Team Project

José Ignacio Díez Ruiz 100487766
Carlos Roldán Piñero 100484904
Pablo Vidal Fernández 100483812

2023-01-17

Descriptive Analysis and Preprocessing

Reading the data and setting the NAs

Before starting the analysis, we need to make some preprocessing on our data. Let us start by loading it into memory and listing the names of the columns.

```
data <- read.csv("diabetes.csv")
colnames(data)
```

```
## [1] "Pregnancies"              "Glucose"
## [3] "BloodPressure"            "SkinThickness"
## [5] "Insulin"                  "BMI"
## [7] "DiabetesPedigreeFunction" "Age"
## [9] "Outcome"
```

Of these, our target variable is `Outcome`, which has two levels. For convenience, we transform it into a factor variable which R can trat accordingly.

```
data$Outcome <- factor(data$Outcome, c(0, 1), c("Negative", "Positive"))
```

Now we need to address a particularity of the chosen data: not-a-number (NaN) instances are encoded as zeros in variables where that value is imposible [1]. These are:

- Glucose

- BloodPressure

- SkinThickness

- Insulin

- BMI

In order for us to later trat them correctly, we need to manually change them to the existing `NA` type. As we do so, we record the number of NaNs instances in each of those variables. For convenience, we define a function.

```
set_nas <- function(data, fields) {
    percentage <- list()
    for (field in fields) {
        data[[field]][data[[field]] == 0]   <- NA
        percentage[[field]] <- 100 * sum(is.na(data[[field]])) / nrow(data)
```

---

[1] Remember we are dealing with medical data, not with artificial one. Hence, there are constraints on the values a variable may take.
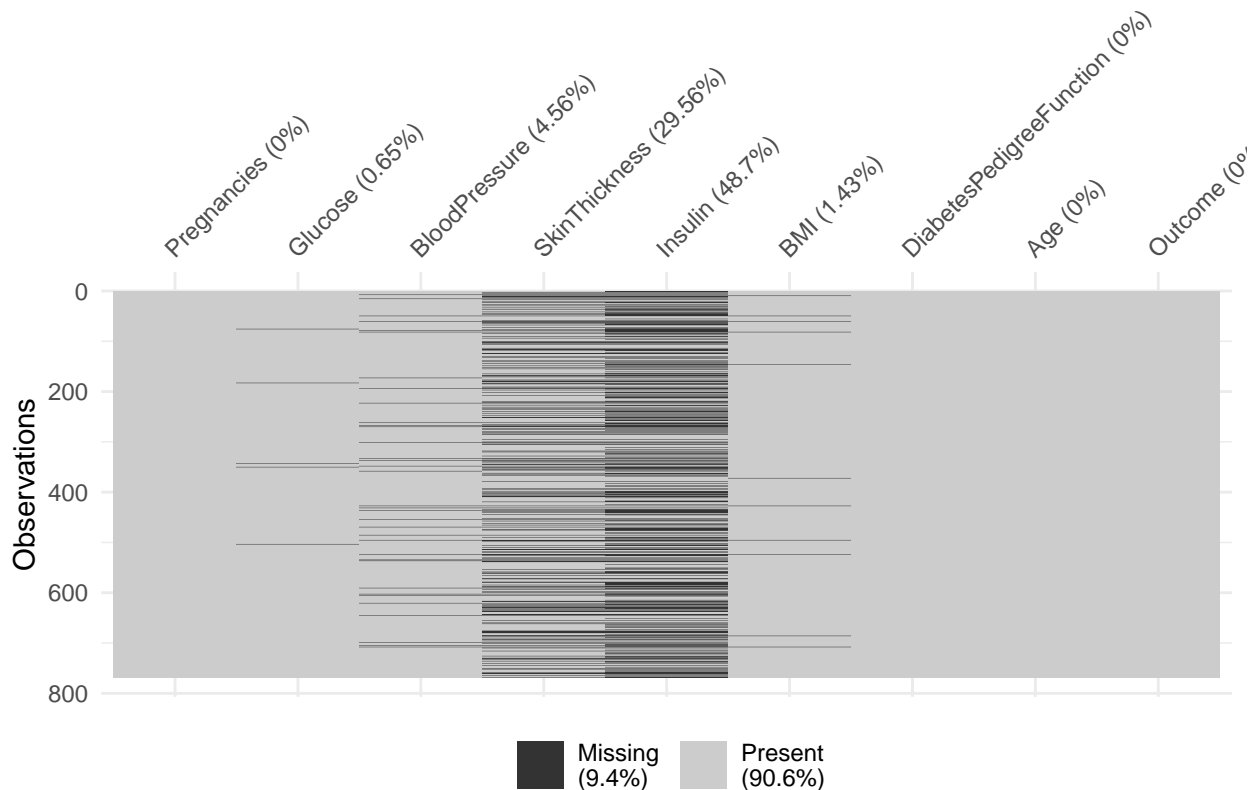
```
    }
    return(list(data = data, percentage = percentage))
}

# Correctly label NaNs
na_fields   <- c("Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI")
data_na     <- set_nas(data, na_fields)
data        <- data_na$data
percentages <- data_na$percentage

# Visualize them
vis_miss(data)
```


