Introduction to Machine Learning

Reindert Visser

27-9-2021

Exploratory Data Analysis

Research topic

For this project thyroid disease is the main area of research. Problems with the thyroid will result in mental health problems and fatigue. This can be a severe problem and hard to track down. Luckily machine learning can help figure out and calculate patterns. Patterns that otherwise wouldn't be found otherwise.

Which will bring the research question:

Is it possible to predict if a person has thyroid disease by using machine learning, looking at expression values, medicine usage and external factors?

With machine learning, great amounts of data can be loaded up and patterns can be made up from that. When the algorithm encounters new data, these patterns can be identified and based on earlier data; machine learning will become better with more training data. Since the data has a lot of instances, machine learning will be a great tool to use. Since the data has a Sick column, it's either True or False. So this is supervised learning with classification on Sick. Because this is what a patient will look for.

Readying the data

Before the research can start, the data needs to be readied. The data has a lot of information from various topics, such as age, sex but also medicine usage and external factors.

Loading the data

When initially downloaded, the data had an additional column. This was the patient's case, before starting with machine learning; this needs to be removed as soon as possible. The column `Diagnose Letter` was combined with an id of the patient. This was removed using the regular expression `\[[0-9]+\]`, in order to create a true machine learning algorithm.

The labels from the variable `my_labels` were used from the disclosed information in `data/thyroid0387.names`. Except the additional text and explanation.

```
my_data <- read.table("data/thyroid0387.data", sep = ",", header=F, na="?")
my_labels <- read.table("data/labels.txt", sep="\n", header=F)
colnames(my_data) <- as.vector(my_labels[[1]])

(head(dplyr::as_tibble(my_data), n =5))
```

```
## # A tibble: 5 x 30
##     age sex   on_thyroxine query_on_thyroxi~ on_antithyroid_medi~ sick  pregnant
##   <int> <chr> <chr>        <chr>             <chr>                <chr> <chr>
## 1    29 F     f            f                 f                    f     f
## 2    29 F     f            f                 f                    f     f
## 3    41 F     f            f                 f                    f     f
```

```
## 4    36 F     f            f                 f                    f     f
## 5    32 F     f            f                 f                    f     f
## # ... with 23 more variables: thyroid_surgery <chr>, I131_treatment <chr>,
## #   query_hypothyroid <chr>, query_hyperthyroid <chr>, lithium <chr>,
## #   goitre <chr>, tumor <chr>, hypopituitary <chr>, psych <chr>,
## #   TSH_measured <chr>, TSH <dbl>, T3_measured <chr>, T3 <dbl>,
## #   TT4_measured <chr>, TT4 <dbl>, T4U_measured <chr>, T4U <dbl>,
## #   FTI_measured <chr>, FTI <dbl>, TBG_measured <chr>, TBG <dbl>,
## #   referral_source <chr>, Diagnose_letter <chr>
```

The data will be loaded using the `read.table` R build-in function. The data does not have a header and uses a `,` as separator. NA (Not available) is defined with a `?` in the data. Not every instance (patients) have a specific type of thyroid disease, most of these conditions are labelled with `False`. For the hormone measurement columns, these columns form a combination. The hormone name is simply a true or false, while the measured value is the exact amount of hormone expression measured. If there is no hormone expression measured, it will default to NA. This will not form a big issue since the hormone column gives enough information to work with.

For the label file, the data does not have a header either. The separator is a `\n` and there are no NA's. This data is used to overwrite the current column names with the more easy to understand and practical names.

Exploring the data

For the majority of the data, the data is categorical binary data. Which indicates it's either a yes or a no, 1 or 0, True or False. But not all, there are also a few numeric values; which are intervals.

The thyroid plays a big part in the endocrine system, it gives off multiple hormones to regulate the body. There are multiple hormones that the thyroid can give off. The values are tests in order to measure the amount of hormone in the blood. Since most of the hormones transport with blood cells, the thyroid values are testable. If there is a value measured, the default false will become true. With the matching value in the "Measured" column. [1]

- **TSH** (thyroid stimulating hormone): this hormone helps the thyroid release the T3 and T4 hormones. Without TSH, the thyroid would not be able to function at all. If there is too much TSH, the thyroid isn't releasing enough T3 and T4. TSH is released from the hypothalamus and targets the thyroid.
- **T3** (triiodothyronine): is one of the two thyroid hormones, the hormones will influence each other and affect a lot of different processes in the body. Such as metabolism of glucose, the breakdown of cholesterol or increasing the heart rate.
- **T4** (thyroxine): this is the other hormone that will cooperate with the T3 hormone. They have the same function and are involved in the same general processes of the body: Metabolism, growth and increased catecholamine. [2]

All these hormones can be found with tests, in order to prove activity, compare the expression values or prove absence. The test results will be displayed under the measured column in the data, which is continuous data.

- **T3 test**: will measure the amount of T3 in the blood. Since all transport of the thyroid hormones is done via blood cells, this will give an accurate amount of hormones.
- **TT4 test**: will measure the total amount of free T4 and bound to blood T4. Since there are two types of T4, the way that it's used can differ. It can be bound to the blood or roam in the fluid. There for it is also provable by the test.
- **T4U test**: will measure the amount of T4 Uptake, which tests the usage of available T4.
- **TFI test**: will measure the amount of Free T4, Total Free (T4) Index. This can range between 0.7-1.9 nanogram per decilitre of blood.

