EDA Jorick Baron

Jorick Baron

```r
library(tidyr)
library(dplyr)
library(ggplot2)
library(gridExtra)
library(knitr)
library(kableExtra)
library(e1071)
library(foreign)
```

Start of research

Research question

How accurate can a model be trained to detect the difficult to diagnose pancreatic cancer utilising a patient's urine sample?

Codebook

In this EDA we will explore the data downloaded from here. For future reference we will describe the data in the codebook below.

```r
codebook <- read.delim("Data/codebook.csv", sep = ",")
kable(codebook, caption = "Codebook", align = "lcccr", booktabs = T) %>%
  kable_styling(latex_options = c("scale_down"))
```

Table 1: Codebook

Name	Fullname	Description	Type	Unit
sample_id	Sample ID	Unique string identifying each subject	string	NA
patient_cohort	Patient's Cohort	Cohort 1, previously used samples; Cohort 2, newly added samples	string	NA
sample_origin	Sample Origin	BPTB: Barts Pancreas Tissue Bank; ESP: Spanish National Cancer Research Centre; LIV: Liverpool University;UCL: University College	string	NA
age	Age	Age in years	int	years
sex	Sex	M = male, F = female	char	NA
diagnosis	Diagnosis (1=Control, 2=Benign, 3=PDAC)	1 = control, 2 = benign hepatobiliary disease; 3 = Pancreatic ductal adenocarcinoma, i.e. pancreatic cancer	int	NA
stage	Stage	For those with pancratic cancer, what stage was it? One of IA, IB, IIA, IIIB, III, IV	string	NA
benign_sample_diagnosis	Benign Samples Diagnosis	For those with a benign, non-cancerous diagnosis, what was the diagnosis?	string	NA
plasma_CA19_9	Plasma CA19-9 U/ml	Blood plasma levels of CA 19–9 monoclonal antibody that is often elevated in patients with pancreatic cancer.	float	plasma units/milliliter
creatinine	Creatinine mg/ml	Urinary biomarker of kidney function	float	mg/ml
LYVE1	LYVE1 ng/ml	Urinary levels of Lymphatic vessel endothelial hyaluronan receptor 1, a protein that may play a role in tumor metastasis	float	ng/ml
REG1B	REG1B ng/ml	Urinary levels of a protein that may be associated with pancreas regeneration.	float	ng/ml
TFF1	TFF1 ng/ml	Urinary levels of Trefoil Factor 1, which may be related to regeneration and repair of the urinary tract	float	ng/ml
REG1A	REG1A ng/ml	Urinary levels of a protein that may be associated with pancreas regeneration.	float	ng/ml

Table 2: The first records of the loaded data

sample_id	patient_cohort	sample_origin	age	sex	stage	benign_sample_diagnosis	plasma_CA19_9	creatinine	LYVE1	REG1B	TFF1	REG1A	has_cancer
S1	Cohort1	BPTB	33	F	NA	NA	11.7	1.83222	0.8932192	52.94884	654.2822	1262.000	0
S10	Cohort1	BPTB	81	F	NA	NA	NA	0.97266	2.0375850	94.46703	209.4882	228.407	0
S100	Cohort2	BPTB	51	M	NA	NA	7.0	0.78039	0.1455889	102.36600	461.1410	NA	0
S101	Cohort2	BPTB	61	M	NA	NA	8.0	0.70122	0.0028049	60.57900	142.9500	NA	0
S102	Cohort2	BPTB	62	M	NA	NA	9.0	0.21489	0.0008596	65.54000	41.0880	NA	0
S103	Cohort2	BPTB	53	M	NA	NA	NA	0.84825	0.0033930	62.12600	59.7930	NA	0

