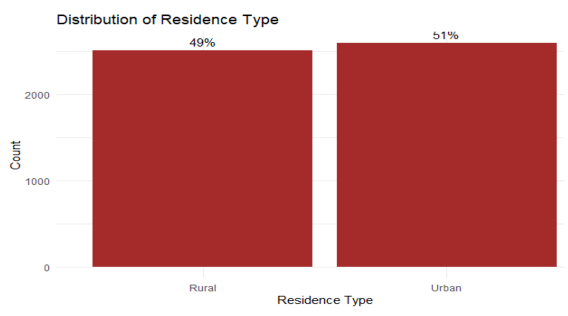

Figure 2.6: Distribution of Residence Type

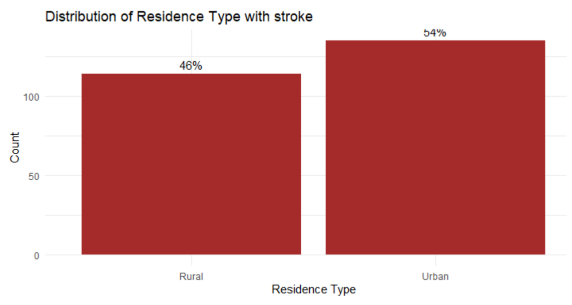

Figure 2.7: Distribution of Residence Type with Stroke

The Homogeneity test returns:

```
‘Residence_typeUrban-stroke‘
ChiSquareStatistic
X-squared
 1.074971
PValue
[1] 0.2998252
```

Listing 2.8: R-Code: Homogeneity test for Residence type (Urban) and stroke

Observations: The homogeneity test does not really give us any decisive result, therefore it is not possible to reject the null hypothesis of these two classes to be un-influent for strokes.

2.8 What is the average glucose level distribution in the population, and how does this affect the stroke tendency?

Let's firstly plot the distribution of the glucose level for the whole population, and then the one restricted to the stroke cases.

```
# Plot distribution of glucose level for the whole population
ggplot(patients_df, aes(x = avg_glucose_level)) +
  geom_density(fill = "skyblue", alpha = 0.7) +
  labs(title = "Distribution of Glucose Level (Whole Population)", x = "Average
      Glucose Level", y = "Density") +
  theme_minimal()

# Plot distribution of glucose level for individuals with stroke = 1
ggplot(patients_df %>% filter(stroke == 1), aes(x = avg_glucose_level)) +
  geom_density(fill = "orange", alpha = 0.7) +
  labs(title = "Distribution of Glucose Level (Stroke Cases)", x = "Average
      Glucose Level", y = "Density") +
  theme_minimal()
```

Listing 2.9: R-Code: Distribution of glucose - whole population and stroke

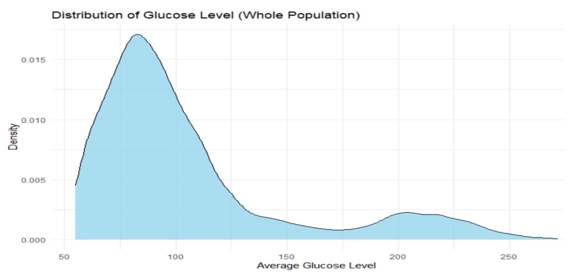

Figure 2.8: Distribution of Glucose Level (Whole population)

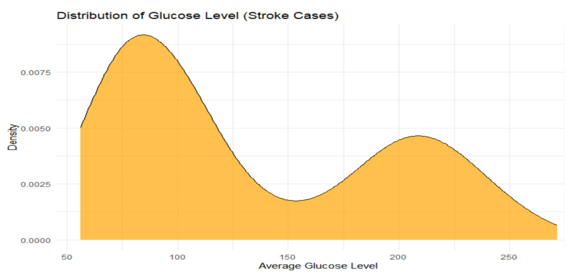

Figure 2.9: Distribution of Glucose Level (Stroke Cases)

Observations: From these distributions it looks like the 2.9 is more concentrated on high values than 2.8, while keeping always the maximum peak slightly below 100. Moreover, the yellow density shows a maximum peak which is almost half of the former one. So it seems that the distribution of average glucose level might be useful to detect something related to strokes.
Since the distribution is far from being assumed normal, no t-test can be performed.