- **TBG test**: will measure the amount of TBG (thyroxine-binding globulin) enzymes, which is cleaved to produce the T4 hormone. Which means this is an indirect test to measure T4. [3]

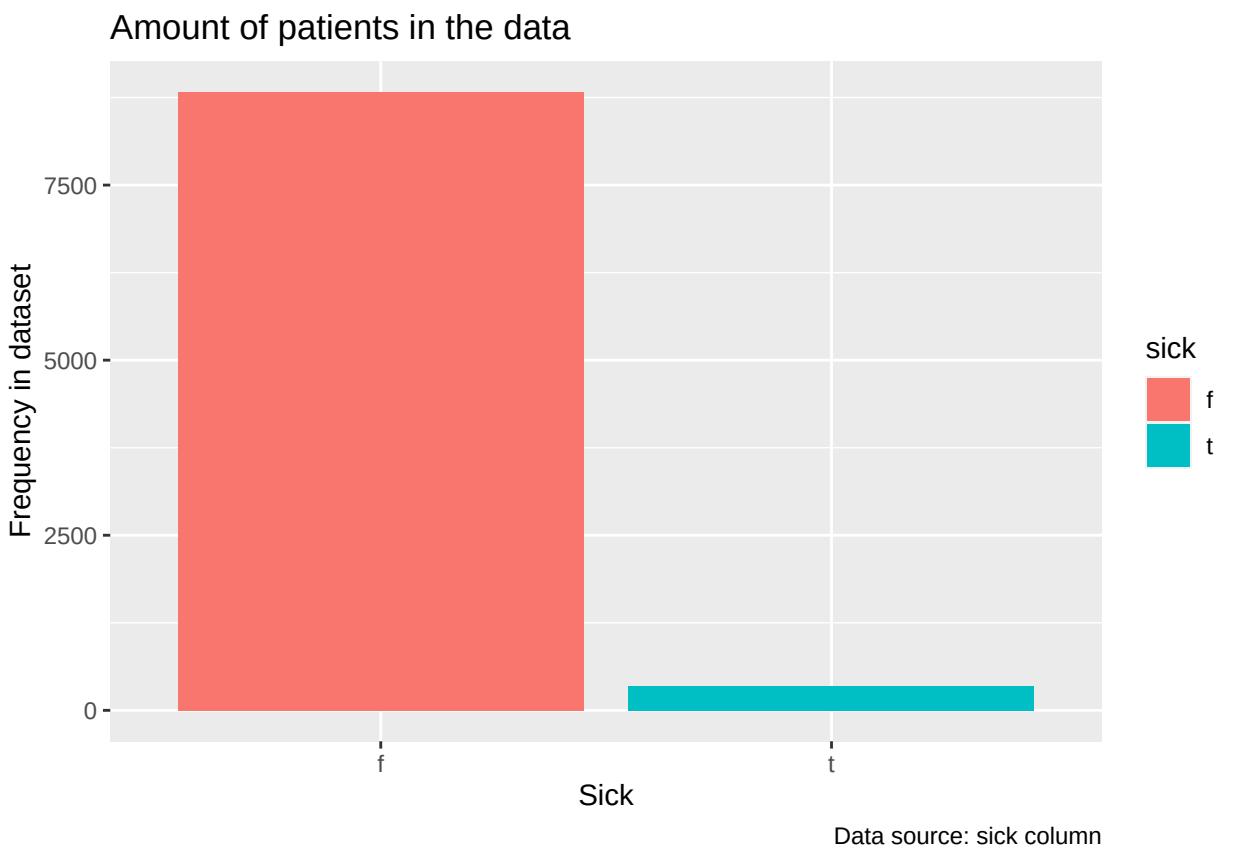
The remaining columns are about external factors. Where the body has either abnormalities or lack of hormones/body resources. These include the following columns:

- **Lithium**: this is an element that circulates in the body and helps different processes. Patients can take extra to stimulate the thyroid and becoming more active. Which will result in true in the data.
- **Goitre**: a major increase in size of the thyroid or local swelling. Which can be diagnosed, if this is the case then the value is true in the data.
- **Hypopituitary**: is a rare case of the pituitary gland does not produce enough hormones. [4]

Visualizing the data

Plotting the data is the fastest way to give a visual view of the data. This can be done by all sorts of ways, since the dataset is mostly categorical; it will be hard to make graphs. Most of the visualization will be done through tables, due to the lack of numeric values.

