Journal: What is the most accurate model in predicting the stroke risk of a person given 10 attributes.

Orfeas Gkourlias

2022-10-23

Introduction

This document aims to explore, analyse and explain the data set being used in answering the following research question: "What is the most accurate model in predicting the stroke risk of a person given 10 attributes?". As the research question implies, the data set consists of 10 attributes. This project has the goal of analyzing those attributes, so that the best machine learning algorithm can be trained. Some attributes affect each other, while others may not. Analysis of these correlations can help in finding the rankings of the attributes. The final answer to the research question will be found using Weka, a data platform which utilizes machine learning.

To get a feel for what the scope and attributes of the data set consists of, it will be loaded and the first 10 results will be displayed.

```
main <- read.csv("../data/stroke-data.csv")
head(main)
```

```
##      id gender age hypertension heart_disease ever_married     work_type
## 1  9046   Male  67            0             1          Yes       Private
## 2 51676 Female  61            0             0          Yes Self-employed
## 3 31112   Male  80            0             1          Yes       Private
## 4 60182 Female  49            0             0          Yes       Private
## 5  1665 Female  79            1             0          Yes Self-employed
## 6 56669   Male  81            0             0          Yes       Private
##   Residence_type avg_glucose_level  bmi  smoking_status stroke
## 1          Urban            228.69 36.6 formerly smoked      1
## 2          Rural            202.21  N/A    never smoked      1
## 3          Rural            105.92 32.5    never smoked      1
## 4          Urban            171.23 34.4          smokes      1
## 5          Rural            174.12   24    never smoked      1
## 6          Urban            186.21   29 formerly smoked      1
```

```
nrow(main)
```

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

There are 12 attributes, 10 of which will be used in the analysis: Gender, age hypertension, heart_disease, ever_married, work_type, residence_type, avg_glucose_level, BMI and smoking_status. The last column indicates whether the person has already experienced a prior stroke. This can be used to the train the machine learning model which will be utilized to answer the research question.

There are 5110 entries in this data set. This is also why the row numbers will not be replaced with the id's, because there is no order in the id numbers. They exceed the number 5110.

The attributes and their units can be seen in the code book on the next page.

Codebook.

```
knitr::kable(codebook)
```

Column	Unit	Description
ID	Number	Unique patient identifier
Gender	Text	“Male”, “Female” or “Other”
Age	Number	Age of patient
Hypertension	Boolean	Whether patient has hypertension
Heart_disease	Boolean	Whether patient has a heart disease
Ever_married	Boolean	Whether patient has ever been married
Work_type	Text	Occupation status of patient
Residence_type	Text	Patient living enviroment
Avg_glucose_level	Number	Average glucose level in blood
BMI	Number	Body mass index of patient
Smoking_status	Boolean	Whether patient smokes or not
Stroke	Boolean	Whether patient has ever experienced a stroke

1. Exploratory data analysis.

The exploratory data analysis consists of multiple sections. In the first section, the raw data will be observed and interpreted. Any cleaning up or filtering of the data will also be performed in this first section. Following the first section will be the exploration of correlations in the data. This part will explain how the attributes may affect each other, and whether any trends can be observed.

1.1. Initial data and attributes.

In this section, the attributes will be examined individually. What these attributes could mean for the research question will be discussed. Any pre-processing or cleanup required will also be performed in this section.

ID.

This column is neither noteworthy for analysis or data structure. This column will therefore be dropped, because the data frame used already has row numbers and this makes the ID redundant.

```
main <- main[2:12]
```

Age.

The age of the patient. At first sight, it might look redundant for this data to be stored as a float, since most of the data consists of a rounded age number. Some of the entries contain very young patients. The younger a patient is, the more important the specificity of the age is, since the age difference is still significant at that point. It is for that reason that any patient under the age of 2 will contain a float number, with two decimal numbers. A couple of those instances will be shown in vector format below:

```
head(c(main[main$age < 2, 2]))
```

```
## [1] 1.32 0.64 0.88 1.80 0.32 1.08
```

The likelihood of a person experiencing a stroke increases with age. This will therefore be an important attribute in the analysis.

Gender.

Gender, containing three separate values: Male, Female and other. There could be a correlation in the gender and stroke risk of a person. An example of gender indirectly affecting the stroke risk of a patient would be the age difference in genders. Women tend to live longer lives than men, therefore the stroke risk may increase for women, because they generally become older. This could be one of many hypothesis.

```
main$gender <- factor(main$gender)
```

Hypertension.