Loading data

First we will load in the data and to check if it has loaded in properly we look at the structure of the loaded data.

```r
data <- read.table(file="Data/source/Debernardi et al 2020 data.csv", sep = ",",
                   header = T, na.strings = "")
data$has_cancer <- ifelse(data$diagnosis == 3, 1, 0)
data$has_cancer <- factor(data$has_cancer)
data$sex <- factor(data$sex)
data <- subset(data, select = c(-diagnosis))

str(data)
```

```
## 'data.frame':    590 obs. of  14 variables:
##  $ sample_id              : chr  "S1" "S10" "S100" "S101" ...
##  $ patient_cohort         : chr  "Cohort1" "Cohort1" "Cohort2" "Cohort2" ...
##  $ sample_origin          : chr  "BPTB" "BPTB" "BPTB" "BPTB" ...
##  $ age                    : int  33 81 51 61 62 53 70 58 59 56 ...
##  $ sex                    : Factor w/ 2 levels "F","M": 1 1 2 2 2 2 2 1 1 1 ...
##  $ stage                  : chr  NA NA NA NA ...
##  $ benign_sample_diagnosis: chr  NA NA NA NA ...
##  $ plasma_CA19_9          : num  11.7 NA 7 8 9 NA NA 11 NA 24 ...
##  $ creatinine             : num  1.832 0.973 0.78 0.701 0.215 ...
##  $ LYVE1                  : num  0.89322 2.03758 0.14559 0.0028 0.00086 ...
##  $ REG1B                  : num  52.9 94.5 102.4 60.6 65.5 ...
##  $ TFF1                   : num  654.3 209.5 461.1 142.9 41.1 ...
##  $ REG1A                  : num  1262 228 NA NA NA ...
##  $ has_cancer             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

Thus far it seems to have loaded correctly.

We will also check the first few records to maybe catch some possible errors.

```r
kable(head(data), caption = "The first records of the loaded data", booktabs = T,
      align = "lcccccccccccr") %>%
  kable_styling(latex_options = c("scale_down"))
```

The data seems to have quite a few NAs, reading further into the description most NAs would be expected i.e. no stage if there is no cancer thus an NA.

NAs

The NAs in columns "plasma_CA19_9" and "REG1A" are supposed to be there because not every patient had been fully tested:
"REG1A … Only assessed in 306 patients", "plasma_CA19_9 … Only assessed in 350 patients" see Debernardi et al 2020 documentation.csv in the source files.
However to make sure everything is correct these numbers will be tested.

```
n_plasma_CA19_9 <- nrow(data) - sum(is.na(data$plasma_CA19_9))
n_REG1A <- nrow(data) - sum(is.na(data$REG1A))
paste("REG1A:", n_REG1A, "plasma_CA19_9:", n_plasma_CA19_9)
```

```
## [1] "REG1A: 306 plasma_CA19_9: 350"
```

These numbers are correct.

Are there more NAs?

```
sum(is.na(data[, c(1:5, 9:12)]))
```

```
## [1] 0
```

0 NAs remaining.

Data exploration

Distribution

Class label checking the different diagnoses should be in similar number to each has_cancer.

```
paste("Amount of patients without cancer:", nrow(subset(data, has_cancer == 0)))
```

```
## [1] "Amount of patients without cancer: 391"
```

```
paste("Amount of patients with cancer:", nrow(subset(data, has_cancer == 1)))
```

```
## [1] "Amount of patients with cancer: 199"
```

These are quite balanced and should not influence statistics.

Let's look at a summary of the data for a quick overview of the distributions.

```
summary(data[,c(4, 9:14)])
```

```
##       age          creatinine          LYVE1              REG1B
##  Min.   :26.00   Min.   :0.05655   Min.   : 0.000129   Min.   :   0.0011
##  1st Qu.:50.00   1st Qu.:0.37323   1st Qu.: 0.167179   1st Qu.:  10.7572
##  Median :60.00   Median :0.72384   Median : 1.649862   Median :  34.3034
##  Mean   :59.08   Mean   :0.85538   Mean   : 3.063530   Mean   : 111.7741
##  3rd Qu.:69.00   3rd Qu.:1.13948   3rd Qu.: 5.205037   3rd Qu.: 122.7410
##  Max.   :89.00   Max.   :4.11684   Max.   :23.890323   Max.   :1403.8976
##
##       TFF1               REG1A          has_cancer
##  Min.   :    0.005   Min.   :    0.00   0:391
##  1st Qu.:   43.961   1st Qu.:   80.69   1:199
##  Median :  259.874   Median :  208.54
##  Mean   :  597.869   Mean   :  735.28
##  3rd Qu.:  742.736   3rd Qu.:  649.00
##  Max.   :13344.300   Max.   :13200.00
##                      NA's   :284
```

Much of the data seems to be imbalanced with outliers.

Now let's take a closer look at the data itself using box-plots.