Now the next logical step is to impute those NaN values. We do have some concern about the imputation of the "Insulin" variable which is almost half-filled of NaNs. Nevertheless, we decide to impute it. On the followed strategy, we use the "Predictive Mean Matching Imputation" (PMM in short) as it behaves much more robustly than naive mean or median imputations.

```
data_im <- mice(data, m = 1, method = "pmm")
```

```
##
##  iter imp variable
##   1   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   2   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   3   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   4   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
##   5   1  Glucose  BloodPressure  SkinThickness  Insulin  BMI
```

```
data    <- complete(data_im)
```

Visualization of the data

Before we proceed any further, we are going to describe our data. First, we look individually to each of the attributes, we see both visually and with with a two-sample Wilcox-test whether there is significant difference between the two groups (having or not diabetes), and if some transformation may be desirable to ensure normality compatibility. For that purpose, we define the following function.

```r
histogram_by_groups <- function(data, var, label = NULL) {
    stat_t <- wilcox.test(as.formula(paste(var, "~ Outcome")), data)
    data0  <- data[data$Outcome == "Negative", ]
    data1  <- data[data$Outcome == "Positive", ]
    if (is.null(label)) {
        label <- var
    }
    p <- ggplot(data0, aes(x = eval(parse(text = var)))) +
        geom_histogram(
            aes(y = after_stat(count / sum(count)), fill = "Negative"),
            bins = 10, colour = "white", alpha = 0.8, boundary = 0
            ) +
        geom_histogram(data = data1,
            aes(
                x = eval(parse(text = var)),
                y = after_stat(count / sum(count)), fill = "Positive"
                ),
            bins = 10, colour = "white",
            alpha = 0.5, boundary = 0, inherit.aes = FALSE) +
            theme_bw() +
            scale_fill_manual(
                name = "",
                breaks = c("Positive", "Negative"),
                values = c("Positive" = "deeppink4", "Negative" = "pink2")
            ) +
            xlab(label) + ylab("Relative frequency") + ggtitle(label) +
            geom_vline(xintercept = mean(data1[[var]]), colour = "deeppink4") +
            geom_vline(xintercept = mean(data0[[var]]), colour = "pink2")
    p + annotate(
            "text",
            x = 0.9 * max(data1[var]),
            y = 0.9 * max(ggplot_build(p)$data[[1]]["y"]),
            label = sprintf("p-value = %.4e", stat_t$p.value),
            size = 3
        )
}
```