This indicates with a 0 or 1 whether the patient is affected by hypertension. The first attribute which is relevant to the heart status of a patient. These types of attributes will always be important, because any heart condition tends to come with an increased risk of experiencing a stroke. Since this is indicated with a binary number, the patient either has hypertension, which is indicated with a 1, or not, which is indicated with a 0. Correlations will probably be found between hypertension and the other attributes. In this case, it will be easier to work with "Yes" or "No" values. Not only for the sake of consistency, but because Weka otherwise recognizes the 0's and 1's as numeric values, not boolean values This will therefore be changed with the following code:

```
main$hypertension <- factor(main$hypertension, labels = c("No", "Yes"))
```

The datatypes were also changed to that of a factor. This has been done so that later functions will properly work on the data.

Heart Disease.

Similar to the previous attribute, this is also an important element when trying to predict stroke risk. The previous observation also applies to this attribute.

```
main$heart_disease <- factor(main$heart_disease, labels = c("No", "Yes"))
```

Ever married.

This displays whether the patient has ever been married in their lifetime. This will most likely not bet detrimental in predicting the stroke risks of patients. But this is part of the data set, so it will therefore be compared with the other attributes, to see where it ranks with it's prediction.

```
main$ever_married <- factor(main$ever_married, labels = c("No", "Yes"))
```

Work Type.

A similar attribute to the prior one. Will most likely not be a good predictor for stroke risk. But it may rank higher than the marriage attribute. Some sectors could theoretically expose a person to environments where strokes are more likely.

```
main$work_type <- factor(main$work_type)
```

Residence Type.

Considering that some types of residency might be healthier than others, this attribute may be slightly important in determining the stroke risk of a person.

```
main$Residence_type <- factor(main$Residence_type)
```

Average Glucose Level.

This may be more important than the prior three attributes, especially when these levels are unusually low or high. The literature concerning glucose levels and their connection to strokes is still being debated. Some papers conclude that it is not detrimental when observed in non-diabetic people.

BMI.

The body mass index is an indicator for how a person's weight/height ratio. Age and gender also being taken into consideration for the calculation. Both extreme ends of this attribute could be important to the stroke risk of a person. Higher BMI levels are also associated with developing heart disease. Glucose levels may also be affected. Several other attributes are most likely going to have a correlation to this attribute.

It is worth noting that this is the only attribute with missing values. This can be shown with a summary. Before doing that however, it seems that the BMI column consists of characters, not numbers. This must first be changed.

```
typeof(main$bmi)
```

```
## [1] "character"
```

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

```
## Warning: NAs introduced by coercion
```

```
summary(main$bmi)
```

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

There appears to be 201 missing values. Imputation will be chosen to remedy this. Because of the insignificant amount, and the attribute being a numerical one, making it easier to fill in missing data properly. This will be done with the usage of the "Hmisc" package. Median will be chosen for the new values because the median is not affected by extreme values, unlike mean. Extreme values are plenty under the BMI attribute in this dataset.

```
main$bmi <- as.vector(impute(main$bmi, median))
main$bmi <- round(main$bmi, 1)
```

Smoking Status.

Whether a patient is smoking will most likely affect some of the other attribute s in this data set. Whether these significant correlations will need to be tested. The smoking status is unknown for some of the patients.

```
main$smoking_status <- factor(main$smoking_status)
```

Stroke.

The column indicating whether the patient has ever had a stroke. This is the classifier which will be predicted for in the final model, with the highest possible accuracy.

```
main$stroke <- factor(main$stroke, labels = c("No", "Yes"))
```

1.2. Correlations.

Following the examination of the initial values and attributes, they may now be compared so that trends and correlations can be observed. Starting with the first attribute, which will most likely be important in determining stroke risk, age. By plotting the summaries of patients who had a stroke, and those who did not, maybe a link can be seen between the two.

```
no_stroke <- main[main$stroke == "No",]
stroke <- main[main$stroke == "Yes",]
ggplot(main, aes(x=factor(stroke), y=age))+geom_boxplot()+
  ggtitle("Ages grouped by stroke history")+
  theme(plot.title = element_text(hjust = 0.5), plot.caption = element_text(hjust=0),
        axis.title.x = element_text(margin = unit(c(5, 0, 0, 0), "mm")))+
  xlab("Stroke Status")+
  ylab("Age")
```