```
p1<- ggplot(data=data)+
  geom_boxplot(mapping = aes(x = "",
                             y = age))+
  ylab("age in years") +
  xlab(NULL)

p2<- ggplot(data=data)+
  geom_boxplot(mapping = aes(x = "",
                             y = creatinine))+
  ylab("creatinine in mg/ml") +
  xlab(NULL)

p3<- ggplot(data=data)+
  geom_boxplot(mapping = aes(x = "",
                             y = LYVE1))+
  ylab("LYVE1 in ng/ml") +
  xlab(NULL)+
  ylim(0,17)

p4<- ggplot(data=data)+
  geom_boxplot(mapping = aes(x = "",
                             y = REG1B))+
  ylab("REG1B in ng/ml") +
  xlab(NULL)+
  ylim(0,600)

p5<- ggplot(data=data)+
  geom_boxplot(mapping = aes(x = "",
                             y = TFF1))+
  ylab("TFF1 in ng/ml") +
  xlab(NULL)+
  ylim(0,5000)

p6<- ggplot(data=data)+
  geom_boxplot(mapping = aes(x = "",
                             y = REG1A))+
  ylab("REG1A in ng/ml") +
  xlab(NULL) +
  ylim(0,5000)

grid.arrange(p1, p2, p3, p4, p5, p6, nrow = 2)
```

There are many outliers to take a good look at the whiskers y-limits are in place. Still it's a lot, maybe adding another dimension can correct this.

To add this extra dimension let's look at the difference in diagnoses. To properly do this we will also assign levels to the has_cancer column in the dataframe.

```
my_colours <- c("cyan3" ,"coral2")

gp1 <- ggplot(data = data)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = age,
                             group=has_cancer,
                             fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                             "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
```

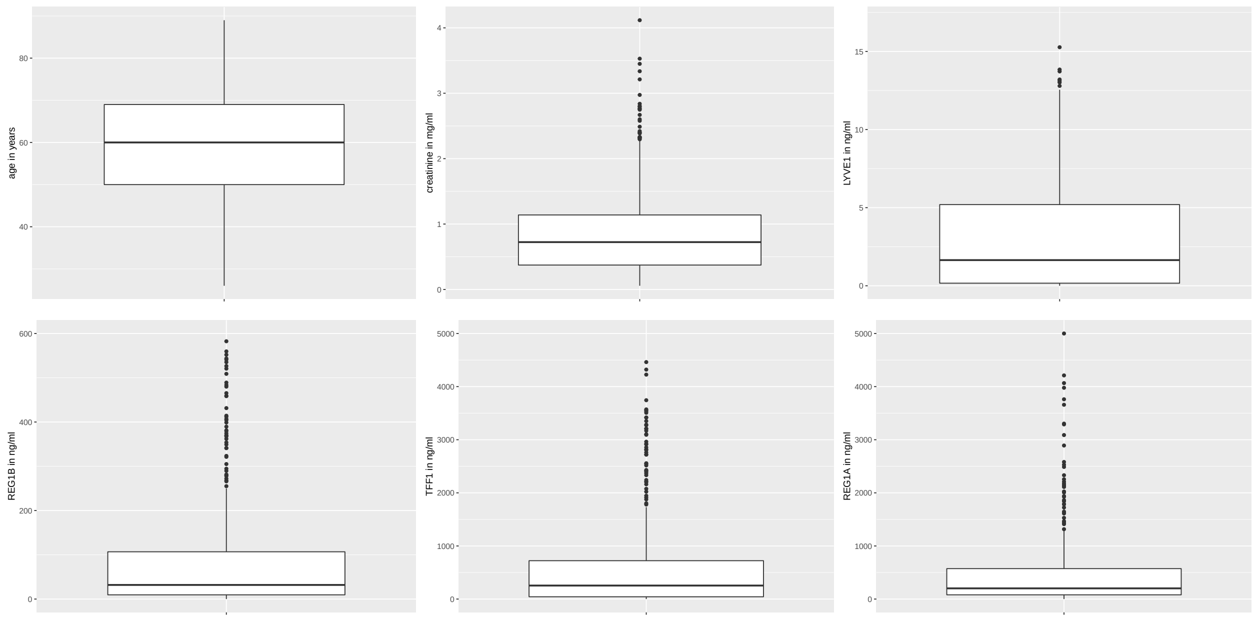

Figure 1: boxplots of diferent values