Let us start with the number of pregnancies. We can see that people who have diabetes have had more pregnancies than those who do not have diabetes. We see that it seems to be somewhat based on the p-value alone. We also note it exhibits a heavily right-skewed behaviour. As such, a logarithmic transformation would make sense to get a distribution more compatible with the normal one. Nonetheless, a problem here arises in dealing with the null values [2]. For that purpose we shift the variable by one unit. As a consistency measure, we will apply this shift to all variables we log-transform.
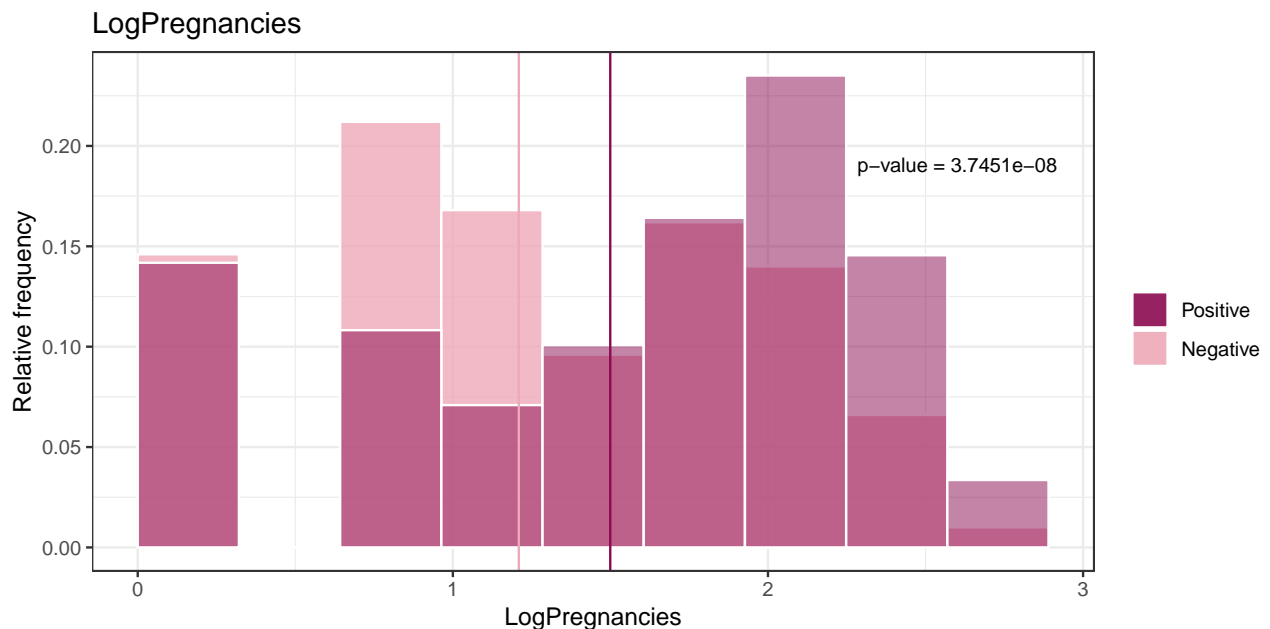
---

[2]Remember the domain of the logarithmic function is $(0, \infty)$.

```r
histogram_by_groups(data, "Pregnancies")
```
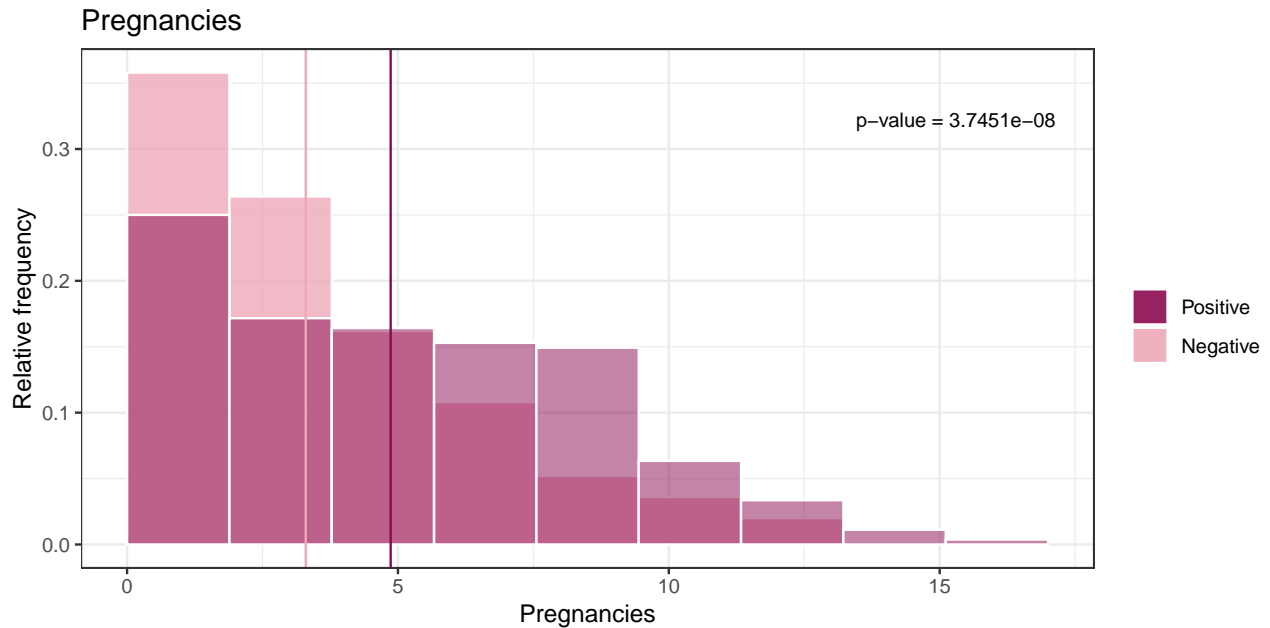
Pregnancies