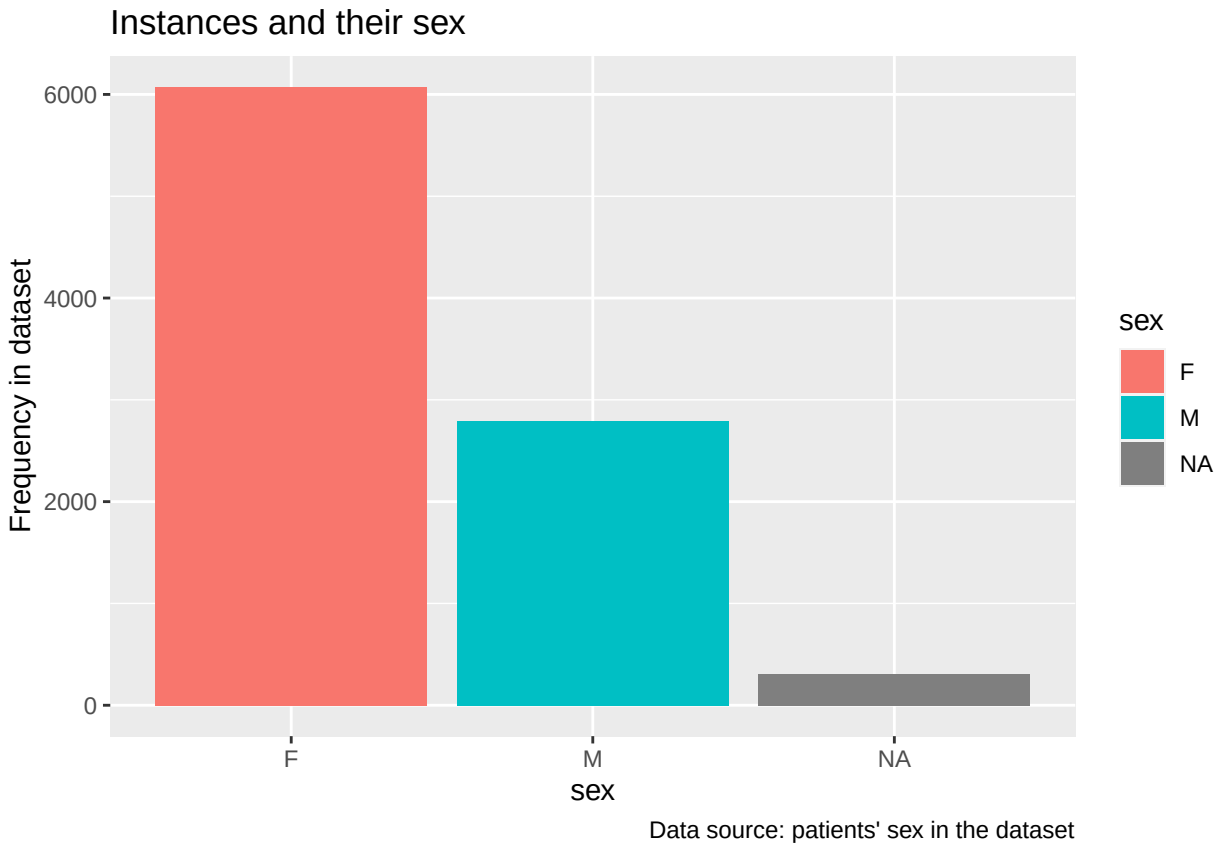
```
ggplot(data = my_data, mapping = aes(x = sick)) +
  geom_bar(aes(fill = sick)) +
  ggtitle("Amount of patients in the data") +
         labs(caption = "Data source: sick column") +
         xlab("Sick") +
         ylab("Frequency in dataset")
```



The great majority of the instances in the data are not patients. Only around 3% of the instances are patients. Which might become tricky to correctly diagnose the real patients according to their thyroid values.

One of the main factors in the data is sex, men tend to go less to the doctor. Since thyroid involves many mental problems, fatigue and eating disorders there will be a difference between sex.

```
ggplot(data = my_data, mapping = aes(x = sex)) +
  geom_bar(aes(fill = sex)) +
  ggtitle("Instances and their sex") +
         xlab("sex") +
         ylab("Frequency in dataset") +
         labs(caption = "Data source: patients' sex in the dataset")
```

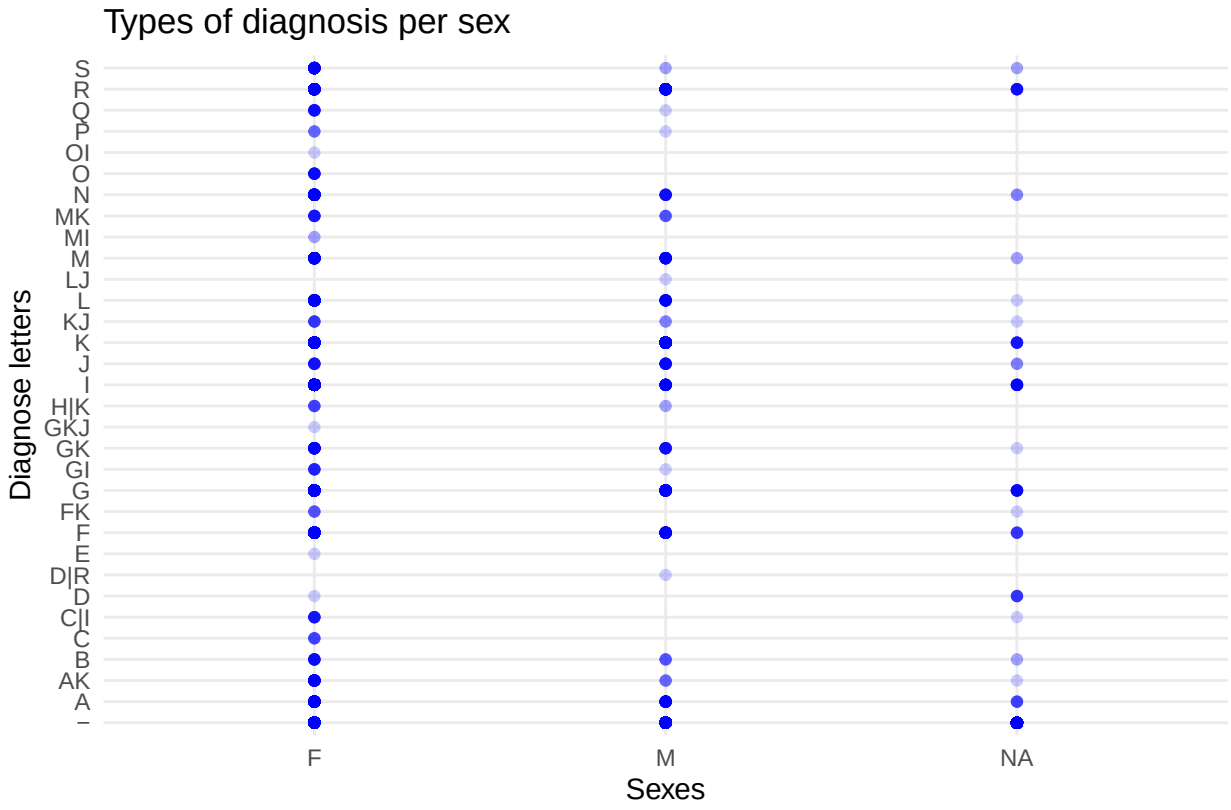


As predicted there are more female instances in the data. There are also people that did not want to specify their sex, these instances are labelled under the 400 NA in the data.