2.9 Can we predict the likelihood of strokes based on demographic factors and lifestyle choices?

2.9.1 Brief theoretical introduction to GLM

In this analysis, a vector **Y** comprising dichotomous response variables (1 or 0) is considered, it seems therefore reasonable to assume a Bernoulli distribution for the response variable.
The assumption of the model tells that each component can be modelled according to the law:

$$g(\pi_i) = \log\left(\frac{\pi_i}{1-\pi_i}\right) = X_i\beta$$

Where π_i is the probability of the observation with the row-vector X_i. Which, expressed in matrix form, becomes: $\mathbf{g}(\pi) = X\beta$.
The focus is on understanding the relationship between the mean of the variables Y_i and the coefficient vector β based on the specified relationship, while considering the covariate vector X_i, which may not contain all available covariates but could include selected ones.
The goal is to estimate the generic Y_i using its expected value, corresponding to the estimated success probability $\hat{\pi}_i$.

2.9.2 Model fitting

The model just described is called **Logistic regression model**, and the code to implement it in R is the following:

```
# Fit a logistic regression model
stroke_model <- glm(stroke ~ age + hypertension + heart_disease + avg_glucose_
    level + bmi +
                      genderMale + ever_marriedYes + Residence_typeUrban +
                      work_typeGovt_job + work_typeNever_worked + work_typePrivate
                           + work_typeSelfemployed +
                      smoking_statusneversmoked + smoking_statussmokes,
                    data = encoded_df, family = "binomial")
```

Listing 2.10: R-Code: GLM - Model fitting

Output:

```
Call:
glm(formula = stroke ~ age + hypertension + heart_disease + avg_glucose_level +
    bmi + genderMale + ever_marriedYes + Residence_typeUrban +
    work_typeGovt_job + work_typeNever_worked + work_typePrivate +
    work_typeSelfemployed + smoking_statusneversmoked + smoking_statussmokes,
    family = "binomial", data = encoded_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.1324  -0.3204  -0.1645  -0.0883   3.6231

Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                -6.658608   0.764597  -8.709  < 2e-16 ***
age                         0.073832   0.005777  12.781  < 2e-16 ***
hypertension                0.413096   0.164248   2.515 0.011901 *
heart_disease               0.284988   0.190826   1.493 0.135322
avg_glucose_level           0.004174   0.001201   3.474 0.000512 ***
bmi                        -0.005734   0.010948  -0.524 0.600446
genderMale                  0.010115   0.141770   0.071 0.943119
ever_marriedYes            -0.177743   0.225369  -0.789 0.430302
Residence_typeUrban         0.082754   0.138442   0.598 0.550007
work_typeGovt_job          -0.815623   0.823309  -0.991 0.321850
work_typeNever_worked     -10.277387 308.949764  -0.033 0.973463
work_typePrivate           -0.672831   0.807555  -0.833 0.404748
work_typeSelfemployed      -1.050083   0.828468  -1.267 0.204977
smoking_statusneversmoked  -0.212949   0.158308  -1.345 0.178575
smoking_statussmokes        0.039388   0.187976   0.210 0.834027

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 1990.3  on 5108  degrees of freedom
Residual deviance: 1581.0  on 5094  degrees of freedom
AIC: 1611

Number of Fisher Scoring iterations: 14
```

Listing 2.11: R-Code: GLM - Model output

2.9.3 STEP algorithm for features pruning

It is possible to implement a procedure to minimize in a greed way the AIC, details follow:
The step() function in R, by default, aims to minimize the AIC (Akaike Information Criterion) value during the stepwise model selection process. The goal is to find the model with the lowest AIC, not the highest AIC.
The stepwise model selection algorithm implemented in step() typically starts with a full or initial model that includes all predictors. It then iterativly considers adding or removing variables one at a time, evaluating the AIC for each modification.
The process followed by step() can be summarized as follows:

- Fit the complete model that includes all predictors.
- For each predictor, fit another model that excludes that predictor.
- Compare the AIC of all these models, including the one with no predictors.
- Select the model with the lowest AIC among all the considered models.
- Repeat steps 2 to 4 until the best model is found, typically indicated by having the highest AIC score.

