

Statistical Learning Project

Filippo Santin, Gurjeet Singh, Francesca Zen

18/5/2021

1 Introduction

In the following report, we present an analysis computed on stroke disease in which we try to explain from statistical analysis some correlation factors and statistics of the given features by designing models to predict the presence of stroke disease. In addition, we will highlight possible linear and non-linear relationships among the given features (predictors) and the stroke disease variable (predicted variable).

“Stroke” is the medical term for damage to brain tissue or the death of a portion of it, due to insufficient blood supply to an area of the brain. According to the World Health Organization (WHO) stroke is the 2nd leading cause of death globally, responsible for approximately 11% of total deaths and our aim is to see if and how the variables we are dealing with are related, in order to predict which individual is more probable to have a stroke.

The symptoms of stroke vary from patient to patient, depending on the severity of the condition, the affected brain area, causes, type of stroke, etc. Stroke is characterized by sudden onset and for this reason it involves the need for immediate therapeutic intervention and adapted to the needs of the patient. In this sense, looking for relation between features may help to prevent or assess it.

In order to have a guide for the interpretation of the data we underline the following information:

- The normal values of glucose level are between 60 and 110 mg/dl and with a value greater than 126 mg/dl a person is considered diabetic;
- a body mass index (BMI) between 18.5-24.9 indicates a normal/healthy weight, below 18.5 indicates underweight, 25.0-29.9 indicates overweight and above 30.0 indicates obese person.

2 Exploring the Dataset

The dataset we used is provided by Kaggle ¹ and it is composed of 5,110 entries with a total of 12 columns: `id`, `gender`, `age`, `hypertension`, `heart_disease`, `ever_married`, `work_type`, `Residence_type`, `avg_glucose_level`, `bmi`, `smoking_status`, `stroke`.

```
library(knitr)
stroke_data <- read.csv('healthcare-dataset-stroke-data.csv')
kable(stroke_data[1:5], format = 'simple', align='ccccccccc',
      col.names = c('id','gender','age', 'hypert.', 'hd' , 'ev_marr',
      'work_type','res_type','glucose', 'bmi','smoking','stroke'))
```

id	gender	age	hypert.	hd	ev_marr	work_type	res_type	glucose	bmi	smoking	stroke
9046	Male	67	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
51676	Female	61	0	0	Yes	Self-employed	Rural	202.21	N/A	never smoked	1
31112	Male	80	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
60182	Female	49	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
1665	Female	79	1	0	Yes	Self-employed	Rural	174.12	24	never smoked	1

¹<https://www.kaggle.com/fedesoriano/stroke-prediction-dataset>

2.1 Preprocessing

The preliminary part of the analysis focuses on the study of the dataset and its pre-processing: we looked at the `id` column and verified that all the data collected were referring to different people, thus no recidivist status were involved.

After this check we removed the column from the dataset since it did not hold useful information for our study.

```
stroke_data <- stroke_data[,-1]
```

In order to use the variables through the analysis we transformed the categorical variables into factors:

```
stroke_data$gender <- as.factor(stroke_data$gender)
stroke_data$ever_married <- as.factor(stroke_data$ever_married)
stroke_data$work_type <- as.factor(stroke_data$work_type)
stroke_data$Residence_type <- as.factor(stroke_data$Residence_type)
stroke_data$smoking_status <- as.factor(stroke_data$smoking_status)
stroke_data$heart_disease<-as.factor(stroke_data$heart_disease)
```

The variable `bmi` was not numeric because of the presence of “N/A” string values which identify missing information, hence we transformed its elements into numerical values and then removed the NA values generated.

```
stroke_data$bmi <- as.numeric(stroke_data$bmi)

## Warning: NA introdotti per coercizione
stroke_data <- na.omit(stroke_data)
```

We ended up having 4,909 entries and 11 total columns.

Here we give a quick overview of the main information about the dataset:

```
summary(stroke_data)
```

```
##      gender          age      hypertension      heart_disease ever_married
## Female:2897   Min.   : 0.08   Min.   :0.00000   0:4666       No :1705
##  Male :2011    1st Qu.:25.00  1st Qu.:0.00000   1: 243        Yes:3204
##  Other :     1   Median :44.00  Median :0.00000
##                      Mean   :42.87  Mean   :0.09187
##                      3rd Qu.:60.00  3rd Qu.:0.00000
##                      Max.   :82.00  Max.   :1.00000
##      work_type      Residence_type avg_glucose_level      bmi
## children       : 671   Rural:2419      Min.   :55.12   Min.   :10.30
## Govt_job       : 630   Urban:2490     1st Qu.:77.07   1st Qu.:23.50
## Never_worked  : 22    Median :91.68   Median :28.10
## Private        :2811    Mean   :105.31   Mean   :28.89
## Self-employed  : 775    3rd Qu.:113.57  3rd Qu.:33.10
##                      Max.   :271.74  Max.   :97.60
##      smoking_status      stroke
## formerly smoked: 837   Min.   :0.00000
## never smoked    :1852   1st Qu.:0.00000
## smokes          : 737   Median :0.00000
## Unknown         :1483   Mean   :0.04257
##                      3rd Qu.:0.00000
##                      Max.   :1.00000
```

2.2 Descriptive Statistic

```
attach(stroke_data)
```

In order to highlight and analize better the data, we used some plots to study their statistics and distribution. A relevant and important information is provided by the following barplot, in which we see an unbalance dataset issue: 209 people on a total of 4909 get a stroke, i.e. the 4.26% of the people.

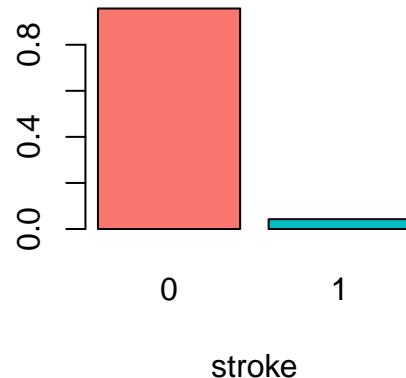
```
table(stroke)
```

```
## stroke
##    0    1
## 4700 209
```

```
table(stroke)/dim(stroke_data) [1]
```

```
## stroke
##      0         1
## 0.95742514 0.04257486
```

```
barplot(table(stroke)/dim(stroke_data) [1], xlab='stroke',
       col = c('#F8766D', '#00BFC4'))
```



These values are representative of the real situation in which there are not many stroke cases compared with the whole population, the people who have had a stroke are much less than the ones who did not have it. The incidence of stroke in Europe at the beginning of the 21st century varies from 95 to 290 cases/100,000². Furthermore in many clinical disease analyses this issue is commonly present.

A visual transformation of the values seen in the `summary` function is provided in the following boxplots:

```
par(mfrow=c(1,3))
boxplot(avg_glucose_level, xlab= 'average glucose level' , col='#00BA38')
boxplot(bmi, xlab = 'body mass index', col='#00BA38')
boxplot(age, xlab = 'age', pch=20, col='#00BA38')
```

```
par(mfrow=c(1,1))
```

²QUADERNI dell'Italian Journal of Medicine, A Journal of Hospital and Internal Medicine, Michele Meschi, volume 8, issue 2, March-April 2020

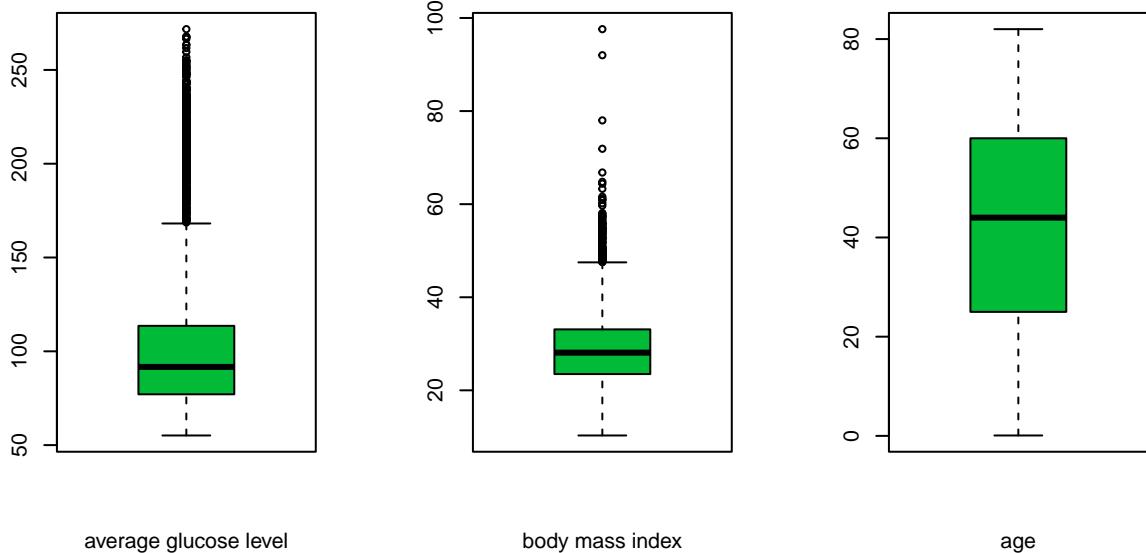


Figure 1: Visual description of some features of the dataset

From Figure 1 we can see that in the first two boxplots (starting from the left) there are lots of outliers, whose range can also be seen from the summary of the `avg_glucose_level` and `bmi` variables and applying the following formula: $\max_value - q_{3/4} * 1.5 \text{ IQR}$, where $q_{1/4}, q_{3/4}$ are the first and third quantile, and $\text{IQR} = q_{3/4} - q_{1/4}$ is the interquartile range. So we have a range of outliers of

$$271 - (113.57 + 1.5 * (113.57 - 77.07)) = 102.68$$

for the glucose, and a range of

$$97.60 - (33.10 + 1.5 * (33.10 - 23.50)) = 50.10$$

for the body mass index.