Since there are differences between sexes, not only body but also in hormones. The thyroid might act different, this might affect the effects people feel and how they respond to this.

```
ggplot(data = my_data,
  mapping= aes(x = sex, y= Diagnose_letter)) +
  geom_point(col= "blue", alpha = 0.2) +
  geom_smooth(method="loess", se=FALSE) +
  ggtitle("Types of diagnosis per sex") +
  xlab("Sexes") +
  ylab("Diagnose letters") +
  labs(caption = "Data source: sex and diagnose columns") +
  theme_minimal()
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Not all males have the same problems as females, some more frequent then others. NA is added this time to give a general overview of the data.

There are many types of diagnosis, in varying degree of frequency. This will give insight in the distribution of the dataset.

```
pander(table(my_data$Diagnose_letter))
```

Table 1: Table continues below

-	A	AK	B	C	C|I	D	D|R	E	F	FK	G	GI	GK	GKJ
6771	147	46	21	6	12	8	1	1	233	6	359	10	49	1

H|K	I	J	K	KJ	L	LJ	M	MI	MK	N	O	OI	P	Q	R	S
8	346	30	436	11	115	1	111	2	16	110	14	1	5	14	196	85

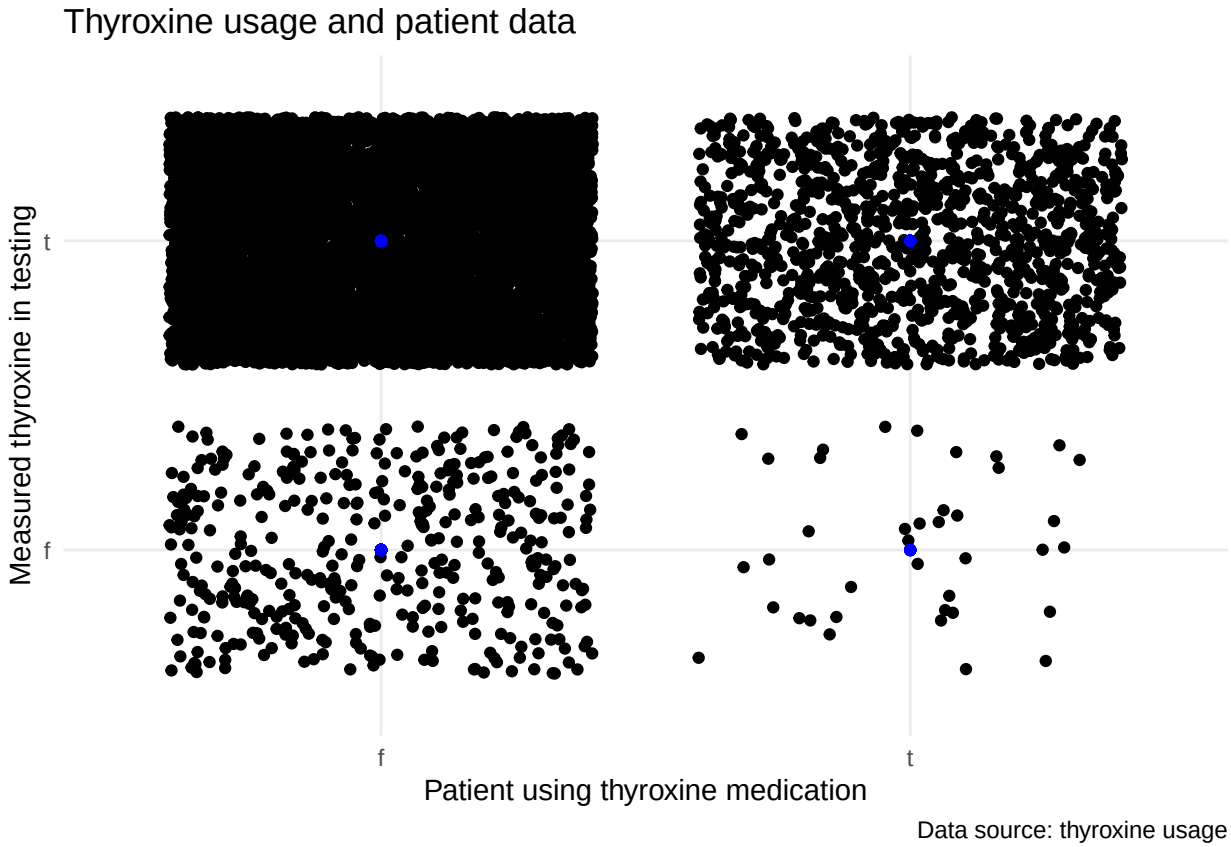
As shown above, not everyone in the dataset is a patient. Some diagnosis are more common than others, a few cases are really rare on a scale of nine thousand instances.

Another common medicine that helps thyroid patients with their problems is thyroxine. This is additional TT4 hormones that will help the body maintain or keep the cycle it is currently in. But how do these effects translate to the patient data set:

```
ggplot(data = my_data,
  mapping= aes(x = on_thyroxine, y= TT4_measured)) +
  geom_jitter(width = 0.4, height = 0.4) +
```