The goal is to find a model with the best trade-off between goodness of fit and model complexity, as reflected in the lowest AIC value.
In summary, the correct interpretation is that the step() function in R aims to minimize the AIC during stepwise model selection, searching for the model with the lowest AIC value.
Notice how this procedure can lead to sub-optimal models, since it doesn't try all possible predictors combinations, but rather finds a greedy solution to this search.

```
step_stroke <- step(stroke_model)
summary(step_stroke)
```

Listing 2.12: R-Code: STEP algorithm implementation

Output: (just the simplest model found)

```
glm(formula = stroke ~ age + hypertension + heart_disease + avg_glucose_level +
    work_typeSelfemployed + smoking_statusneversmoked, family = "binomial",
    data = encoded_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.1498  -0.3251  -0.1692  -0.0788   3.8301

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)               -7.479882   0.367428 -20.357  < 2e-16 ***
age                        0.071965   0.005364  13.417  < 2e-16 ***
hypertension               0.406570   0.162962   2.495 0.012600 *
```

```
heart_disease                0.306235   0.188478   1.625 0.104209
avg_glucose_level            0.004004   0.001165   3.438 0.000587 ***
work_typeSelfemployed       -0.340722   0.161501  -2.110 0.034883 *
smoking_statusneversmoked   -0.231167   0.139201  -1.661 0.096780 .

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1990.3  on 5108  degrees of freedom
Residual deviance: 1584.1  on 5102  degrees of freedom
AIC: 1598.1

Number of Fisher Scoring iterations: 7
```

Listing 2.13: R-Code: STEP algorithm output

Observations: This iterative method confirms what has been said in the preliminary analysis conducted on single features, indeed the most important information available are the ones linked with age, hypertension, heart-health and level of glucose together with the work situation and smoking status. It was crucial to perform such analysis because, sometimes, the conclusions inferred by simply checking one-by-one each feature against the target may be biased due to unwanted linkage between features themselves, as in this case happens with the marital status, which comes up to be negligible, maybe biased from the age.

2.9.4 Conclusions

Yes, we can predict the likelihood of strokes based on demographic factors and lifestyle choices. The coefficients just found are, indeed, the tool used to make the prediction of the probability, once it is given the vector of features, for an individual i to get a stroke.
In formulas:

$$\hat{\pi}_i = \frac{e^{\beta_0+\beta_1 x_{i,1}+\ldots+\beta_p x_{i,p}}}{1+e^{\beta_0+\beta_1 x_{i,1}+\ldots+\beta_p x_{i,p}}}$$

Where:
$\hat{\pi}_i$ is the probability of getting a stroke for the individual i given as input;
$x_{i,1}, ..., x_{i,p}$ are the features of the instance i ;
$\beta_0, ..., \beta_p$ are the coefficients estimated in the glm-model shown above.

2.9.5 Predictions

Working with generalized linear models, we can choose whether to get the logit estimate

$$g(\mu) = \eta = X\hat{\beta}$$

or the response probabilities, which is simply the inverse of the logit.
This means that we can choose, for example in the case of the Bernoulli distribution, to obtain $g(\pi)$ or to obtain π.
Remind that in the binomial GLM stands: $g(\pi) = \log(\frac{\pi}{1-\pi})$.
Here it is printed out the (first six) probabilities of the instances in the dataset to have a stroke, based on their features:

```
head(stroke_model$fitted.values)
```

Listing 2.14: R-Code: fitted values

Output:

```
         1          2          3          4          5          6
0.19157284 0.05306559 0.22017030 0.03726381 0.22022831 0.30472400
```

Listing 2.15: R-Code: fitted values output

It follows the distribution of the probabilities predicted:

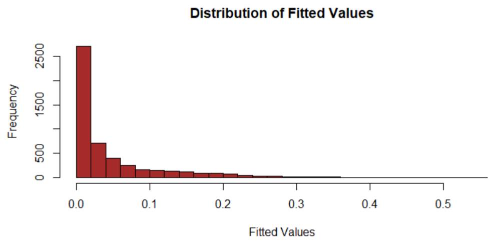

Figure 2.10: Distribution of the prediction

2.9.6 Model evaluation

Let's evaluate the model by setting a threshold above which the target variable is predicted as 1, otherwise 0. Then there will be performed some evaluation metrics for the results obtained.

```
# Set threshold
threshold <- 0.40

# Create binary predictions based on the threshold
binary_predictions <- ifelse(stroke_model$fitted.values > threshold, 1, 0)

# Compute evaluation metrics
confusion_matrix <- table(encoded_df$stroke, binary_predictions)
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
precision <- confusion_matrix[2, 2] / sum(confusion_matrix[, 2])
recall <- confusion_matrix[2, 2] / sum(confusion_matrix[2, ])
f1_score <- 2 * precision * recall / (precision + recall)

# Print evaluation metrics
```

```
cat("Accuracy:", accuracy, "\n")
cat("Precision:", precision, "\n")
cat("Recall:", recall, "\n")
cat("F1 Score:", f1_score, "\n")

# Get the number of predicted values as 1
num_predicted_as_1 <- sum(binary_predictions == 1)

# Print the number of predicted values as 1
cat("Number of predicted stroke as 1:", num_predicted_as_1, "\n")
```

Listing 2.16: R-Code: Model evaluation

Output:

```
Accuracy: 0.9512625
Precision: 0.5
Recall: 0.01606426
F1 Score: 0.0311284
Number of predicted stroke as 1: 8
```

Listing 2.17: R-Code: Model evaluation output

To better appreciate the model now will be proposed an analysis focused on the precision evaluation metric, since there is a strong imbalance ”accuracy” is not the right metric to look at. Namely, it will plot the trend of the precision metric at varying the threshold, while keeping track also of the number of people predicted as suffering from stroke.

```
# Define the thresholds
thresholds <- c(0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50)

# Initialize vectors to store precision values and number of predicted values as 1
precisions <- numeric(length(thresholds))
num_predicted_as_1 <- numeric(length(thresholds))

# Loop over each threshold
for (i in seq_along(thresholds)) {
  # Create binary predictions based on the threshold
  binary_predictions <- ifelse(stroke_model$fitted.values > thresholds[i], 1, 0)

  # Compute precision
  confusion_matrix <- table(encoded_df$stroke, binary_predictions)
  precision <- confusion_matrix[2, 2] / sum(confusion_matrix[, 2])

  # Store precision value and number of predicted values as 1
  precisions[i] <- precision
  num_predicted_as_1[i] <- sum(binary_predictions == 1)
}

# Plot precision against thresholds
plot(thresholds, precisions, type = "b",
     xlab = "Threshold (Predicted as 1)", ylab = "Precision",
     main = "Precision vs. Threshold",
     col = "blue", pch = 19, cex = 1.5, lwd = 2)

# Add number of predicted values as 1 below each threshold on x-axis labels
mtext(text = paste("\n\n", num_predicted_as_1), side = 1, at = thresholds, line =
    2)
```

```r
# Add grid lines
grid()

# Add a legend
legend("topright", legend = "Precision", col = "blue", pch = 19, lwd = 2)
```

Listing 2.18: R-Code: Precision analysis implementation

Output:

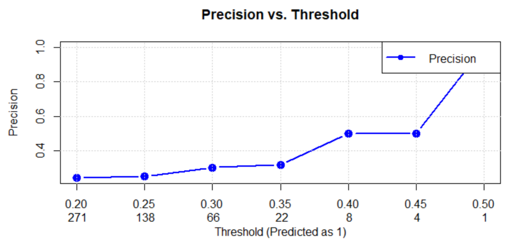

Figure 2.11: Precision's trend

Bibliography

Hassan, Ahmad (2023). *Stroke Prediction Dataset*. `https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset`.