Actually, they represent real-case scenarios (people affected by high glucose levels or with a bmi out of range are only a few) and possible interesting cases of pathologies bounded with diabetes. Hence these data points have to be considered during modeling, they could be helpful to predict stroke cases because it could be due to complications of diabetes, as mentioned in the medical literature.

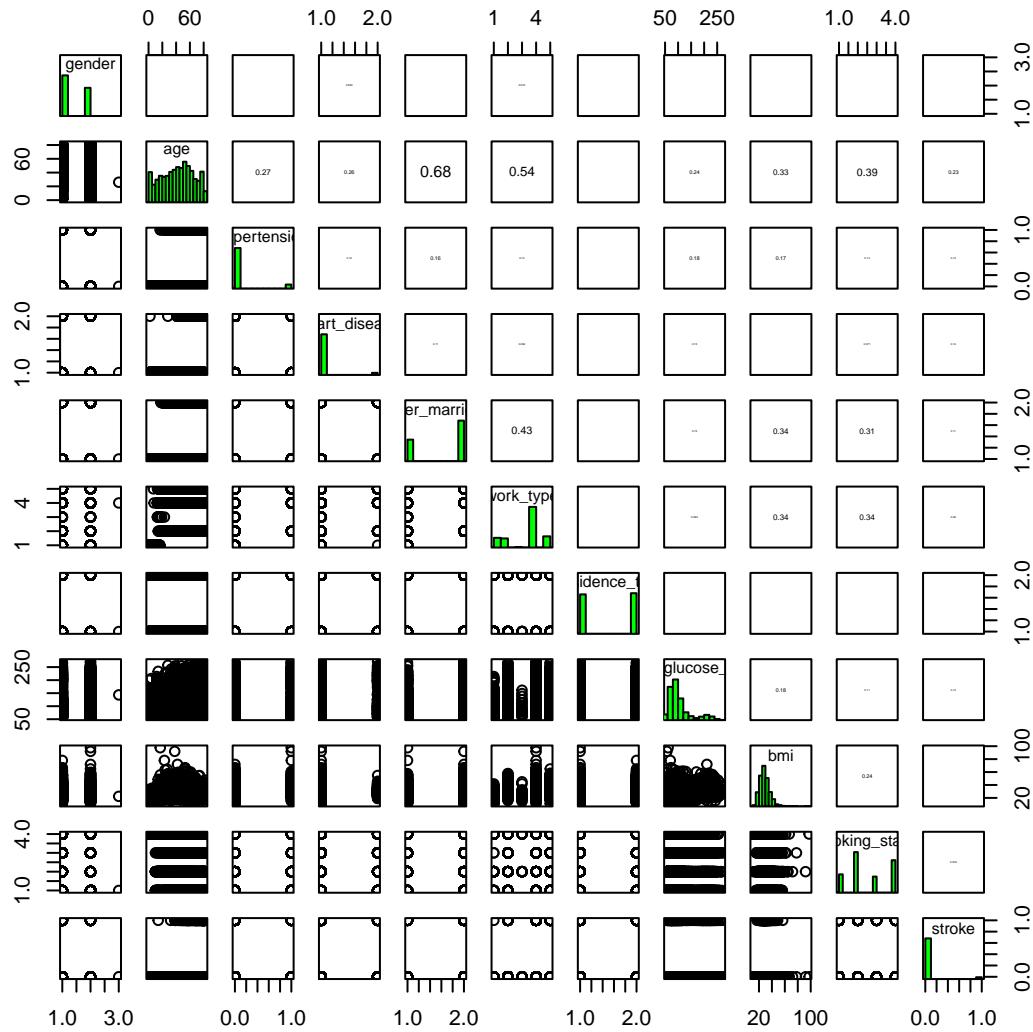
In order to compare features in pairs and judge which of each one is preferred, or has a greater amount of some quantitative property we provide a pair-wise plot. In addition, to involve also the categorical variables we wrote some useful functions:

```
panel.cor <- function(x, y, digits = 2, prefix = "", cex.cor, ...){
  usr <- par("usr"); on.exit(par(usr))
  par usr = c(0, 1, 0, 1)
  r <- abs(cor(x, y))
  txt <- format(c(r, 0.123456789), digits = digits)[1]
  txt <- paste0(prefix, txt)
  if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)
  text(0.5, 0.5, txt, cex = cex.cor * r)
}
panel.hist <- function(x, ...)
```

```
{
  usr <- par("usr"); on.exit(par(usr))
  par(usr = c(usr[1:2], 0, 1.5) )
  h <- hist(x, plot = FALSE)
  breaks <- h$breaks; nB <- length(breaks)
  y <- h$counts; y <- y/max(y)
  rect(breaks[-nB], 0, breaks[-1], y, col = "green", ...)
}
```

And here we show the results from the pairs plot:

```
pairs(stroke_data, diag.panel = panel.hist, upper.panel = panel.cor)
```



The pairs-plot shows that the stronger relationships involve quite often the variable `age`. But there are other small relevant relations, for example between `work_type` and `ever_married` or between `bmi` and `work_type`. In addition, we can see strong collinearity among the variables `age`, `work_type`, and `ever_married`, this means that they are closely related to each another. For this reason we get uncertainty in the coefficient estimates and so we do not consider them together in the fitting of the models.

We go on looking at some intuitive relationship of `stroke` with `age`, `bmi` and `avg_glucose_level`:

```
par(mfrow=c(1,3))
boxplot(avg_glucose_level~stroke, xlab = 'stroke',
        ylab = 'average glucose level', col = c('#F8766D','#00BFC4'))
boxplot(bmi~stroke, xlab = 'stroke', ylab = 'bmi', col = c('#F8766D','#00BFC4'))
boxplot(age~stroke, xlab = 'stroke' , ylab = 'age', col = c('#F8766D','#00BFC4'))
```

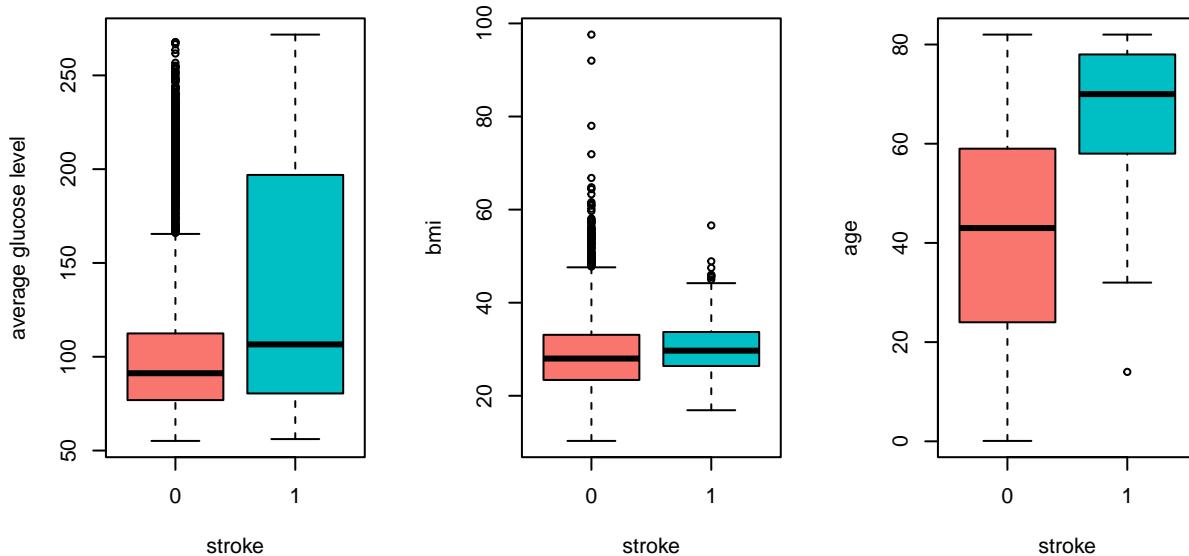


Figure 2: Visual representation of the comparison between `stroke`(1) and not-`stroke`(0) of the features: `avg_glucose_level`, `bmi` and `age`.

```
par(mfrow=c(1,1))
```

Looking at Figure 2 we can see that the incidence of the disease increases progressively with age, in particular the summary of `age` shows that the data observed cover people of all ages from babies of 8 days to seniors of 82 years old and this can also be seen in the boxplot. We get also the information that the youngest person affected of stroke disease is 14 years old (an outlier), while the oldest one is 82 years old.

We make a more detailed analysis looking at the following tables:

```
table(stroke.less.35 <- stroke_data[stroke_data$age<35, 'stroke'])

##
##      0      1
## 1796     2

table(stroke.35.50 <- stroke_data[stroke_data$age>=35 & stroke_data$age<50 , 'stroke'])

##
##      0      1
## 1005    16

table(stroke.major.50 <- stroke_data[stroke_data$age>=50 , 'stroke'])

##
##      0      1
```

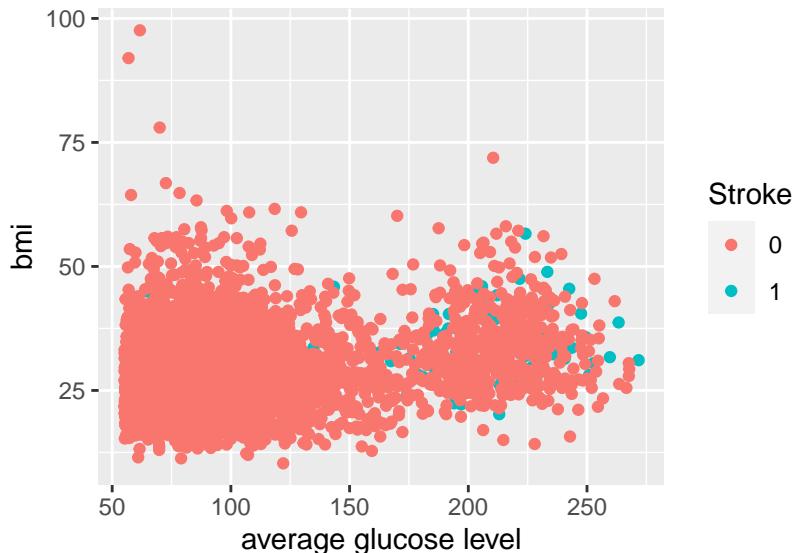
```
## 1899 191
```

In `stroke.less.35` we can notice that there are only two people under the age of 35 which suffered a stroke illness, while in `stroke.35.50` we highlight the number of people between the age of 35 and 50 that are ill, i.e. 16. The major difference in the numbers can be seen in `stroke.major.50` where stroke patients older than 50 are much more than the one of the previous cases.