```
  geom_point(col= "blue", alpha = 0.2) +
  geom_smooth(method="loess", se=FALSE) +
  ggtitle("Thyroxine usage and patient data") +
  xlab("Patient using thyroxine medication") +
  ylab("Measured thyroxine in testing") +
  labs(caption = "Data source: thyroxine usage") +
  theme_minimal()
```

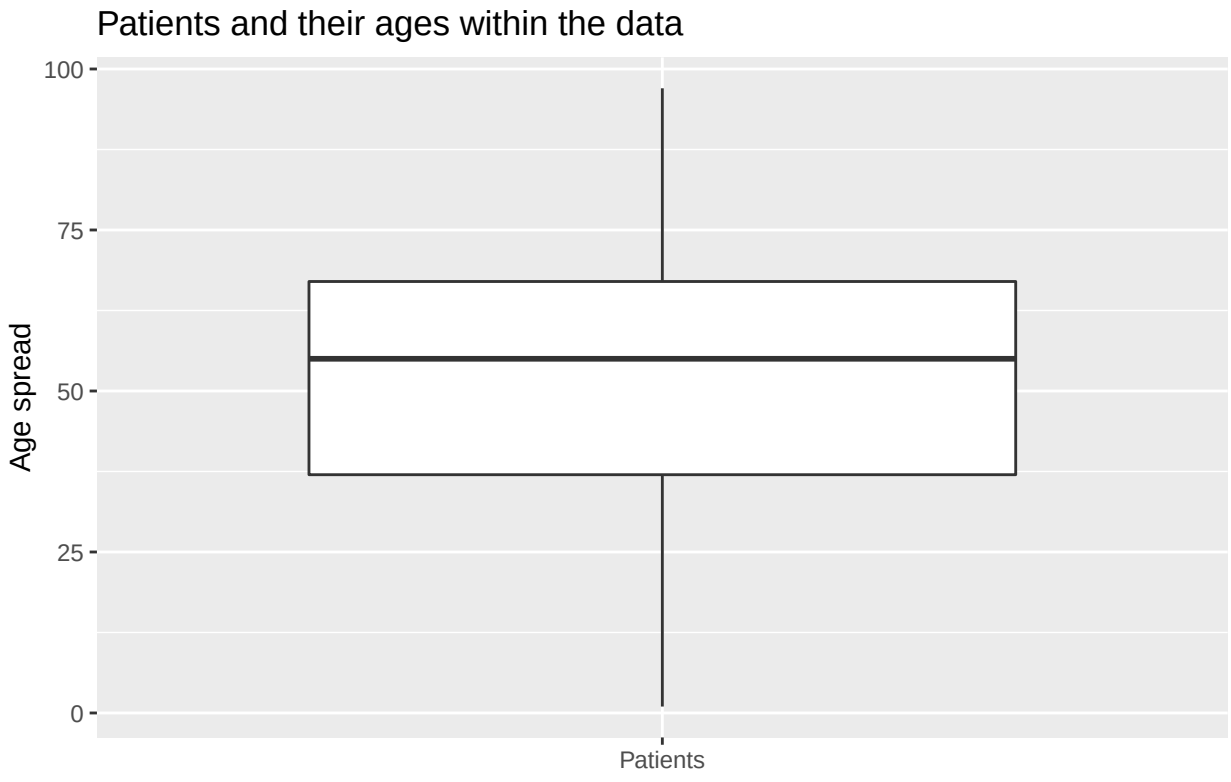
```
## `geom_smooth()` using formula 'y ~ x'
```



As seen, the patients that use thyroxine actually translates really well to the data. Due to the scope of the data and the big amounts of instances, it might look insignificant. But medication proves results in our data.

Overall the data looks really good, but if you look a bit closer, there is something strange going on in the data. The age column has some outliers, 3 to be exact. Two persons are in their nine thousands' and one person is over four hundred years old. There must have gone something wrong with these values, since it's biologically impossible to be this age. These values need to be either removed or edited.

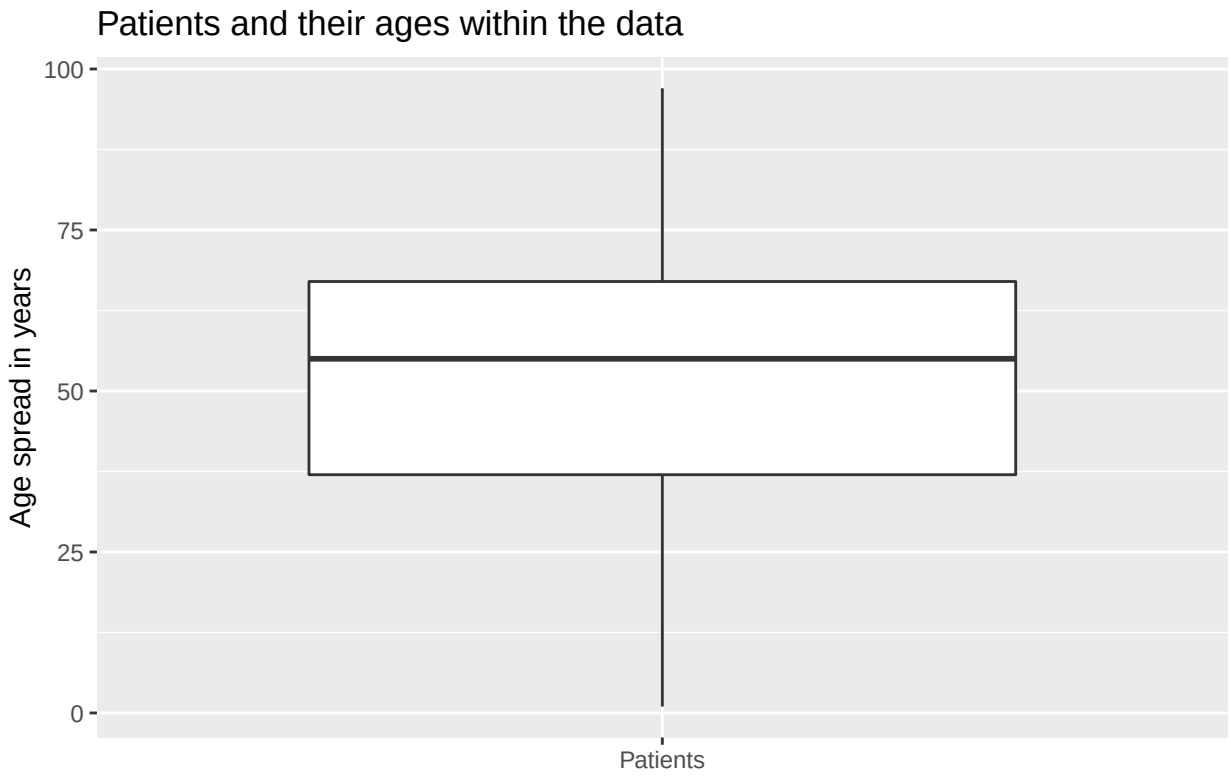
```
ggplot(my_data, aes(y = age, x=as.factor("Patients"))) +
  geom_boxplot() +
  ggtitle("Patients and their ages within the data") +
  xlab("") +
  ylab("Age spread") +
  labs(caption = "Data source: patients' ages from the data")
```



The instances with unnatural ages are now edited to only use the first two digits. If someone would say fourty and a half, and entered it like 40.5; the values combined will be 405 wich enters the range of one of the instance's age. More on this in results.

Now that the unnatural ages have been modified, the data is now clean and seems accurate with the normal age of people. The best way to visualize this, is via a boxplot. To directly show the mean and the quartiles within the age distribution.