```
  xlab("has_cancer")+
  ylab("age in years")

gp2 <- ggplot(data = data)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = creatinine,
                             group=has_cancer,
                             fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                             "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("creatinine in mg/ml")

gp3 <- ggplot(data = data)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = LYVE1,
                             group=has_cancer,
                             fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                             "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("LIVE1 in ng/ml")+
  ylim(0,17)

gp4 <- ggplot(data = data)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = REG1B,
                             group=has_cancer,
                            fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                            "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("REG1B in ng/ml")+
  ylim(0,600)

gp5 <- ggplot(data = data)+
  geom_boxplot(mapping = aes(x = has_cancer,
                            y = TFF1,
                            group=has_cancer,
                            fill=has_cancer))+
  scale_x_discrete(labels=c("control",
                            "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("TFF1 in ng/ml")+
  ylim(0,5000)

gp6 <- ggplot(data = data)+
  geom_boxplot(mapping = aes(x = has_cancer,
                            y = REG1A,
                            group=has_cancer,
                            fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                            "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("REG1A in ng/ml")+
  ylim(0,5000)

grid.arrange(gp1, gp2, gp3, gp4, gp5, gp6, nrow = 2)
```

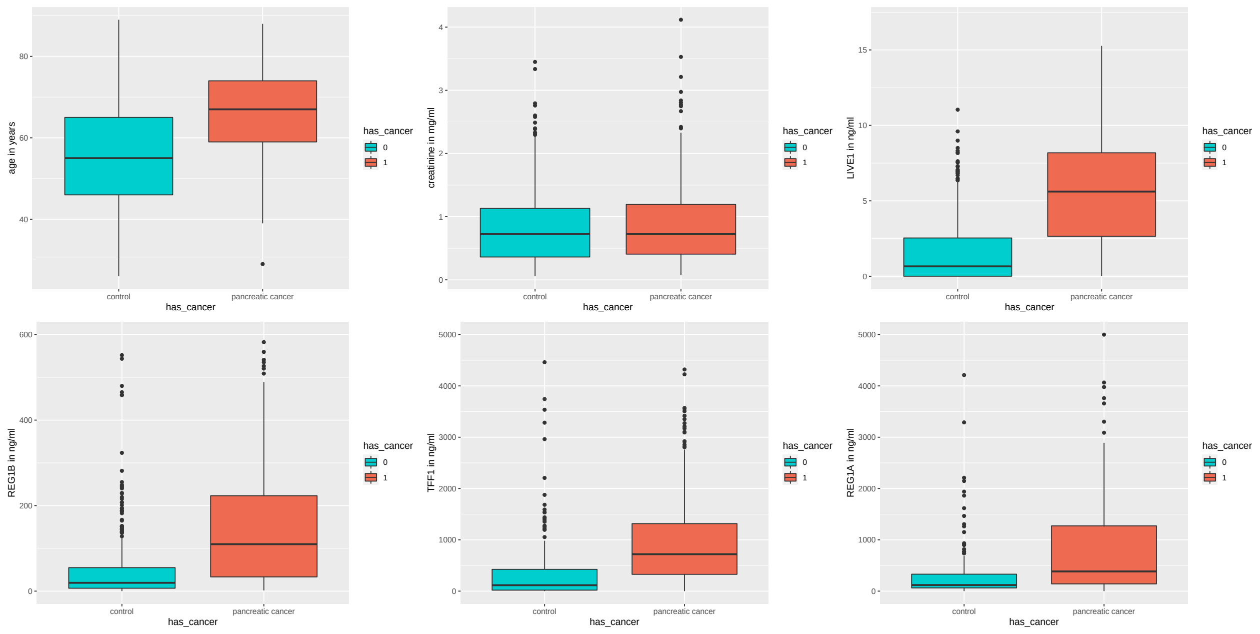

Figure 2: boxplots with added dimension (has_cancer)

The data still has many outliers but by many columns a pattern does emerge.

Log transformation

Let's use statistical tests to test the skewness to see how imbalanced the data is.

```r
s1 <- skewness(data$age)
s2 <- skewness(data$creatinine)
s3 <- skewness(data$LYVE1)
s4 <- skewness(data$REG1B)
s5 <- skewness(data$TFF1)
s6 <- skewness(data$REG1A, na.rm = T)
```

Table 3: Results of skewness test.

Variable	Skewness	Interpretation
age	-0.2157312	Fairly symmetrical
creatinine	1.4589654	Greatly positively skewed
LYVE1	1.3869334	Greatly positively skewed
REG1B	3.3169925	Greatly positively skewed
TFF1	5.1321035	Greatly positively skewed
REG1A	4.4254038	Greatly positively skewed

Here we see that everything is greatly skewed except age.

A way of dealing with this skewness is to apply a log transformation on the data due to the high positively skewed data.