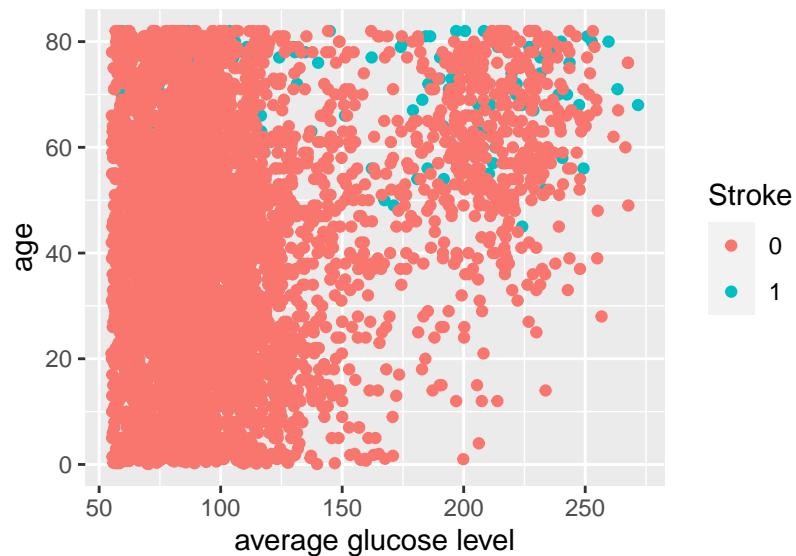
In Figure 2, if we sum to `age` the information about `avg_glucose_level` we may wonder if diabetic people are more probable to get a stroke or not (first boxplot from the left). On the other hand, there is no apparent relation of `stroke` with `bmi` (central boxplot).

We now highlight other visual relationship between the variables used before:

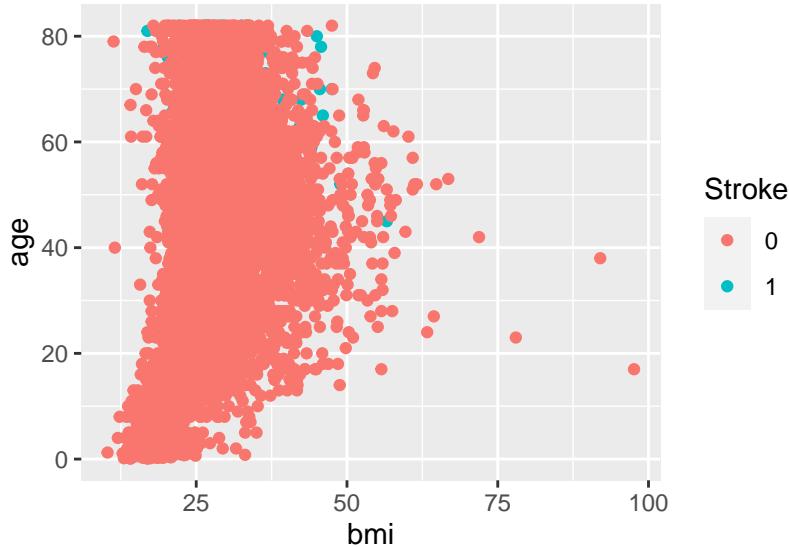
```
library(ggplot2)
ggplot(stroke_data, aes(x = avg_glucose_level, y = bmi, col = as.factor(stroke))) +
  labs(x = "average glucose level", y = "bmi", color = "Stroke") + geom_point()
```



```
ggplot(stroke_data, aes(x = avg_glucose_level, y = age, col = as.factor(stroke))) +
  labs(x = "average glucose level", y = "age", color = "Stroke") + geom_point()
```



```
ggplot(stroke_data, aes(x = bmi, y = age, col = as.factor(stroke))) +
  labs(x = "bmi", y = "age", color = "Stroke") + geom_point()
```



What we can see from these scatter plots is that if we have low `avg_glucose_level` values in relation with `age` or `bmi`, the stroke disease tends to be not present. On the other hand, if we look at the `bmi` predictor we can notice that there are samples of stroke cases spread all over in all of its range of values, so it highlights no clear correlation of `bmi` with stroke, even when we put this features in relation with the others.

`avg_glucose_level` and `bmi` could not be so strictly related to the disease but maybe correlated to other illnesses linked (or not) to it.

In the end we can see that it is not easy to identify the direct relationship with the stroke while dealing with the features that we have, and it is difficult a further interpretation of the data.

2.3 Data Science Questions

At this point we can ask some questions:

- Which factors are the most related to the stroke disease?
- How strong are the relations between the features?
- Are the given variables enough to predict a good accuracy of some possible person affected by stroke?
- Is it possible to prevent it?

We will explore the data trying to answer these questions.

3 Modeling

In order to explore different approaches for predicting the qualitative response `stroke`, in the following paragraph we will cover three of the most widely-used classifiers: *logistic regression*, *linear discriminant analysis (LDA)* and *quadratic discriminant analysis (QDA)*. The prediction process is known as *classification*.

3.1 Logistic Regression

Logistic regression can be defined as a type of *generalized linear model (GLM)*. In particular, it is a statistical model used to examine the association of categorical or continuous independent variables with one dichotomous dependent variable.

Rather than computing directly the response Y , logistic regression computes the probability that Y belongs to a particular category, in our case 1 if the person has a stroke disease, 0 otherwise.

In this part of the predictive analysis we will present three different type of models, compared to discover the best one that can better interpret the data. In particular, our model selection will be guide both from the p-values and probabilistic statistical measure that attempts to quantify both the model performance on the training dataset and the complexity of the model. An example of this last strategy is given by the *Akaike Information Criterion (AIC)*.

Compared with the *BIC* method, the *AIC* statistic penalizes complex model less, meaning that it may put more emphasis on model performance on the training dataset, and, in turn, select more complex models. Its description is given by the following formula

$$AIC = -2 \cdot \ell(\hat{\Theta}) + 2d,$$

where $\ell(\hat{\Theta})$ is the maximized value of the log-likelihood function for the estimated model and d the number of predictors. We decided to exclude the Mallow's C_p and the R^2 adjusted techniques because they work efficiently in the case of linear regression.

3.1.1 Full and Reduced Models

We start with the full model to see if all the features of the dataset contribute on the prediction of a stroke. Up to now we have already mentioned if some predictors seem to be more or less correlated to the response variable, and a way to analyze such relation is by introducing the *null hypothesis*. In the case of the full model, we are dealing with 10 explanatory variables and so we have

$$H_0 : \beta_1 = \cdots = \beta_{10} = 0$$

and this means that we are making the assumption that no predictors is somehow related with `stroke`. If we reject the null hypothesis we are assuming that there is in fact a correlation and we have to find which feature is involved.

To decide whether or not reject H_0 we set the alpha value to $\alpha = 0.1$ in order to decide whether a test statistic is statistically significant, thus for p-values smaller than α we keep the feature otherwise we remove it from the model.

Let's have a look now at the characteristics of the full model:

```
mod.full <- glm(stroke~., data = stroke_data, family = binomial)
summary(mod.full)
```

```
##
## Call:
## glm(formula = stroke ~ ., family = binomial, data = stroke_data)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.1823   -0.2947   -0.1524   -0.0744    3.5251
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -7.360e+00  1.067e+00 -6.895 5.37e-12 ***
## genderMale                  -1.463e-02  1.544e-01 -0.095 0.924525
## genderOther                 -1.135e+01  2.400e+03 -0.005 0.996225
## age                          7.348e-02  6.347e-03 11.578 < 2e-16 ***
## hypertension                 5.249e-01  1.750e-01  2.999 0.002711 **
## heart_disease1              3.488e-01  2.072e-01  1.683 0.092381 .
## ever_marriedYes             -1.152e-01  2.473e-01 -0.466 0.641394
## work_typeGovt_job            -6.817e-01  1.114e+00 -0.612 0.540660
## work_typeNever_worked       -1.082e+01  5.090e+02 -0.021 0.983036
## work_typePrivate              5.208e-01  1.100e+00 -0.473 0.635943
## work_typeSelf-employed       -9.459e-01  1.119e+00 -0.845 0.397906
```

```

## Residence_typeUrban      4.514e-03  1.500e-01  0.030  0.975990
## avg_glucose_level       4.652e-03  1.294e-03  3.595  0.000324 ***
## bmi                      4.062e-03  1.188e-02  0.342  0.732387
## smoking_statusnever smoked -6.722e-02  1.886e-01 -0.356  0.721556
## smoking_statussmokes     3.139e-01  2.295e-01  1.368  0.171310
## smoking_statusUnknown    -2.753e-01  2.471e-01 -1.114  0.265193
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1728.4  on 4908  degrees of freedom
## Residual deviance: 1363.2  on 4892  degrees of freedom
## AIC: 1397.2
##
## Number of Fisher Scoring iterations: 15

```

Here we see that `age`, `avg_glucose_level` and `hypertension` are the variables most related to `stroke`, with a p-value smaller than 0.005, but also `heart_disease` contributes a bit to the model with a p-value of 0.092381. As expected, the collinearity factors, i.e. `work_type`, `Residence_type` and `ever_married`, do not contribute.