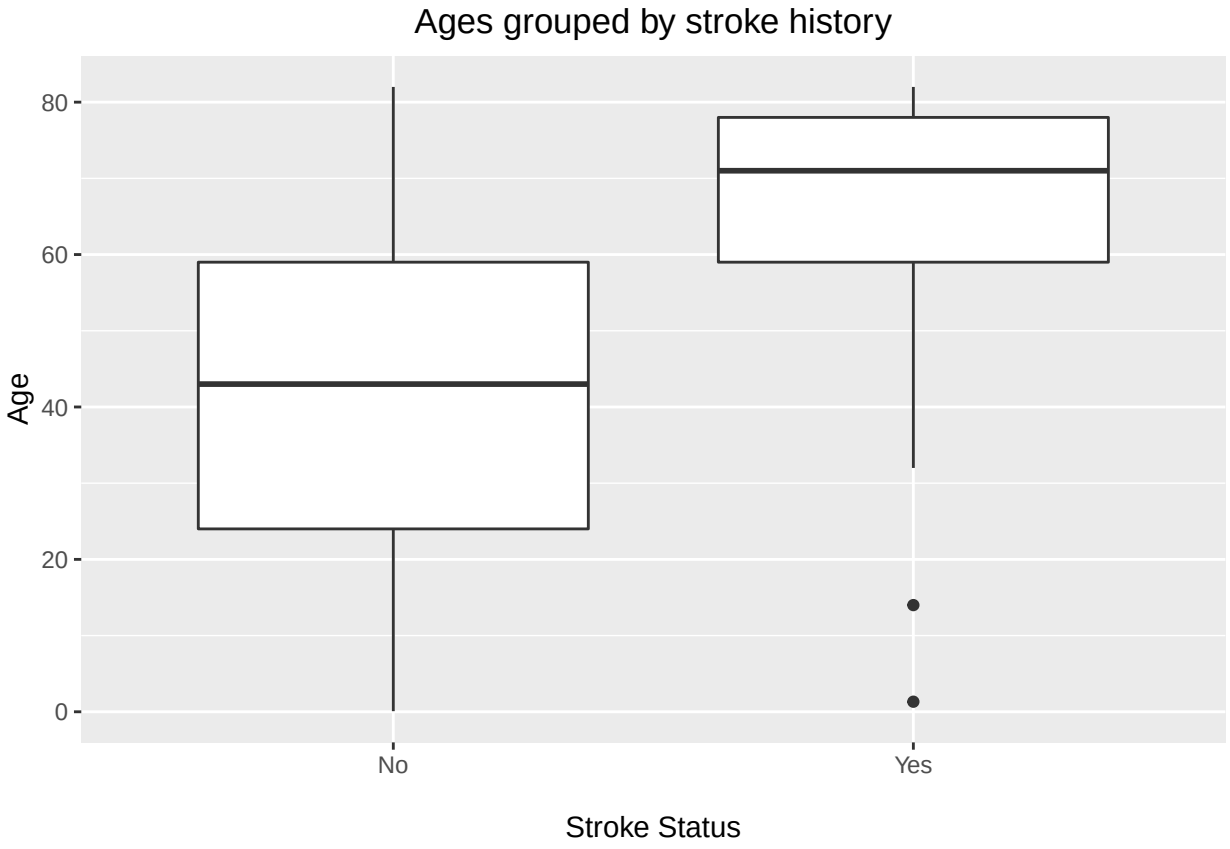

Figure 1: Boxplots of age distribution grouped by stroke status.

This plot shows that the group of people who have had strokes are, on average, older than the group who has not experienced a stroke. This may also be observed with a summary

```
summary(no_stroke$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.08   24.00   43.00   41.97   59.00   82.00
```

```
summary(stroke$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.32   59.00   71.00   67.73   78.00   82.00
```

It is worth noting that the no stroke group does contain plenty of patients of higher age.

Now, a t-test may be performed to determine whether age is a significant difference between the two groups. A one sample t-test is most appropriate here, considering that every patient's risk is independent from another

```
t.test(no_stroke$age, stroke$age)[3]
```

```
## $p.value
## [1] 2.115685e-95
```

Without considering the other attributes, it seems that with that p-value, the two age means are significantly different.