```r
trans <- as.data.frame(log2(data$creatinine))
names(trans) <- "creatinine"
trans$LYVE1 <- log2(data$LYVE1)
trans$REG1B <- log2(data$REG1B)
trans$TFF1 <- log2(data$TFF1)
trans$REG1A <- log2(data$REG1A + 1)
trans$has_cancer <- data$has_cancer
kable(head(trans), caption = "the first few records of the log2 transformed data",
      align = "lccccr", booktabs = T) %>%
  kable_styling(latex_options = c("HOLD_position"))
```

Table 4: the first few records of the log2 transformed data

creatinine	LYVE1	REG1B	TFF1	REG1A	has_cancer
0.8735927	-0.1629138	5.726527	9.353769	10.302639	0
-0.0399925	1.0268602	6.561739	7.710725	7.841766	0
-0.3577328	-2.7800277	6.677593	8.849064	NA	0
-0.5120610	-8.4778452	5.920746	7.159367	NA	0
-2.2183297	-10.1841140	6.034304	5.360645	NA	0
-0.2374386	-8.2032229	5.957125	5.901905	NA	0

Now having transformed the data lets see how this influences the distribution.

```r
tgp1 <- ggplot(data = trans)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = creatinine,
                             group=has_cancer,
                             fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                             "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("creatinine in mg/ml (log2 transformed)")

tgp2 <- ggplot(data = trans)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = LYVE1,
                             group=has_cancer,
                             fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                             "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("LIVE1 in ng/ml (log2 transformed)")

tgp3 <- ggplot(data = trans)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = REG1B,
                             group=has_cancer,
                             fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                                 "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("REG1B in ng/ml (log2 transformed)")

tgp4 <- ggplot(data = trans)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = TFF1,
                             group=has_cancer,
                             fill=has_cancer))+
  scale_x_discrete(labels=c("control",
                             "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("TFF1 in ng/ml (log2 transformed)")

tgp5 <- ggplot(data = trans)+
  geom_boxplot(mapping = aes(x = has_cancer,
                             y = REG1A,
                             group=has_cancer,
                             fill=has_cancer))+
    scale_x_discrete(labels=c("control",
                             "pancreatic cancer"))+
  scale_fill_manual(values=my_colours)+
  xlab("has_cancer")+
  ylab("REG1A in ng/ml (log2 transformed)")

grid.arrange(tgp1, tgp2, tgp3, tgp4, tgp5, nrow = 2)
```

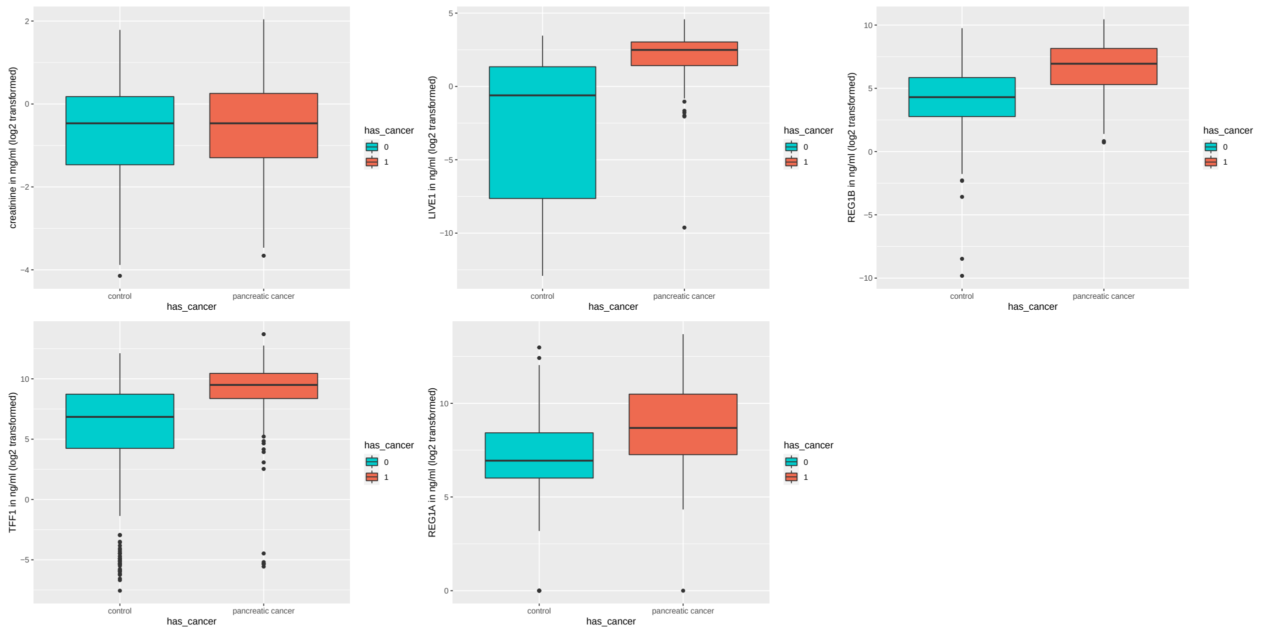

Figure 3: boxplots of transformed values

This data looks more normalized than before.

However it's good practice to test normality after transformations.