Let's use the residual plots to get more information about this model.

```

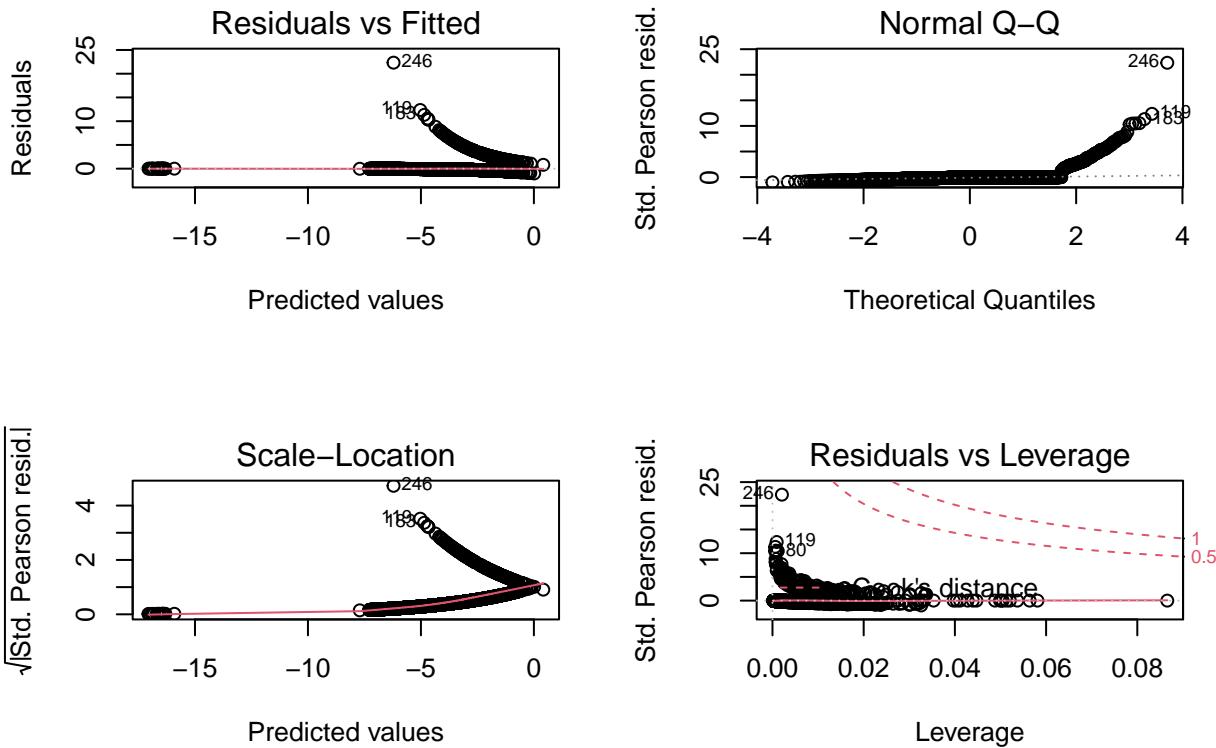
par(mfrow=c(2,2))
plot(mod.full)

```

```

## Warning: not plotting observations with leverage one:
##      2971

```



```
par(mfrow=c(1,1))
```

The plots are not satisfactory because they are not easy to interpret. It is simply evident that data do not follow a linear trend (we will discuss more in detail in the next paragraphs).

We now go on with some tests, we take the full model and we remove all the predictors that show up collinearity between each other, following a sort of greedy backward selection:

```
mod.red1 <- glm(stroke ~ age + bmi + avg_glucose_level + hypertension +
                  smoking_status + gender + heart_disease, family = binomial)
summary(mod.red1)
```

```
##
## Call:
## glm(formula = stroke ~ age + bmi + avg_glucose_level + hypertension +
##       smoking_status + gender + heart_disease, family = binomial)
##
## Deviance Residuals:
##    Min      1Q   Median      3Q     Max
## -1.0751 -0.2957 -0.1572 -0.0734  3.6706
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -7.823946   0.588445 -13.296 < 2e-16 ***
## age                      0.069041   0.005847  11.808 < 2e-16 ***
## bmi                      0.003458   0.011745   0.294  0.768441
## avg_glucose_level        0.004697   0.001289   3.644  0.000269 ***
## hypertension              0.517649   0.174438   2.968  0.003002 **
##
```

```

## smoking_statusnever smoked -0.057792  0.187972 -0.307 0.758502
## smoking_statussmokes      0.321264  0.228512  1.406 0.159754
## smoking_statusUnknown     -0.256978  0.245259 -1.048 0.294740
## genderMale                -0.011195  0.154011 -0.073 0.942055
## genderOther               -7.287812 324.743860 -0.022 0.982096
## heart_disease1            0.372836  0.206072  1.809 0.070411 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1728.4  on 4908  degrees of freedom
## Residual deviance: 1369.4  on 4898  degrees of freedom
## AIC: 1391.4
##
## Number of Fisher Scoring iterations: 11

```

Also in this case the more important covariates are the same of the ones found in the full model. In this step we remove the `gender` variable, because it is not significant and we end up with more or less the same results as before.

```
mod.red2 <- glm(stroke ~ age + bmi + avg_glucose_level + hypertension +
                  smoking_status + heart_disease, family = binomial)
summary(mod.red2)
```

```

##
## Call:
## glm(formula = stroke ~ age + bmi + avg_glucose_level + hypertension +
##       smoking_status + heart_disease, family = binomial)
##
## Deviance Residuals:
##    Min      1Q   Median      3Q      Max
## -1.0766 -0.2964 -0.1572 -0.0733  3.6720
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -7.830942  0.582317 -13.448 < 2e-16 ***
## age                      0.069065  0.005842  11.822 < 2e-16 ***
## bmi                      0.003487  0.011744   0.297 0.766535
## avg_glucose_level        0.004689  0.001285   3.648 0.000264 ***
## hypertension              0.517599  0.174427   2.967 0.003003 **
## smoking_statusnever smoked -0.055884  0.186343 -0.300 0.764254
## smoking_statussmokes      0.321811  0.228437  1.409 0.158907
## smoking_statusUnknown     -0.255822  0.244827 -1.045 0.296066
## heart_disease1            0.371145  0.204739  1.813 0.069867 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1728.4  on 4908  degrees of freedom
## Residual deviance: 1369.4  on 4900  degrees of freedom
## AIC: 1387.4
##
## Number of Fisher Scoring iterations: 7
```

The parameters left to delete are `bmi` and `smoking_status` that continue to have a high p-value. Now, we are able to define the final reduced model:

$$\text{stroke} = \beta_0 + \beta_1 \times \text{age} + \beta_2 \times \text{heart_disease} + \beta_3 \times \text{avg_glucose_level} + \beta_4 \times \text{hypertension}$$

with four explanatory variables.

```
mod.red <- glm(stroke~age + avg_glucose_level + heart_disease + hypertension,
                family = binomial)
summary(mod.red)
```

```
##
## Call:
## glm(formula = stroke ~ age + avg_glucose_level + heart_disease +
##      hypertension, family = binomial)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.0995 -0.2940 -0.1599 -0.0778  3.5885
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -7.660740   0.387152 -19.787 < 2e-16 ***
## age                   0.067547   0.005571  12.124 < 2e-16 ***
## avg_glucose_level    0.004802   0.001255   3.828 0.000129 ***
## heart_disease1       0.404298   0.203447   1.987 0.046895 *
## hypertension          0.539613   0.173055   3.118 0.001820 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1728.4 on 4908 degrees of freedom
## Residual deviance: 1374.6 on 4904 degrees of freedom
## AIC: 1384.6
##
## Number of Fisher Scoring iterations: 7
```

An important thing to notice is that there has been a decreasing on the p-value of the explanatory variable `heart_disease`, that now have a p-value of 0.046895.

This means that our probability to get a stroke becomes:

$$p(X) = \frac{e^{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4}}{1 + e^{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4}}$$

where β_0, \dots, β_4 can be read in the `summary` of `mod.red`, as the coefficient estimates, with the respectively variables X_1, \dots, X_4 which represent `age`, `avg_glucose_level`, `heart_disease` and `hypertension`.