These tests may be performed for all of the other attributes too. Showing them all individually would be redundant, since the correlations between having a stroke and the other attributes relating to the heart already have literature confirming them.

The attributes, which may seem less influential at first would be worth checking manually. The marriage status being the first one to consider. This column is not a boolean number, even though it is comparable to the other 1 or 0 columns. For consistency, yes will become 1 and no will become 0.

```
no_stroke <- main[main$stroke == "No",]
stroke <- main[main$stroke == "Yes",]

no_s_table <- log(table(no_stroke$ever_married))
s_table <- log(table(stroke$ever_married))

vec <- c(no_s_table[1], no_s_table[2], s_table[1], s_table[2])

barplot(vec, space = c(0,0,1,0), col = c("brown4", "cadetblue"),
        names.arg = c("                        No Stroke","",
                      "                        Stroke",""), ylab = "Log(Occurance)",
        main = "Normalized marriage numbers comparison")
legend("topright", c("Not Married", "Married"), fill = c("brown4", "cadetblue"))
```

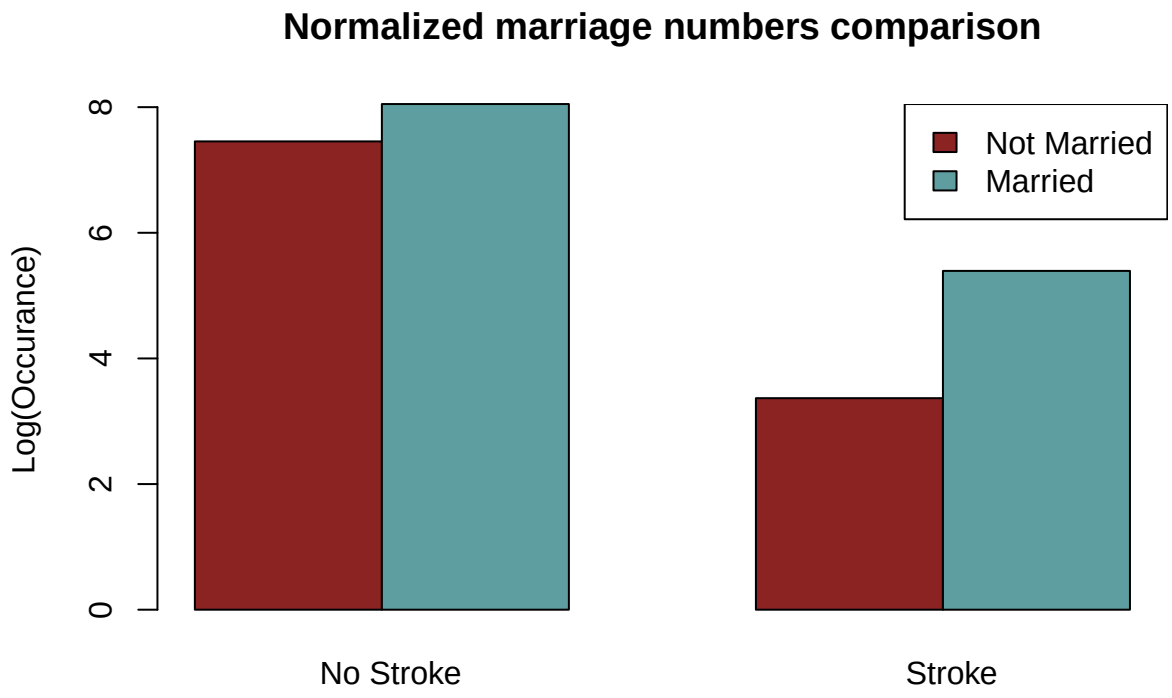

Figure 2: Barplots of normalized marriage status grouped by stroke status

The data has been normalized, so that it may be properly displayed. The group of people with no stroke quite lower than the other group. This plot shows that for both groups, the amount of people who have ever been married is bigger than the not married group. It makes sense for there to be proportionally more people who have married than less when looking at the stroke bars. This is because people who have had a stroke are on average older than the other group. It is also safe to assume that the older a person is, the more likely that they have ever been married in their life.