```r
swt1 <- shapiro.test(trans$creatinine)
qq1 <- ggplot(trans, aes(sample = creatinine, colour = has_cancer)) +
  stat_qq() +
  stat_qq_line() +
  scale_color_manual(values=my_colours)+
  ggtitle("creatinine")

swt2 <- shapiro.test(trans$LYVE1)
qq2 <- ggplot(trans, aes(sample = LYVE1, colour = has_cancer)) +
  stat_qq() +
  stat_qq_line() +
  scale_color_manual(values=my_colours)+
  ggtitle("LYVE1")

swt3 <- shapiro.test(trans$REG1B)
qq3 <- ggplot(trans, aes(sample = REG1B, colour = has_cancer)) +
  stat_qq() +
  stat_qq_line() +
  scale_color_manual(values=my_colours)+
  ggtitle("REG1B")

swt4 <- shapiro.test(trans$TFF1)
qq4 <- ggplot(trans, aes(sample = TFF1, colour = has_cancer)) +
  stat_qq() +
  stat_qq_line() +
  scale_color_manual(values=my_colours)+
  ggtitle("TFF1")

swt5 <- shapiro.test(trans$REG1A)
qq5 <- ggplot(trans, aes(sample = REG1A, colour = has_cancer)) +
  stat_qq() +
  stat_qq_line() +
  scale_color_manual(values=my_colours)+
  ggtitle("REG1A")

grid.arrange(qq1, qq2, qq3, qq4, qq5, nrow = 2)
```

The data is despite the transformation still not fully normalised however we can still continue but this should be kept this in mind in case of future problems.

Table 5: Results and interpertation of shapiro wiks test of normalcy

Variable	p-value	Interpretation
creatinine	1.2542643×10^{-6}	this data is not normally distributed
LYVE1	$1.7752934 \times 10^{-25}$	this data is not normally distributed
REG1B	$9.8879478 \times 10^{-10}$	this data is not normally distributed
TFF1	$1.0585334 \times 10^{-24}$	this data is not normally distributed
REG1A	5.5640181×10^{-6}	this data is not normally distributed

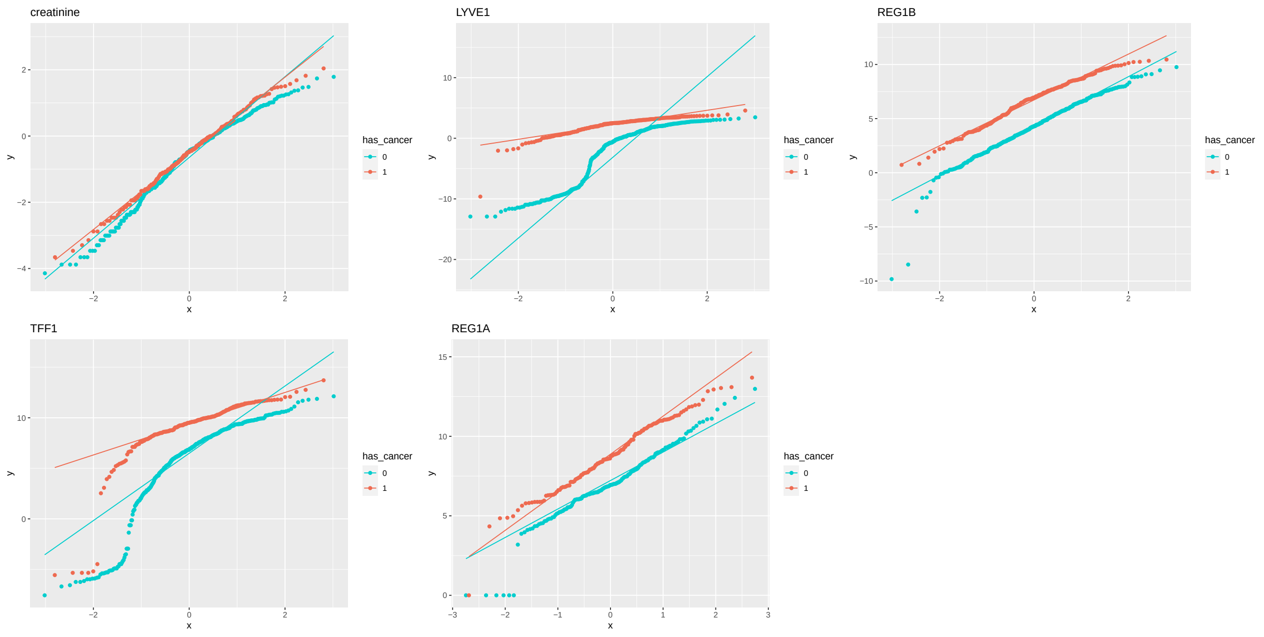

Figure 4: qqplots displaying normalcy

Correlations

Now using the transformed data let's create a new dataframe.