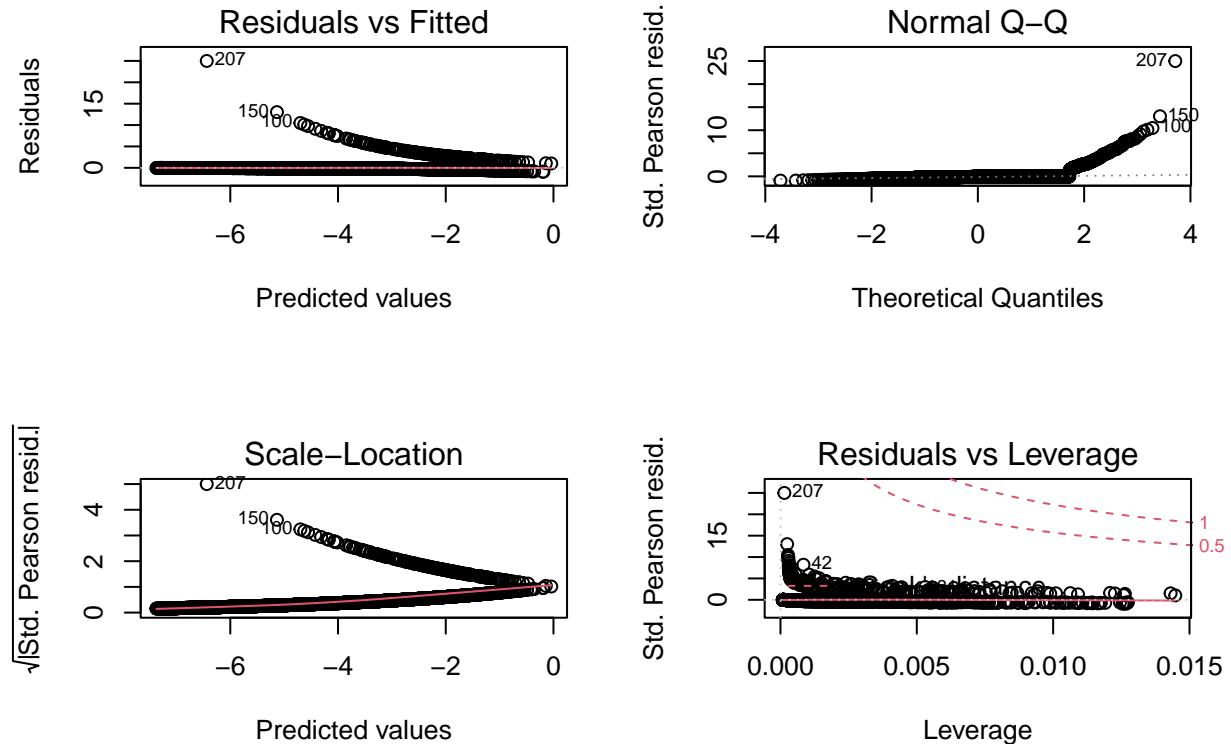
From $p(X)$ we can get the quantity $p(X)/(1 - p(X))$ which is called *odds*, and it can take any value between 0 and ∞ . Values of odds close to 0 and ∞ indicates very low and very high probabilities to get a stroke, respectively.

In our logistic regression model, increasing X_i by one unit multiplies the odds by e^{β_i} , for $i = 1, \dots, 4$. The amount of change in $p(X)$ due to a one-unit change in X_i depends on the current value of X_i . But regardless of the value of X_i , if β_i is positive than increasing X_i will be associated with increasing $p(X)$, and if β_i is negative than increasing X_i will be associated with decreasing $p(X)$, for $i = 1, \dots, 4$.

The fact that there is not a straight-line relationship between $p(X)$ and X_i , and the fact that the rate of change in $p(X)$ per unit change in X_i depends on the current value of X_i .

For this model we also give a descriptive statistic through the residual plots:

```
par(mfrow=c(2,2))
plot(mod.red)
```



```
par(mfrow=c(1,1))
```

Reading and interpreting these graphs, we can state some important outcomes:

Residuals vs Fitted

We can observe the presence of high non-linearity in the dataset. Indeed we can find not spread equal residuals around a horizontal line without distinct patterns, attesting to the fact that there is no linear relationship.

Normal Q-Q

We discover that residuals do not follow a normal distribution.

We know that obtaining positive coefficient estimates, we have a positive association. So, the larger the value the higher is the estimated probability of stroke.

Scale–Location

We can see that homoscedasticity does not hold since the line of the standard residual is not flat, hence even by standardizing the residuals we end up having high variance among them. But in our case, as you can notice from this plot, the red line is slightly curved and the residuals increase as the fitted Y values increases. So, the inference here is, heteroscedasticity exists.

Residual vs Leverage

Looking at the leverage plot, we see the presence of some samples with high leverage value (bottom right), which could influence the prediction of the model (even if the R-plot does not show us its index).

	gender	age	hypert.	hd	ev_marr	work_type	res_type	glucose	bmi	smoking	stroke
119	Female	38	0	0	No	Self-employed	Urban	82.28	24.0	formerly smoked	1
183	Female	32	0	0	Yes	Private	Rural	76.13	29.9	smokes	1
246	Female	14	0	0	No	children	Rural	57.93	30.9	Unknown	1

Moreover, in the above table, there are outliers (with no connection with our regression) that have high variance, and they represent some rare cases of disease development; as we saw previously there was a case of a 14 years old girl that had stroke. If we look at the results, we can affirm that the reduced model seems to make a more accurate prediction and fits better data than the full one. Indeed, its *AIC* is the lower of the two.

To have a confirmation of this concept we use `anova`, a function which allows us to provide a comparison. The null hypothesis is that the two models fit the data equally well, and the alternative hypothesis is that the reduced model is superior.

```
anova(mod.red, mod.full, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: stroke ~ age + avg_glucose_level + heart_disease + hypertension
## Model 2: stroke ~ gender + age + hypertension + heart_disease + ever_married +
##           work_type + Residence_type + avg_glucose_level + bmi + smoking_status
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      4904    1374.7
## 2      4892    1363.2  12     11.42  0.4933
```

As we know, the `anova` table gives us the deviance difference, the degrees of freedom and above all the statistical test. As expected, the `anova` test rejects that the full model is more pertinent than the reduced one, since the p-value of its features are not less than 5% or even 10%. Therefore the full model does not enhance the prediction.

3.1.2 Interaction Models

Let's try to check other convenient models using a mixed approach. We use the reduced model `mod.red` as landmark and we proceed with interactions between the explanatory variables with the aim to understand which variables are relevant on our research and to improve the performance of the model. We remember that we move on with our selection choosing significance levels equal to 0.1 and searching for an *AIC* value less and less.

The model `mod.red` has `stroke` as response and `age`, `avg_glucose_level`, `hypertension` and `heart_disease` as predictors. We recall that its *AIC* is 1384.6.

Initially we consider the interaction of `age` with the other numerical features, i.e. `avg_glucose_level`, `hypertension` and `heart_disease` and we achieve that only `age*heart_disease` gives a relevant contribution. We get *AIC* = 1384.

```
mod1 <- glm(stroke ~ age + avg_glucose_level + heart_disease + hypertension +
             age*heart_disease, family=binomial)
summary(mod1)

##
## Call:
## glm(formula = stroke ~ age + avg_glucose_level + heart_disease +
##       hypertension + age * heart_disease, family = binomial)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.000000 -0.999999 -0.999999 -0.999999  1.000000
```

```

## -0.9897 -0.2980 -0.1557 -0.0737  3.6232
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -7.816578  0.407175 -19.197 < 2e-16 ***
## age                  0.070133  0.005889  11.908 < 2e-16 ***
## avg_glucose_level    0.004702  0.001253   3.752 0.000176 ***
## heart_disease1       2.765299  1.396557   1.980 0.047694 *
## hypertension          0.536550  0.172602   3.109 0.001880 **
## age:heart_disease1 -0.032872  0.019486  -1.687 0.091604 .
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1728.4  on 4908  degrees of freedom
## Residual deviance: 1372.0  on 4903  degrees of freedom
## AIC: 1384
##
## Number of Fisher Scoring iterations: 7

```

Furthermore we can see that the interaction variable is statistically significant for an alpha value set to 0.1, hence we can say that the two slopes for the two different groups, affected by the heart disease and not, are significantly different to each other. Last but not least, we saw that by considering the interaction we have also a high increment in the beta value of the heart disease.

We consider now all the interaction of `avg_glucose_level` with the remaining predictors and find out that `avg_glucose_level*hypertension` is the best of the possible relations even if it can not improve the previous model.

It has an *AIC* of 1385.9.

```

mod2 <- glm(stroke ~ age + avg_glucose_level + heart_disease + hypertension +
            avg_glucose_level*hypertension, family=binomial)
summary(mod2)

```

```

##
## Call:
## glm(formula = stroke ~ age + avg_glucose_level + heart_disease +
##     hypertension + avg_glucose_level * hypertension, family = binomial)
##
## Deviance Residuals:
##      Min        1Q        Median         3Q        Max
## -1.0453  -0.2958  -0.1590  -0.0775   3.5977
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -7.731877  0.395629 -19.543 < 2e-16 ***
## age                   0.067246  0.005586  12.038 < 2e-16 ***
## avg_glucose_level    0.005525  0.001484   3.723 0.000197 ***
## heart_disease1        0.401439  0.203302   1.975 0.048314 *
## hypertension          0.877774  0.413934   2.121 0.033958 *
## avg_glucose_level:hypertension -0.002413  0.002710  -0.890 0.373262
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)

```

```

## Null deviance: 1728.4 on 4908 degrees of freedom
## Residual deviance: 1373.9 on 4903 degrees of freedom
## AIC: 1385.9
##
## Number of Fisher Scoring iterations: 7

Ultimately, we have the interaction heart_disease*hypertension which shows a good model with  $AIC = 1384.5$ .

mod3 <- glm(stroke ~ age + avg_glucose_level + heart_disease + hypertension +
            heart_disease*hypertension, family = binomial)
summary(mod3)

## 
## Call:
## glm(formula = stroke ~ age + avg_glucose_level + heart_disease +
##     hypertension + heart_disease * hypertension, family = binomial)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.0079   -0.2951   -0.1584   -0.0774    3.5900
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -7.657815  0.387787 -19.747 < 2e-16 ***
## age                      0.067108  0.005592  12.001 < 2e-16 ***
## avg_glucose_level         0.004760  0.001256   3.791 0.000150 ***
## heart_disease1            0.592399  0.236015   2.510 0.012073 *
## hypertension                0.660347  0.189663   3.482 0.000498 ***
## heart_disease1:hypertension -0.635251  0.444325  -1.430 0.152803
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1728.4 on 4908 degrees of freedom
## Residual deviance: 1372.5 on 4903 degrees of freedom
## AIC: 1384.5
##
## Number of Fisher Scoring iterations: 7

```

After that, we pick up the reduced model without `heart_disease` which represents the explanatory feature with higher p-value and we continue to work in the same way we have just seen. We repeated the scheme with `age`, without earning benefits.