Now, it is possible to make a similar plot, but with a more relevant attribute. Heart disease would be a good choice.

```
no_s_table <- log(table(no_stroke$heart_disease))
s_table <- log(table(stroke$heart_disease))

vec <- c(no_s_table[1], no_s_table[2], s_table[1], s_table[2])

barplot(vec, space = c(0,0,1,0), col = c("brown4", "cadetblue"),
        names.arg = c("                        No Stroke","",
                      "                        Stroke",""), ylab = "Log(Occurance)",
        main = "Normalized heart status comparison")
legend("topright", c("No Heart disease", "Heart Disease"),
       fill = c("brown4", "cadetblue"))
```

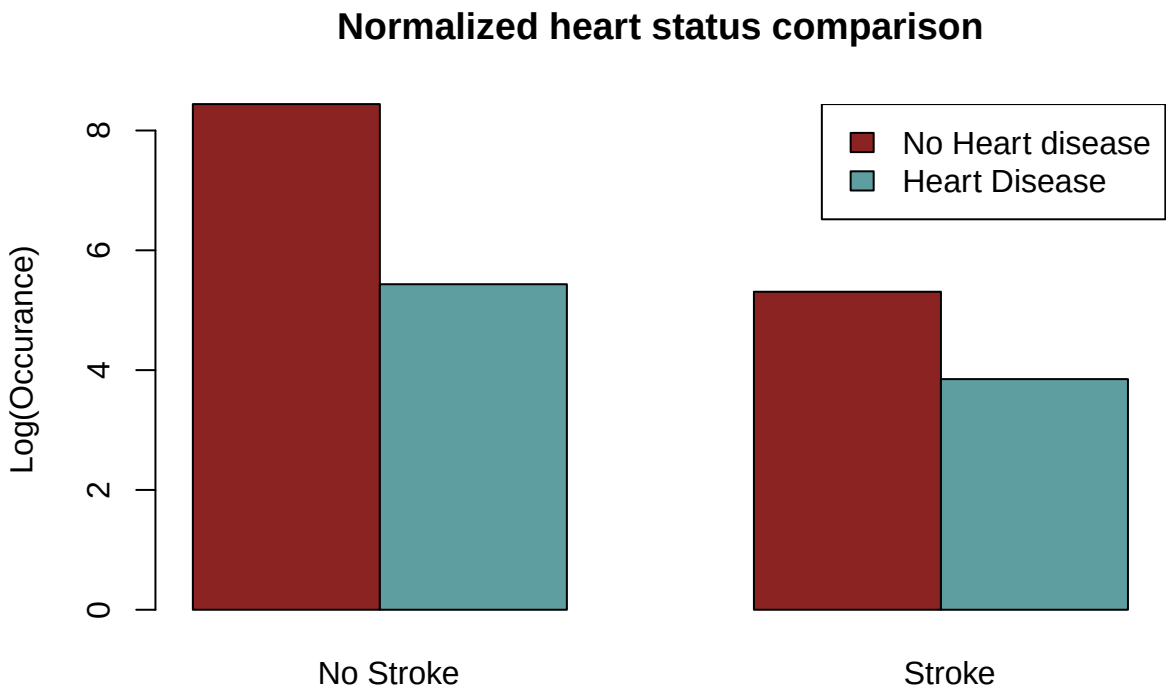

Figure 3: Barplots of normalized heart disease status grouped by stroke status

The assumption would be that a heart disease has a correlation with having a stroke. The bar plot above does not show anything out of the ordinary. It might be difficult to find any correlation by just looking at these bar plots. Calculating the differing odds between two groups could help.

```
ratio1 <- as.vector(table(no_stroke$heart_disease)[2])/nrow(no_stroke)
ratio2 <- as.vector(table(stroke$heart_disease)[2])/nrow(stroke)
ratio1
```

```
## [1] 0.04710965
```

```
ratio2
```

```
## [1] 0.188755
```

In this case, it does appear that the ratio is different. A calculation can be made to determine how much more likely patients with heart conditions may get a stroke.

```
ratio2/ratio1
```

```
## [1] 4.006717
```

Judging by that simple calculation, a patient with a heart condition has 4 times the likelihood of getting a stroke than someone without a heart condition.