```
new_data <- cbind(data[4:5], trans)
new_data$sex <- factor(new_data$sex)
```

Using the new dataframe let's explore if the data is correlated.

```
matrix_data <- drop_na(new_data[,c(1, 3:7)])
cor_matrix <- cor(matrix_data)
heatmap(cor_matrix, scale = "none", col = heat.colors(6, rev = T), main = "Heatmap depicting correlatio
legend(x="right", legend=c("full","very strong", "strong", "moderate", "weak", "negligible"),fill=heat.
```

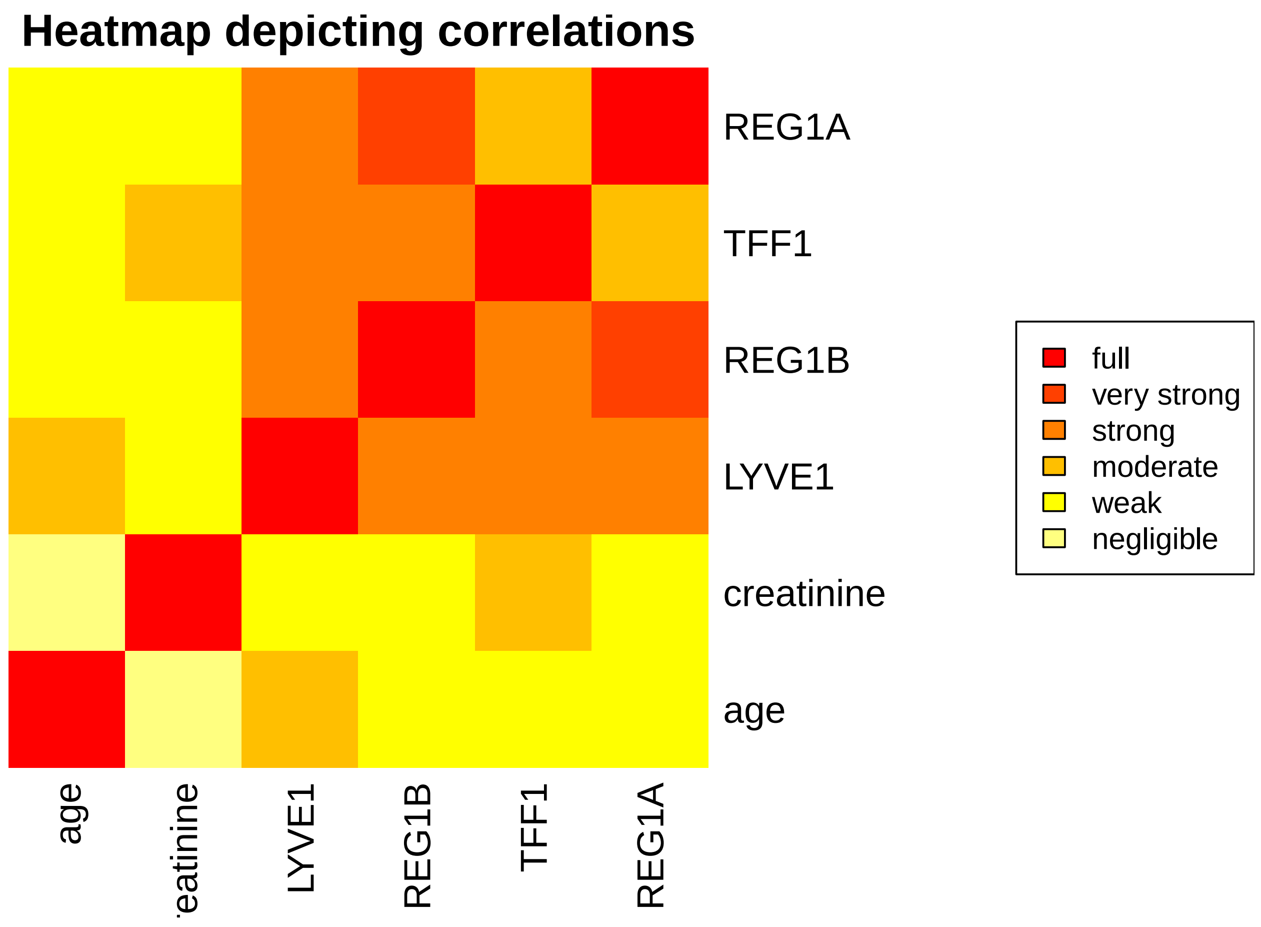

Figure 5: heatmap displaying correlation of values

REG1 A and B seem moderately correlated (0.7641084), otherwise no real strong correlation is observed.

Now we also should check if any variable is seemingly influential for the has_cancer so we can see later if the machine learning picks up on this.