The AIC has a value of 1386.2.

```

mod4 <- glm(stroke ~ age + avg_glucose_level + hypertension + age*hypertension,
            family=binomial)
summary(mod4)

```

We go further analyzing also `avg_glucose_level*hypertension`.

```

mod5 <- glm(stroke ~ age + avg_glucose_level + hypertension +
            avg_glucose_level*hypertension, family = binomial)
summary(mod5)

```

In the end, we include the possible best interactions into the reduced model. It could be a good way, but it does not represent our best solution due to a high p-value.

The corresponding *AIC* is 1384.2.

```
mod6 <- glm(stroke ~ age + avg_glucose_level + heart_disease + hypertension +
             age*heart_disease + heart_disease*hypertension, family = binomial)
summary(mod6)
```

As conclusion, we can promote mod1 as the model which better fits our data:

$$\text{stroke} = \beta_0 + \beta_1 \times \text{age} + \beta_2 \times \text{avg_glucose_level} + \beta_3 \times \text{heart_disease} + \beta_4 \times \text{hypertension} + \beta_5 \times \text{age} \times \text{heart_disease}$$

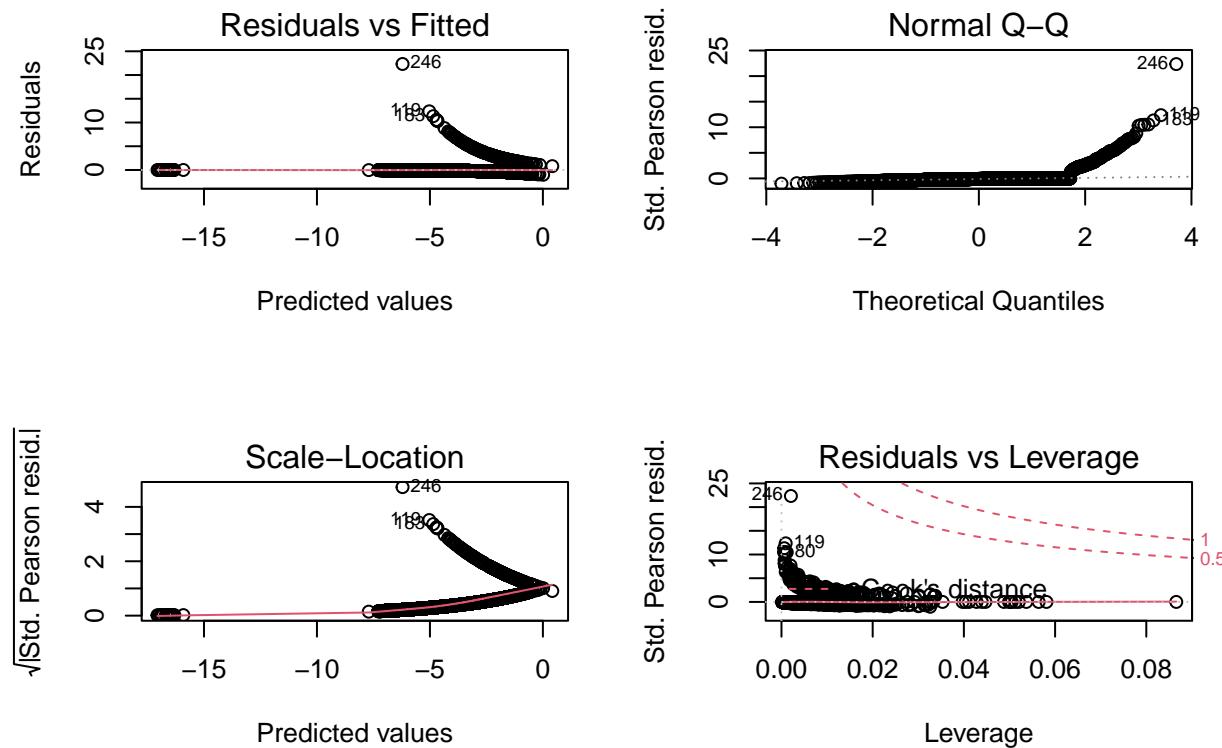
Let's now see some relevant information, such as outliers of mod1:

	gender	age	hypert.	hd	ev_marr	work_type	res_type	glucose	bmi	smoking	stroke
207	Female	81	0	0	Yes	Private	Rural	80.13	23.4	never smoked	1
150	Female	70	0	1	Yes	Private	Rural	239.07	26.1	never smoked	1
100	Female	69	0	0	Yes	Govt_job	Urban	82.81	28.0	never smoked	1

and other properties and characteristics by the plots:

```
par(mfrow=c(2,2))
plot(mod.full)
```

```
## Warning: not plotting observations with leverage one:
##      2971
```



```
par(mfrow=c(1,1))
```

3.1.3 Polynomial models

Let's try to do one more test in logistic regression using the polynomial model starting from the reduced model `mod.red`, adding also `bmi` as predictor. We provide the square of `bmi`, `avg_glucose_level` and then both of them, obtaining `mod.poly1`, `mod.poly2` and `mod.poly3` respectively.

```
mod.poly1 <- glm(stroke ~ age + heart_disease + avg_glucose_level+ hypertension
                  + bmi + I(bmi^2), family = binomial)
mod.poly2 <- glm(stroke ~ age + heart_disease + avg_glucose_level + hypertension
                  + bmi + I(avg_glucose_level^2), family = binomial)
mod.poly3 <- glm(stroke ~ age + heart_disease + avg_glucose_level + hypertension
                  + bmi + I(bmi^2) + I(avg_glucose_level^2), family = binomial)
```

However, we do not achieve anything interesting with polynomial model. There are no improvements in the results, seeing the value of the *AIC* which is extremely high.

3.2 Bayesian modelling

In this section we now study other types of predictive models, which are based on Bayesian concepts. These methods exploit conditional probability and Bayes theorems to make predictions, they are also widely used in classification problems because of their nice structure and properties. Furthermore, we would like to compare their results with the logistic model in order to assess their performances. Recall that we have to consider the LDA and QDA models which approximate the Bayesian classifier (the ideal one) as from a computation point of view it is very expensive and requires that the likelihood and the prior distribution to be conjugate (i.e their distributions are from the same family of distribution).

3.2.1 LDA

We first consider the Linear Discriminant Analysis (LDA) by taking into account the Multivariate case. When we apply LDA model to approximate the Bayesian classifier we have assumed that the likelihood follows a Normal distribution $X|G_j \sim \mathcal{N}(\mu_j, \Sigma) \quad j = 0, 1$, and covariates with same Σ for both classes. Hence we have:

$$P(G_j|x) \propto \pi_j \frac{1}{2\pi|\Sigma|^{1/2}} \exp(-0.5(x - \mu)^T \Sigma^{-1}(x - \mu)) \quad j = 0, 1$$

Where the parameters π, μ, Σ are plugged in with their associated estimator:

$$\hat{\pi} = \frac{n_j}{n}, \quad \hat{\mu} = \bar{x}_j, \quad \hat{\Sigma} = \sum_{j=1}^2 (x_i - \mu_j)^T (x_i - \mu_j)/(n - 2).$$

Furthermore, the LDA will assign for each sample the class with the largest posterior probability using the discrimination score $\delta_j(x)$.

In the following code we fit the data by considering all the predictors independent to each other, in order to not have collinearity issue.

```
library(MASS)
lda.fit <- lda(stroke ~ age + bmi + avg_glucose_level + hypertension + gender
               + smoking_status + Residence_type + heart_disease)
lda.pred <- predict(lda.fit)
lda.pred.stroke <- lda.pred$posterior[, 2]
table(lda.pred$class, stroke)

##      stroke
##      0      1
```

```
##    0 4653 194
##    1   47   15
```

Since we are dealing with an unbalanced dataset we are not satisfied with this criteria of assigning the class, thus in chapter 4 we will cover a better technique to define the best posterior threshold.

3.2.2 QDA

Assuming homoscedasticity among the classes can be very restrictive in our case since the dataset is very unbalanced and skewed. Thus we considered also the Quadratic Discriminant Analysis (QDA) which estimates μ_j and Σ_j for each class separately. This choice will turn out to have a more flexible and reliable model than the previous one

```
qda.fit <- qda(stroke ~ age + bmi + avg_glucose_level + hypertension + heart_disease
                 + smoking_status, data = stroke_data)
qda.pred <- predict(qda.fit, stroke_data)
qda.pred.stroke <- qda.pred$posterior[, 2]
table(qda.pred$class, stroke)

##      stroke
##           0     1
##    0 4260 123
##    1  440   86
```

As previously, we will need to cover better also the threshold of the QDA model.

4 Best Model Selection & Validation Test

In this chapter of the report we are now going to evaluate our elective models that we have seen so far and pick the best among them. Furthermore we will also discuss the metrics used to define the best classification threshold of the predicted variables. The models that we elected for this analysis, which have shown to represent better our problem are: the two logistic regressions (reduced and interaction based), and the models built with the Bayesian modelling approach (LDA and QDA).

In order to assess them we used the *validation testing method*, hence we split the dataset into validation and training sets. But since our dataset is highly unbalanced we need to define the appropriate split in order to have reasonable amount of stroke cases in both sets.

We also recall that the data is made of 4909 instances and just the 4.26% of the people are affected by the stroke. We decided then to keep the 75% of the data for the training data: 3682 samples of which 3562 are the ones that have not incurred to a stroke, while 120 of them had it. The remaining data are part of the validation set.

Cross-validation tests have not been developed since constructing disjoint splits and random sampling from the data is not feasible with just few cases of stroke from the whole dataset.

4.1 Data split

In the following *R* code we developed the above mentioned splitting and sampling methodology.