A bar plot showing the correlation between BMI and other attributes could also show correlations.

```
ggplot(main, aes(x=factor(stroke), y=bmi))+geom_boxplot()+ggtitle("BMI by stroke status")+
  xlab("Stroke")+
  theme(plot.title = element_text(hjust = 0.5), plot.caption = element_text(hjust=0),
        axis.title.x = element_text(margin = unit(c(5, 0, 0, 0), "mm")))
```

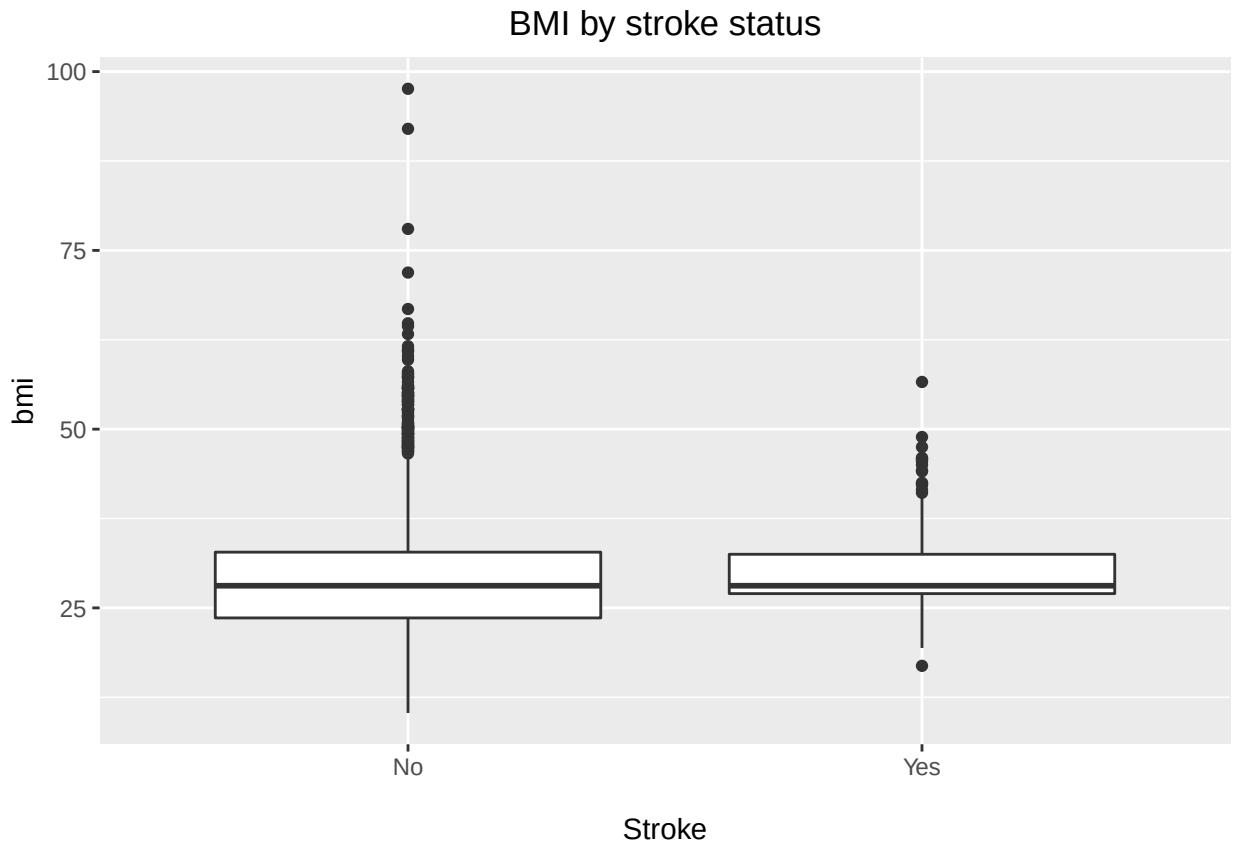

Figure 4: Boxplots of BMI distribution grouped by stroke status.

1.3. Overview.

In finishing the exploratory data analysis section of the journal, more insight was gained into the implications of the data and the attribute properties. Correlations were also found, especially when observing the attributes related to cardiology.