```
ggplot(my_data, aes(y = age, x=as.factor("Patients"))) +
  geom_boxplot() +
  ggtitle("Patients and their ages within the data") +
  xlab("") +
  ylab("Age spread in years") +
  labs(caption = "Data source: patients' ages from the data")
```

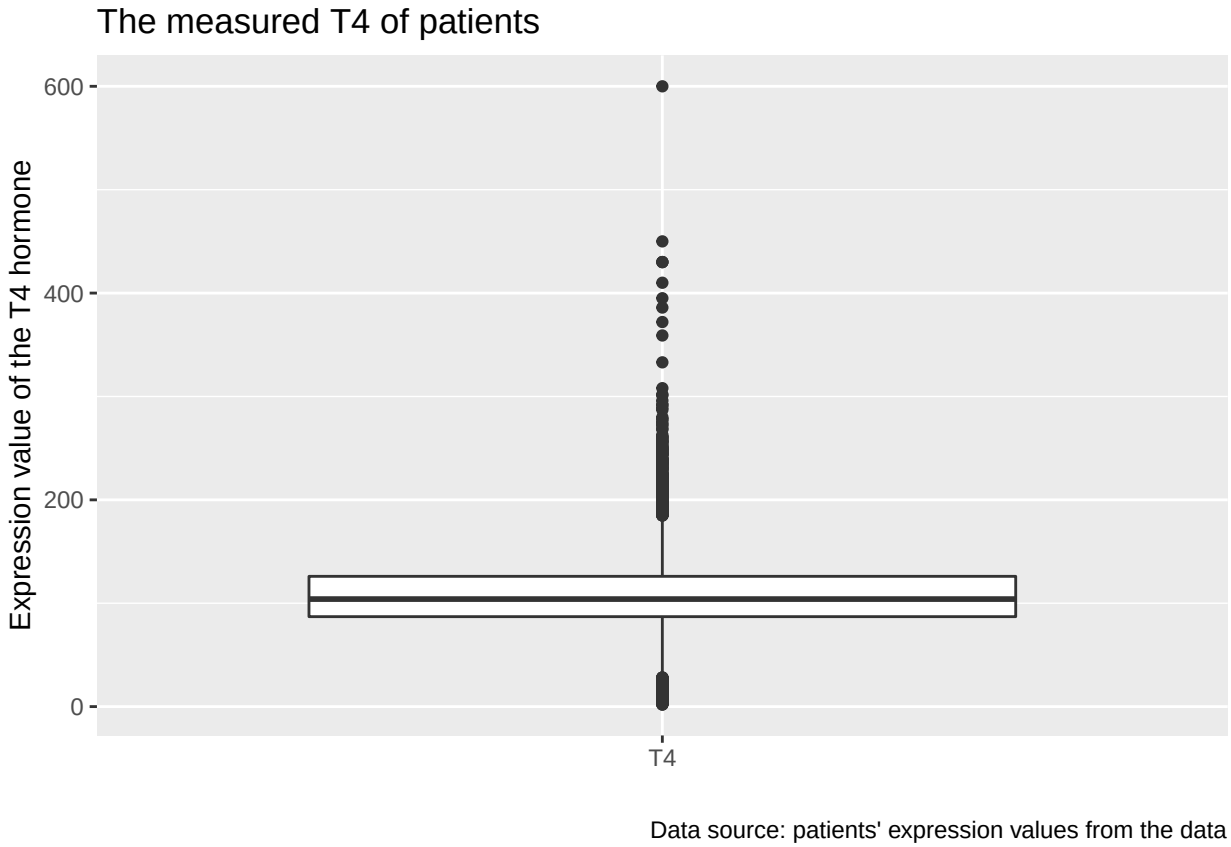


There are no extreme outliers any more and this is believable. Since it is biological impossible to have such extreme ages. Most of the instances have an age of round 54 years old. While the third quantile leans a bit more towards the older age, since it does not align with the 75 year line.

To finalize the analysis, there are the hormone columns. There are six of these columns, with numeric values and they visualize the expression value. With a boxplot all the additional information about the data will be showed.