First we retrieved the shuffled indices of the people affected or not by the stroke in different variables.

```
no.strokes.data <- stroke_data[stroke == 0, ]
rnd.idx.no.strokes <- sample(c(1:dim(no.strokes.data)[1]))

yes.strokes.data <- stroke_data[stroke == 1, ]
rnd.idx.yes.strokes <- sample(c(1:dim(yes.strokes.data)[1]))
```

We construct the training set made of 75% instances of the whole dataset and 4,6% of them are affected by stroke.

```
training.set <- no.strokes.data[rnd.idx.no.strokes[1:3522], ]
training.set <- rbind(training.set, yes.strokes.data[rnd.idx.yes.strokes[1:160], ])
shuffle <- sample(nrow(training.set))
training.set <- training.set[shuffle, ]
```

The remaining data is then used for the validation set and it is made of 25% with 4,1% stroke cases.

```
val.set <- no.strokes.data[rnd.idx.no.strokes[3523:4700], ]
val.set <- rbind(val.set, yes.strokes.data[rnd.idx.yes.strokes[161:209], ])
shuffle <- sample(nrow(val.set))
val.set <- val.set[shuffle, ]
```

4.2 ROC and PRECISION-RECALL Curves

Due to the high unbalance stroke rate, defining the appropriate threshold on the predicted variable helps to overcome this issue and improves the results of the model. Note that all the models used until now give a probability output for each class, hence we will analyze the thresholds based on those outputs.

Since we are dealing with a clinical problem the presence of false negative (FN) cases is more critical than having false positive (FP) ones, for this reason we considered not just the ROC curve but also the Recall-Precision curve for each model, where the Recall function is defined as $Recall = \frac{TP}{TP+FN}$ and the Precision as $Prec = \frac{TP}{TP+FP}$.

In order to make all these analyses, we built a function that takes a list of model predictions, for each of them we evaluate the ROC and the Precision-Recall curves and we retrieve the best threshold of both curves in a data frame table, furthermore plot of the curves are also shown.

The best threshold for ROC curves corresponds to the value that maximize more the true positive rate and minimize the false positive rate. Instead for the Precision-Recall curve case we would like first to minimize the false negative rate (high recall value) and then try to maximize the precision, therefore, in order to have a general optimal rule for satisfying both objectives, we retrieve the maximum value of the sum of recall and precision pairs.

```
library(pROC)

## Type 'citation("pROC")' for a citation.

##
## Caricamento pacchetto: 'pROC'

## I seguenti oggetti sono mascherati da 'package:stats':
##
##      cov, smooth, var

library(ROCR)
get.roc.recall.values <- function(pred_models, true_value, yes_plot=TRUE) {
  result <- data.frame(Thr.ROC=double(), Specificity=double(), Sensitivity=double(),
                        Thr.Prec.Rec=double(), Recall=double(), Precision=double())
  n_models = length(pred_models)
  if (yes_plot==TRUE) {
    png(filename="./curves.png")
    par(mfrow=c(n_models, 2))
  }
  for (pred in pred_models) {
    ### ROC
    roc.res <- roc(true_value, pred, levels=c("0", "1"))
    if (yes_plot==TRUE){
```

```

    plot(roc.res, print.auc=TRUE, legacy.axes=TRUE, xlab="False positive rate", ylab="True positive rate")
}

best.roc <- coords(roc.res, "best")

### PREC-REC
pred.rec = prediction(pred, true_value)
perf = performance(pred.rec, "prec", "rec")
if (yes_plot==TRUE){
  plot(perf)
}
pr_cutoffs <- data.frame(cutrecall=perf@alpha.values[[1]], recall=perf@x.values[[1]],
                           precision=perf@y.values[[1]])
pr_cutoffs <- pr_cutoffs[pr_cutoffs$recall>0 & pr_cutoffs$precision >0, ]
#Maximize Recall + Precision
best_recall <- pr_cutoffs[which.max(pr_cutoffs$recall + pr_cutoffs$precision), ]

result[nrow(result) + 1,] = c(best.roc[1, 1], best.roc[1, 2], best.roc[1, 3],
                                best_recall[1, 1], best_recall[1, 2], best_recall[1, 3])
}
if(yes_plot==TRUE){
  dev.off()
}
par(mfrow=c(1, 1))
return(result)
}

```

4.3 Model evaluation

We first fit all the elective models on our training set.

```

attach(training.set)

## I seguenti oggetti sono mascherati da stroke_data:
##
##      age, avg_glucose_level, bmi, ever_married, gender, heart_disease,
##      hypertension, Residence_type, smoking_status, stroke, work_type
# Logistic reduced model
mod.red.train <- glm(stroke ~ age + heart_disease + avg_glucose_level +
                      hypertension, data = training.set, family = binomial)
mod.red.train.pred <- predict(mod.red.train, data = training.set, type = "response")

# Logistic interaction model
mod1.train <- glm(stroke ~ age + avg_glucose_level+ heart_disease + hypertension +
                     + age*heart_disease, data = training.set, family = binomial)
mod1.train.pred <- predict(mod1.train, data = training.set, type = "response")

# LDA model
lda.fit.train <- lda(stroke ~ age + bmi + avg_glucose_level + hypertension +
                         smoking_status + Residence_type + heart_disease,
                         data = training.set)
lda.fit.train.pred <- predict(lda.fit.train, data = training.set)
lda.fit.train.pred <- lda.fit.train.pred$posterior[, 2]

# QDA model

```

```

qda.fit.train <- qda(stroke ~ age + bmi + avg_glucose_level + hypertension +
                      heart_disease + smoking_status, data = training.set)
qda.fit.train.pred <- predict(qda.fit.train, data = training.set)
qda.fit.train.pred <- qda.fit.train.pred$posterior[, 2]

```

The next step instead involves the calculation of the best threshold by using two mentioned metrics. Thus we use our built-in function by passing the elective models, then we print the summary of the results and extract the thresholds of the two metrics in different variables, which will be helpful when we will need to apply them.

```

res = get.roc.recall.values(list(mod.red.train.pred, mod1.train.pred,
                                 lda.fit.train.pred, qda.fit.train.pred),
                             training.set$stroke, yes_plot=TRUE)
print(res)

##      Thr.ROC Specificity Sensitivity Thr.Prec.Rec Recall Precision
## 1 0.02946776    0.6558773     0.89375 0.0093263679 0.99375 0.07098214
## 2 0.02862206    0.6547416     0.89375 0.0086306120 0.99375 0.07107734
## 3 0.03307245    0.7413401     0.80625 0.0090135012 0.99375 0.07001321
## 4 0.02999548    0.7407723     0.81875 0.0002385786 1.00000 0.06714226

recall_thresholds <- res$Thr.Prec.Rec
roc_thresholds <- res$Thr.ROC

```

Once fitting the models, we would like to see their performance in the training set in order to have a first evaluation, hence for each prediction we apply the associated threshold returned by the built-in function.

```

mod.red.train.pred.class <- as.numeric(mod.red.train.pred) >= roc_thresholds[1])
table(training.set$stroke, mod.red.train.pred.class, dnn=c('Ground thruth', 'Reduced model
predicted'))

##          Reduced model
##          predicted
## Ground thruth   0   1
##                 0 2310 1212
##                 1   17  143

mod.red.train.pred.class <- as.numeric(mod.red.train.pred) >= recall_thresholds[1])
table(training.set$stroke, mod.red.train.pred.class, dnn=c('Ground thruth', 'Reduced model
predicted'))

##          Reduced model
##          predicted
## Ground thruth   0   1
##                 0 1441 2081
##                 1   1   159

mod1.train.pred.class <- as.numeric(mod1.train.pred) >= roc_thresholds[2])
table(training.set$stroke, mod1.train.pred.class, dnn=c('Ground thruth', 'Interaction model
predicted'))

##          Interaction model
##          predicted
## Ground thruth   0   1
##                 0 2306 1216
##                 1   17  143