The data is a good candidate for analysis using machine learning algorithms, now that the data has been polished and trimmed.

```
path_out = "../data/"
write.csv(main, paste(path_out, "main.csv", sep = ""), row.names = FALSE)
```

2. Machine Learning & Weka.

In this section, the various ways of training a machine learning algorithm will be tried and tested. The data frame will first be prepared for Weka usage. Second follows an examination of what results will be important considering the data set. So that the most important metrics can be accounted for. This will all be done to make guarantee that the maximum potential of the algorithms is being observed in the third and last section, in which the program Weka will be used to train machine learning models for predicting the stroke attribute.

2.1. Weka.

The data set will soon be put through weka algorithm testing. The goal here is to train an algorithm which will have the most accuracy when it comes to stroke prediction. RWeka is a package which allows for some Weka procedures to be performed in R. Specifically the writing of .arff files from data frames. Whenever a procedure was performed in the Weka program itself, it will be specified in this document.

The main file is currently a .csv file. Weka works better with .arff files, so it will therefore be copied to an .arff format for future use.

```
RWeka::write.arff(main, file = paste(path_out, "main.arff", sep = ""))
```

2.2. Important Metrics.

The accuracy of a model will be most important, followed by the sensitivity The following section will explain why detecting true negatives and the specificity, are so important in this case.

Specificity

When looking at the significance of false positives and negatives, the effect of both must be considered relative to their data set. In this case, false positive and negatives both have differing degrees of severity. When a false positive gets detected, a patient will falsely be classified as being at risk of having a stroke. Using a machine learning model as the only diagnosis method should never be the only course of action. It is therefore safe to assume that followup medical examinations of the patient will be performed, allowing for the false p positive to be corrected. In this worst case, when the false positive does not get corrected, a patient may have to adjust their lifestyle and take medication. There are no negatives to the first solution, since the majority of lifestyle adjustments will overlap with the general consensus on what it means to live a healthy life. But the potential of medication would be problematic, because exposing someone to unnecessary medication could endanger a patient's health.

Now, to compare that to the consequences of a false negative. In the case of a false negative, a person who is at risk of getting a stroke, will be classified a someone who is not actually at risk. This means that the person will have to go through life with an ever increasing risk, since the risk also gets higher the older a person gets. The implication being that the person will eventually suffer an unexpected stroke, which will most likely lead to life debilitating consequences, or even death.

It is for that reason that the false negatives are weighted significantly stricter than false positives, because the outcomes for a patient are so detrimental for the latter. This is the reasoning as to why the sensitivity will be considered more in training a model over specificity.

Data imbalance and ZeroR

Something else which is of high importance is the data imbalance. To properly test the model, there needs to be more entries where the stroke class indicates "Yes". Otherwise, the ZeroR algorithm will always net the highest accuracy, which makes sense with the current imbalance. 95% of the data entries have not yet had a stroke. ZeroR will therefore offer the highest accuracy of 95% with this balance.

It is for that reason that an oversampling technique was chosen to help combat this problem. Synthetic Minority Oversampling Technique (SMOTE) will be used to this end. SMOTE can be performed using the DMwR package.

Another important part of properly training a model is separation between train and test data sets. In this case, there is only one data set present. This data set can be divided into two separate files, so that one may be used for training purposes and the other for testing purposes. It may also undergo cross-validation to ensure the model is not over fitting.

```
set.seed(101)
sample = sample.split(main$stroke, SplitRatio = .75)
train = subset(main, sample == TRUE)
test  = subset(main, sample == FALSE)
train <- SMOTE(stroke ~., train, perc.over = 1800, perc.under = 200)
train[train$age > 2,]$age <- round(train[train$age > 2,]$age)
train$bmi <- round(train$bmi, 1)
train$avg_glucose_level <- round(train$avg_glucose_level, 2)
table(main$stroke)
```

```
##
##   No  Yes
## 4861  249
```

```
table(train$stroke)
```

```
##
##   No  Yes
## 6732 3553
```

The tables show the difference pre and post SMOTE application. Both the subsets will be written to separate .arff files.

```
RWeka::write.arff(train, file = paste(path_out, "train.arff", sep = ""))
RWeka::write.arff(test, file = paste(path_out, "test.arff", sep = ""))
```

Performing ZeroR on this new .arff file gives an accuracy of 65%. This is still an imbalance, but not as severe as before. Models may now be trained using different algorithms.