```r
t1 <- t.test(new_data$age ~ new_data$has_cancer)
t2 <-t.test(new_data$creatinine ~ new_data$has_cancer)
t3 <-t.test(new_data$LYVE1 ~ new_data$has_cancer)
t4 <-t.test(new_data$REG1B ~ new_data$has_cancer)
t5 <-t.test(new_data$TFF1 ~ new_data$has_cancer)
t6 <-t.test(new_data$REG1A ~ new_data$has_cancer)
```

Table 6: T-test results and interpretation

Variable	p-value	Significant
Age	$1.2428989 \times 10^{-24}$	yes
Creatinine	0.1543498	no
LYVE1	$3.0097865 \times 10^{-54}$	yes
REG1B	$2.4404512 \times 10^{-33}$	yes
TFF1	$2.6006468 \times 10^{-23}$	yes
REG1A	2.687216×10^{-11}	yes

No p-value except Creatinine seems to be small enough to not be statistically significant. We will expect to see this in the model.

Output

Having explored the data and expanded the understanding of the variables to exploit them for machine learning and exterminating unhelpful variables from the data, it's time to write the data away to an Attribute Relation File Format (arff) and to train machine learning models on it.

```r
write.arff(new_data, "Data/data.arff")
```

Machine learning

Algorithm selection

After using the weka Experimenter trying out different algorithms the following results where produced.

```r
algores <- read.delim("Data/algores.csv", sep = ",")
kable(algores, caption = "Preformance of diferent algorithms", align = "lcccccccr", booktabs = T) %>%
  kable_styling(latex_options = c("scale_down", "HOLD_position"))
```

Table 7: Preformance of diferent algorithms

X	ZeroR	OneR..B40	J48..M35	IBk..K19	NaiveBayes	RandomForest	SMO	SimpleLogistic
Percent_correct	66.27	78.03	78.27	79.44	74.95	80.73	82.20	83.19
True_negative_rate	0.00	0.59	0.59	0.61	0.85	0.67	0.71	0.73
Area_under_ROC	0.50	0.73	0.80	0.86	0.85	0.88	0.79	0.89

note: that the OneR, J48 and IBk algorithms have been optimised beforehand.

The model using the SimpleLogistic algorithm is the most accurate, has the biggest area under the curve, and ranks second best in another important category: true positive rate. Thus the SimpleLogistic model will be used.