```

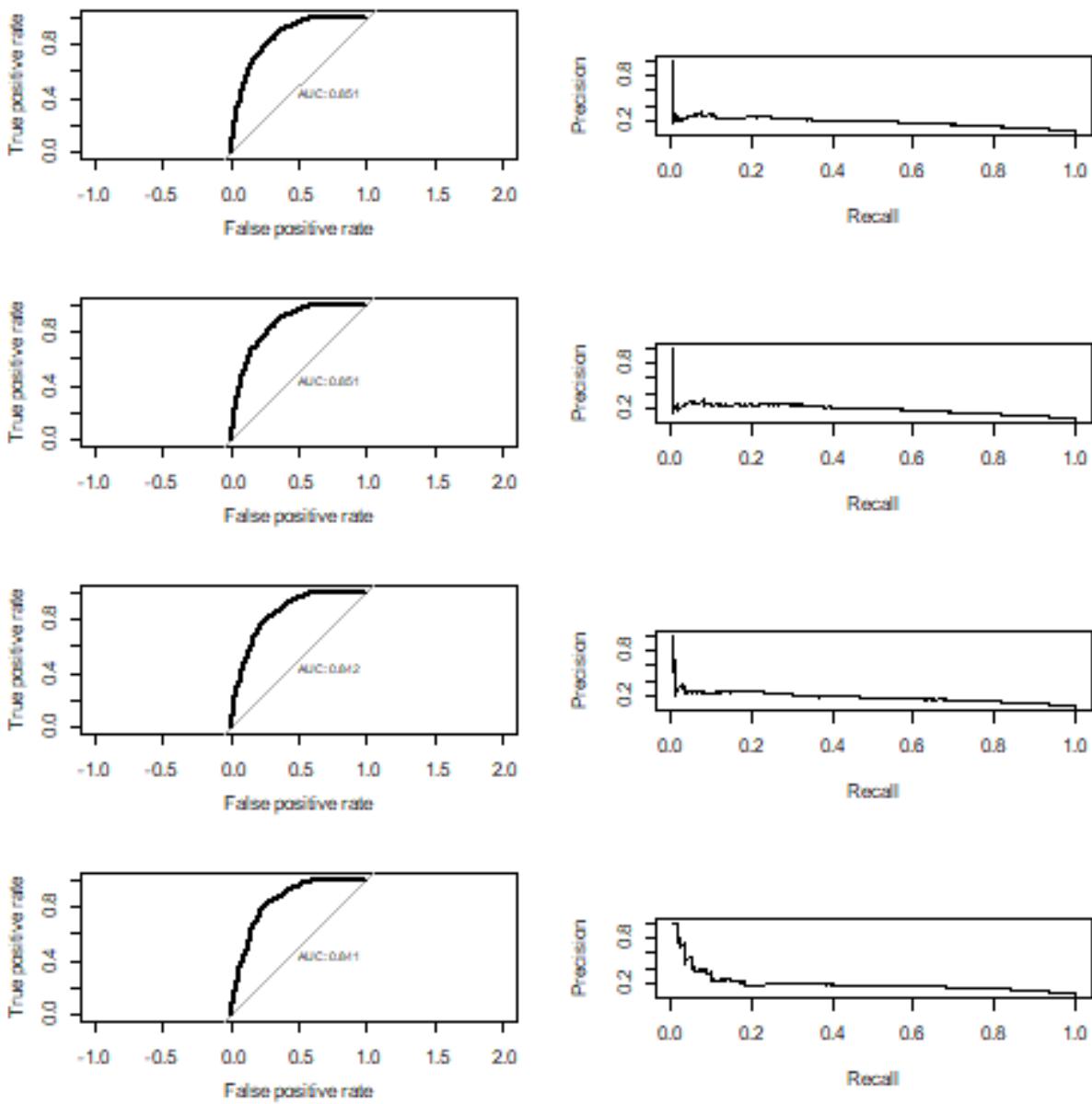


Figure 3: roc and prec-rec curves, with the order: mod.red, mod1, LDA and QDA.

```

mod1.train.pred.class <- as.numeric(mod1.train.pred >= recall_thresholds[2])
table(training.set$stroke, mod1.train.pred.class, dnn=c('Ground thruth','Interaction model
predicted'))

##           Interaction model
##           predicted
## Ground thruth   0   1
##             0 1444 2078
##             1   1   159

lda.fit.train.pred.class <- as.numeric(lda.fit.train.pred>=recall_thresholds[3])
table(training.set$stroke, lda.fit.train.pred.class, dnn=c('Ground thruth','LDA model
predicted'))

##           LDA model
##           predicted
## Ground thruth   0   1
##             0 1410 2112
##             1   1   159

qda.fit.train.pred.class <- as.numeric(qda.fit.train.pred>= recall_thresholds[4])
table(training.set$stroke, qda.fit.train.pred.class, dnn=c('Ground thruth','QDA model
predicted'))

##           QDA model
##           predicted
## Ground thruth   0   1
##             0 1299 2223
##             1   0   160

```

From the confusion matrices of the first two examples of models we can see a clear difference and effectiveness of the two designed thresholds.

The ROC threshold tries to minimize the false positive errors by considering just the positive results, instead the other one, as we see from the output matrix, takes in consideration also the false negative cases. Therefore it results to minimize more the false negative rate, by allowing more errors for false positive values as counter-effect.

For our dataset by using the latter approach we have to highlight that the number of FP has increased a lot, which still could also be a major problem, hence a good rates among these two metrics could be retrieved depending on our desired result.

But as previously mentioned, in clinical cases an error committed to diagnose a disease can be very dangerous, for example when instead of predicting a stroke we classify it as a harmless case. For this reason, from now onward, we will consider only the Recall-Precision optimal threshold since it takes in consideration the False negative cases.

From the above confusion matrix we can already see which model is giving the best result by looking at the FP and FN values of each fitting, in particular we define the optimum choice which minimizes more both values but by giving more importance to false negative minimization.

4.3.1 Model validation

A final assessment has to be done by seeing the performance of our elective models on a set of data where they have not been trained on, for this reason we will use the validation set that we defined previously. In this manner we are able to simulate a real scenario case, detect some possible overfitting issues, and by comparing the performance of each model on this set we can finalize our choice.

In this phase the evaluation is again made by applying a threshold on the predictions, in particular we will use the Precision-Recall thresholds that we calculated in the training phase and see if the given results are

compliant to what we obtained on the training set. Moreover, we will calculate the positive and negative error rates, which represent the misclassification error on the prediction of having, or not, a stroke.

```
# Logistic reduced model
mod.red.val <- predict(mod.red.train, val.set, type="response")
mod.red.val.class <- as.numeric(mod.red.val >= recall_thresholds[1])
conf.matr <- table(val.set$stroke, mod.red.val.class, dnn=c('Ground thruth','reduced model
predicted'))
print(conf.matr)

##           reduced model
##           predicted
## Ground thruth   0   1
##                 0 479 699
##                 1   1   48

## [1] "Positive error rate is : 0.064257"
## [1] "Negative error rate is : 0.002083"

# Logistic interaction model
mod1.val <- predict(mod1.train, val.set, type="response")
mod1.val.class <- as.numeric(mod1.val >= recall_thresholds[2])
conf.matr <- table(val.set$stroke, mod1.val.class, dnn=c('Ground thruth','interaction model
predicted'))
print(conf.matr)

##           interaction model
##           predicted
## Ground thruth   0   1
##                 0 479 699
##                 1   1   48

## [1] "Positive error rate is : 0.064257"
## [1] "Negative error rate is : 0.002083"

# LDA model
lda.val <- predict(lda.fit.train, val.set)
lda.val <- lda.val$posterior[, 2]
lda.val.class = as.numeric(lda.val >= recall_thresholds[3])
conf.matr <- table(val.set$stroke, lda.val.class, dnn=c('Ground thruth','LDA model
predicted'))
print(conf.matr)

##           LDA model
##           predicted
## Ground thruth   0   1
##                 0 481 697
##                 1   2   47

## [1] "Positive error rate is : 0.063172"
## [1] "Negative error rate is : 0.004141"

# QDA model
qda.val <- predict(qda.fit.train, val.set)
qda.val <- qda.val$posterior[, 2]
qda.val.class <- as.numeric(qda.val >= recall_thresholds[4])
conf.matr <- table(val.set$stroke, qda.val.class, dnn=c('Ground thruth','QDA model
```

```

predicted"))
print(conf.matr)

##           QDA model
## predicted
## Ground thruth  0   1
##                0 456 722
##                1   1   48
## [1] "Positive error rate is : 0.062338"
## [1] "Negative error rate is : 0.002188"

```

As we did during the training phase, the evaluation is made by looking at the false negative (FN) and false positive (FP) rates. The ratio of the misclassified non-stroke people is almost zero in all models, so we do almost not make mistake when saying to them that they will not get a stroke disease.

By running the code several times we saw that on average the reduced model (`mod1`) slightly outperforms the other models, by minimizing more the false rates, in addition we preferred it because it allows us to give more insights of the predictors, as we saw in chapter 3.1.

We also noticed that the QDA model shows better results than LDA, since it exploits the heteroscedasticity among stroke classes.

5 Conclusions and Further Analysis

As seen in the course of our analysis, we countered different problems. In particular, many assumptions on the data and on the models have not been satisfied by considering all the given predictors or by taking just a subset of variables, for example the presence of collinearity, data not Normal distributed, heteroscedasticity, high variance, and all the other issues highlighted in the previous chapters.

The tests computed on our data involved various models, in which we have included different combinations of predictors, even by mixing them together to exploits and explore some synergy relationships.

Unfortunately the achieved complexity of the models is not enough for an accurate stroke prediction. We still end up having a high amount of false rates.

Furthermore, by looking at the plots generated in section 2.2 -Descriptive statistic- we can see that the data are not linearly separable, thus it is not so easy to manage them by using linear models.

Despite the encountered predictiveness issues we still can highlight many properties of the relationship among predictors and response variables, in particular thanks to the explanatory analysis made on the usage of logistic regression in this report.

For this reason after the training and validation phases, in which we tested the models we have taken into account, we selected `mod1`

$$\text{stroke} = \beta_0 + \beta_1 \times \text{age} + \beta_2 \times \text{heart_disease} + \beta_3 \times \text{avg_glucose_level} + \beta_4 \times \text{hypertension} + \beta_5 \times \text{age} \times \text{heart_disease},$$

although it also guarantees slightly better performances than others. In particular, in the `summary(mod1)` we can see how each feature contributes on the prediction by looking at the β parameters. The higher contribution on stroke comes from `heart_disease` with a coefficient equal to 2.765, even though its p-value is not the smallest; the other features have all coefficient smaller than 1 but are extremely important too, and highlight strong relationship with having a stroke. Indeed, the strongest relation is always between `stroke` and `age`, older people have a higher probability to get a stroke and this disease will become an increasingly important health problem as the world's population continues to age.

Furthermore, from this model we have been able to find some interactions among `heart_disease` and `age` as described in the chapter 3.1.2, thanks to it we discovered that the response variable has some effects by considering the people with heart disease and not. Finally, we also saw that by considering the interaction we end up having a high increment in the beta value of the heart disease variable.

Resuming the initial question, if it is possible to prevent a stroke, we can answer that with our data we cannot achieve a secure prediction, but only with a moderate confidential interval of 90% on it.

Further analysis have to be explore considering other type of features related with the stroke disease, in order to produce relevant results.

It has also to concern that more data should be collected especially for people affected by stroke and also more correlated pathological variable and health indicators should be considered too, by looking at the available medical literature. These analysis and integration will allow to build more reliable explanatory and predictive models.