```
ggplot(my_data, aes(y = TT4, x= as.factor("T4") )) +
  geom_boxplot() +
  ggtitle("The measured T4 of patients") +
  xlab("") +
  ylab("Expression value of the T4 hormone") +
  labs(caption = "Data source: patients' expression values from the data")
```

```
## Warning: Removed 442 rows containing non-finite values (stat_boxplot).
```



There are some outliers in this column, but because this is on the big scale of thousands of instances. This effect is not significant. For this reason, the outliers won't be normalized. Because the gained effect won't change the outcome.

Machine learning

This is the most important part of the whole process. To see if an algorithm can identify thyroid patients from their expression values.

Type 1 and type 2 errors

There are two types of errors, the type-1 error and type-2 errors. The type-1 error is simply a false alarm. Where the algorithm incorrectly said `Yes`, when it should've been a `No`. The type-2 error is a underestimation or a simply a miss, where the algorithm incorrectly said `No` where it should have `Yes`. In this case for type-1, it is a patients that is diagnosed sick that is actually healthy. For the type-2 error, it is a patient that is incorrectly declared healthy, while actually being sick.

Since the dataset has a lot of patients, where the sick patients are more important. Not only because of the minority but also the sick need to get diagnosed properly. Because around the 90% is healthy, the false positives need to be as low as possible. The patients need to be correctly identified, but the algorithm doesn't need to go too harsh on the healthy instances.

Algorithm

The overall result of the machine learning process is not valid. Due to the skewness of the data, ZeroR will get 97% correct by simply saying False. For OneR will make illogical models, for example "if age is not NA, sick is false." Due to the lack of actual patients in the dataset, the algorithm will won't do a good job on

actual patients. The J48 tree will make a simple model with everything resulting in False. NaiveBayes will try, but perform less than zeroR. The only way to get an acceptable score with an actual algorithm is by overfitting on the trainings data. Unfortunately, this won't work for new instances. So the data needs to be trimmed down a bit. Since the ratio is around 9000 healthy and 200 sick, the amount of healthy instances needs to be brought down. While the patient amount should get more weight.

The extra weight for the sick class, will be done by making a dedicated class column. So there is a sick column, and an additional class column. The only differences are that the NAs will be taken out, so there is a somewhat clean column. There is still a large number of healthy instances, to bring this down way more; around 60% of the healthy instances will be removed. Now the data has 3772 instances, where around 200 instances are sick. With the extra weight of another class column, the results are now a lot better.

With the newly made dataset, all algorithms are tested with 10-fold cross validation (not training data). ZeroR keeps performing good, ZeroR will now do 93% correct by simply saying healthy to everyone. But now OneR will do a better job and make an actual model, and will have 96% correct and will base the model on T3 expression. But the J48 tree will do 98.6% correct, with 10-fold cross validation and a minimum number of objects of 10. Picking a number higher will lower the correctly diagnosed sick instances. Picking a lower number will cause overfitting and resulting in more incorrectly diagnosed sick instances. So 10 seems to be a perfect option. Increasing or decreasing the minimal number of objects drastically, gives about the same results but different decimals. But there is some mild variation in correctly diagnosed sick instances. Which proves that this model is surprisingly functional. Due to this not being trainings data, this tree should be valid in a more realistic scenario. From the 231 patients, 28 are diagnosed incorrectly healthy. With the massive skewness of this data, only having 231 sick instances, this is a good result. The cleaned and trimmed data can be found under `data/thyroid_final.arff`.

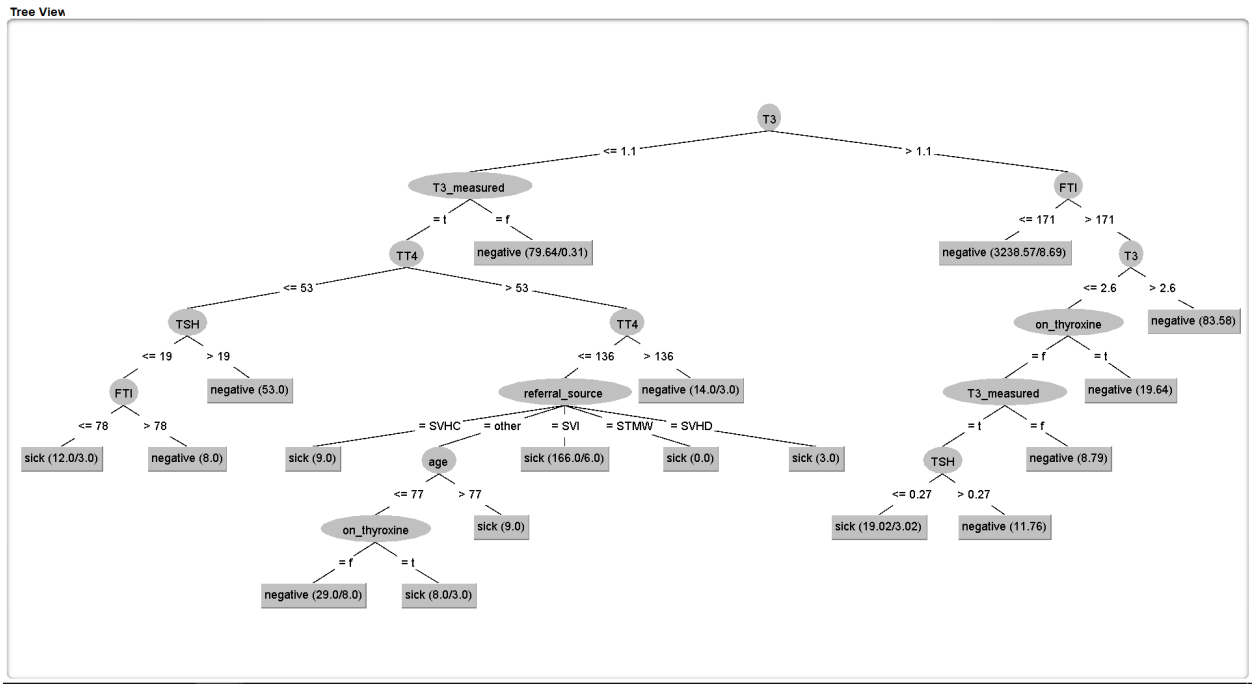


Figure 1: The J48 tree visualized by the Weka 3.8.5 Tree visualize function.

This algorithm performs the best, since it diagnoses the most sick instances correct without a lot of False Positives. The False Positives are a lot worse, since the instance is wrongfully judged healthy. A False Negative also has effects, but after a hypothetical medical investigation; the patient will be proven healthy. While a real patient's condition can worsen. The tree also seems logical, since it uses the most obvious columns. An actual doctor would also check hormone expression, then medicine usage and finally the external factors.

Algorithm performance

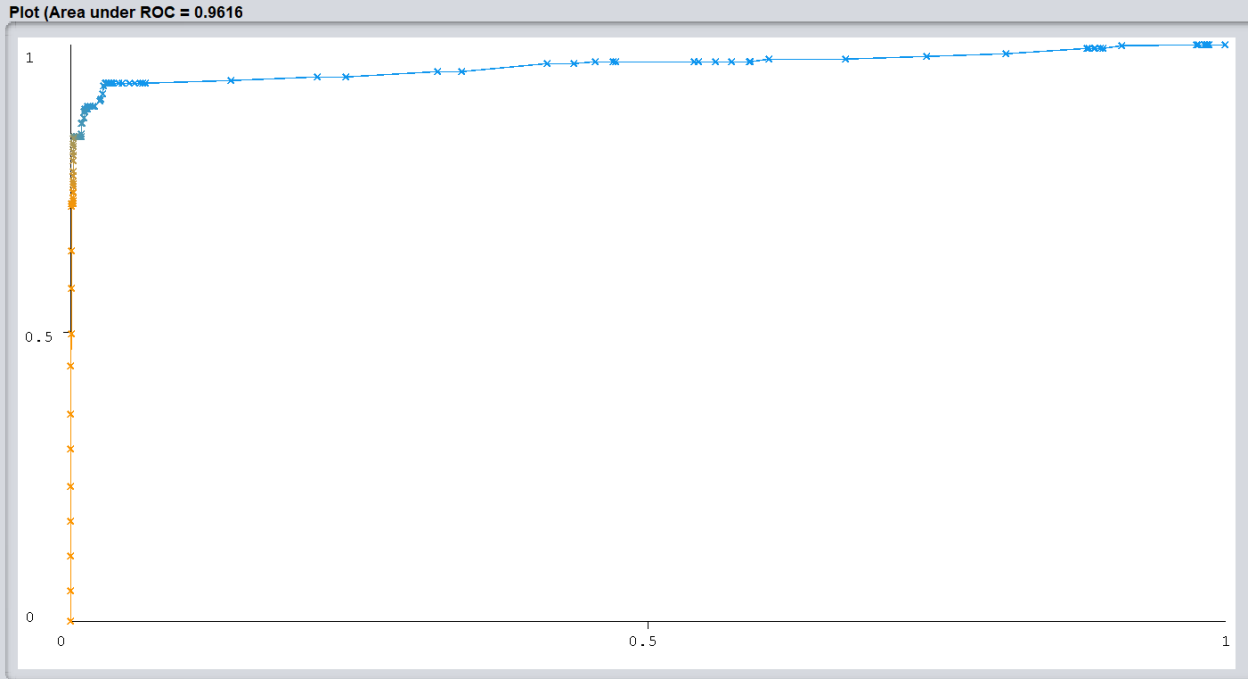
Multiple test results have been run, the most important factors are displayed under the columns.