2.3. Weka Model Exploration.

Following the separation and SMOTE application, the training dataset will be inserted into the Weka experimenter. Using cross-validation (10 folds), multiple different algorithms may be executed and compared. The baseline to which everything will be compared are the ZeroR statistics. It is of high importance that the algorithm is performing significantly better than ZeroR, both in accuracy and false negative rate. A table showing the results of the experimenter will be displayed below.

Table 2: Weka experimenter results showing accuracy, false negative rate and the ROC. ZeroR being the basline for which improvement or degradation is shown.

Algorithm	ZeroR	OneR	NaiveBayes	SimpleLogistic	SMO	IBk	J48	RandomForest
Accuracy	65.45	92.28 ○	80.62 ○	81.91 ○	82.38 ○	89.82 ○	88.08 ○	93.17 ○
False Negative Rate	0.00	0.01 ○	0.20 ○	0.13 ○	0.14 ○	0.05 ○	0.08 ○	0.04 ○
ROC	0.50	0.89 ○	0.88 ○	0.90 ○	0.81 ○	0.88 ○	0.89 ○	0.98 ○

○, • statistically significant improvement or degradation

The table shows that there are significant differences between the algorithms. Every algorithm will be examined and discussed, so that the best one can be found. All the algorithm tests were done with the default settings. Now, considering the importance of false negatives, these are also displayed. The ROC value is also being compared.

The accuracy of the algorithms are quite close to eachother. All of them being higher than 65.45%, the ZeroR accuracy. The lowest being NaiveBayes, at 80.62%. The highest scoring one is RandomForest with 93.17%. RandomForest seems to outperform the other algorithms except OneR on every important metric. The False negative rate is slightly lower for OneR, while having an accuracy difference lower than 1%. The ROC values do differ a bit, with RandomForest having an excellent ROC value of 0.98. This is 0.09 higher than the ROC value of OneR. These two algorithms are the main contenders for now.

Because of the importance of false negatives, another experiment will be performed. This experiment will use the same algorithms, but with a cost sensitive classifier, where the matrix has an increased cost value of 2 for false negatives. False positives remain at a scoring value of 1. Performing the algorithms with these settings results in the following table.

Table 3: Weka experimenter results showing accuracy, false negative rate and the ROC. ZeroR being the basline for which improvement or degradation is shown.

Algorithm	ZeroR	OneR	NaiveBayes	SimpleLogistic	SMO	IBk	J48	RandomForest
Accuracy	65.45	69.92 ○	81.19 ○	79.09 ○	78.47 ○	89.85 ○	86.25 ○	91.67 ○
False Negative Rate	0.00	0.08 ○	0.15 ○	0.07 ○	0.06 ○	0.05 ○	0.05 ○	0.03 ○
ROC	0.50	0.60 ○	0.88 ○	0.90 ○	0.72 ○	0.88 ○	0.89 ○	0.98 ○

○, • statistically significant improvement or degradation

ZeroR

The baseline which all other tests must surpass in accuracy. This simply shows that 65.45% of the data has "No" as classifier.

Naive Bayes

Applying a default Naive Bayes on the data sets provides an accuracy of 77.603%. With 991 correctly classified instances and 286 incorrectly classified instances. The attribute that is being predicted for is a binary one. The application of the "Naive Bayes Multinominal Text" algorithm provide a higher accuracy of 95.14%. This is already a quite impressive result being the first of the linear classifiers.

K-Nearest Neighbors

This algorithm is named as "IBk" in weka, meaning instanced based learner. Using the default settings on the test, an accuracy of 78.856% is reached. While better than the default Naive Bayes test, it is not better than the multinominal text variation. Changing the nearest neighbor search algorithm from the default linear NN search does not net any better results.

Now some decision tree algorithms will be explored.

Decision Stump

Default execution of this algorithm nets an accuracy of 59.122%. Any alteration to the settings is unlikely to show a significant increase in accuracy when the default is already this low.

J48

This algorithm results in a 85.905% accuracy. Playing with the settings either gives a similar or lower accuracy. Looking at other decision tree algorithms will most likely provide higher accuracy.

Random tree

The random tree algorithm gives a 77.291% accuracy. Changing the break ties setting from false to true, allows for slightly better accuracy. The result of this is 79.404%. Changing other settings does not provide a better accuracy.

Random forest

A random forest algorithm results in 81.989% accuracy. The only t hing affecting the accuracy to a significant degree is the number of iterations, which gives a percentage of increase in accuracy.