```
Openend_ML_results <- readxl::read_xlsx("ML_results.xlsx")
print(Openend_ML_results)
```

```
## # A tibble: 7 x 8
##   algorithm_name speed accuracy error_rate true_positives false_positives
##   <chr>          <chr>    <dbl>      <dbl>          <dbl>           <dbl>
## 1 Zero R         0.01      0.94      0.062           3541             231
## 2 One R          0.01      0.96      0.04            3466              53
## 3 Naïve Bayes    0.02      0.93      0.07            3314              52
## 4 J48-Tree       0.14      0.99      0.01            3524              28
## 5 Ibk            0.00      0.96      0.04            3484              87
## 6 SMO            0.24      0.94      0.06            3540             231
## 7 Random Forest  0.42      0.98      0.02            3533              49
## # ... with 2 more variables: false_negatives <dbl>, true_negatives <dbl>
```

ROC-Curve

Running the J48-algorithm with 10-fold cross validation, using the optimal 10 minimal number of objects. With the Weka 3.8.5 Experimenter function, rerunning this test 100 times. Generating a total of 1000 trees,



Using the function 'read_xlsx" there is a simple way to open an Excel file and displaying the file as a tibble. The J48-Tree proves to be best, since it has the lowest amount of false negatives and false positives. On top of that, it also runs the best with true negatives.

References

[1] Boron, W. F., Boulpaep, E. L. (2012). *Medical Physiology Chapter 49, "Synthesis of Thyroid Hormones"* (2nd ed.). Elsevier/Saunders. ISBN 9781437717532.

[2] Basile, L. M. (z.d.). *What are T3, T4, and TSH?* EndocrineWeb. used on 23 September 2021, from https://www.endocrineweb.com/thyroid-what-are-t3-t4-tsh

[3] Shahid, A. H., Singh, M. P., Raj, R. K., Suman, R., Jawaid, D., Alam, M. (2019). *A Study on Label TSH, T3, T4U, TT4, FTI in Hyperthyroidism and Hypothyroidism using Machine Learning Techniques* 2019 International Conference on Communication and Electronics Systems (ICCES). Published. https://doi.org/10.1109/icces45898.2019.9002284

[4] James R Mulinda, Arthur B Chausmer, Francisco Talavera *Hypopituitary Hypopituitarism Causes, Symptoms and Treatment* 2018, January 3rd. EMedicineHealth. *https* : *//www.emedicinehealth.com/hypopituitary/article*$_e$*m.htm*