The change as we see does help in obtaining a more centered distribution with which we may better apply the posterior analysis.

```r
data$LogPregnancies <- log(data$Pregnancies + 1)
histogram_by_groups(data, "LogPregnancies")
```

LogPregnancies

The next variable to visualize is the glucose. People with diabetes exhibit higher glucose values. The p-value is very small which indicates a highly significant difference between the two groups. We also observe that the distribution is already well-centered and resembles a normal distribution. Hence, we decide not to transform the data.

```r
histogram_by_groups(data, "Glucose")
```

Glucose
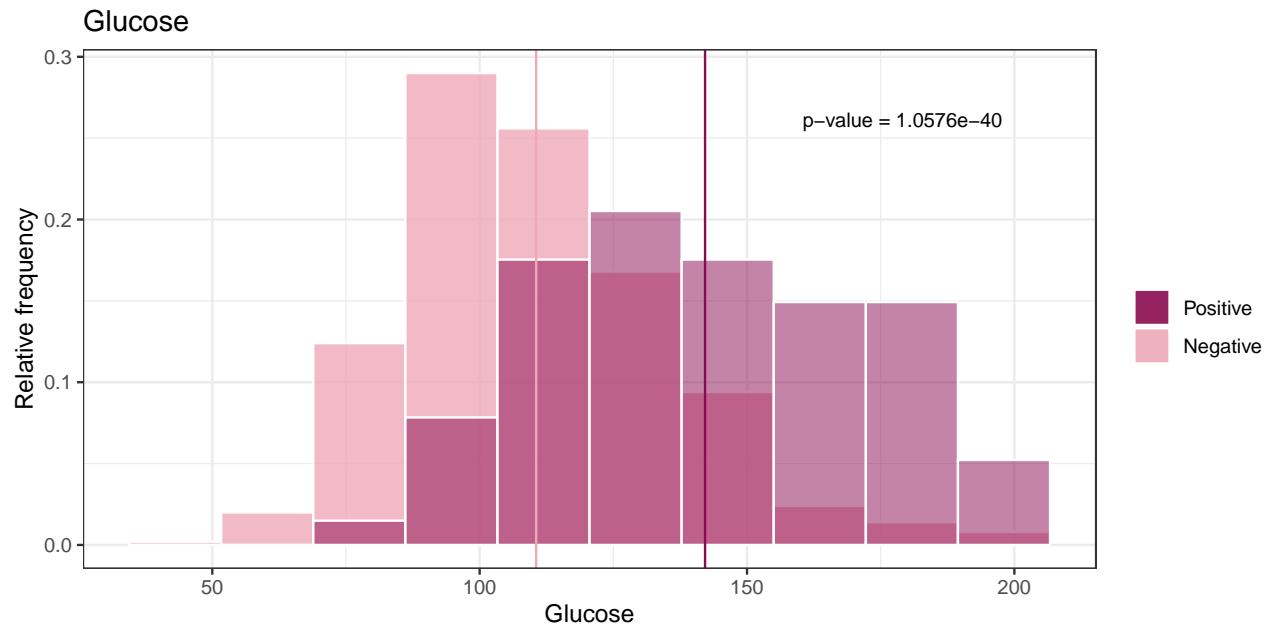
We move onwards to blood pressure. In this case, although the distributions appear as normal, there does not seem to exist a highly significant difference between the groups in contrast to what the p-value states, more so compared with the glucose variable. Nonetheless, there seem to be an slight indication of higher blood pressure for people with diabetes.

```
histogram_by_groups(data, "BloodPressure")
```
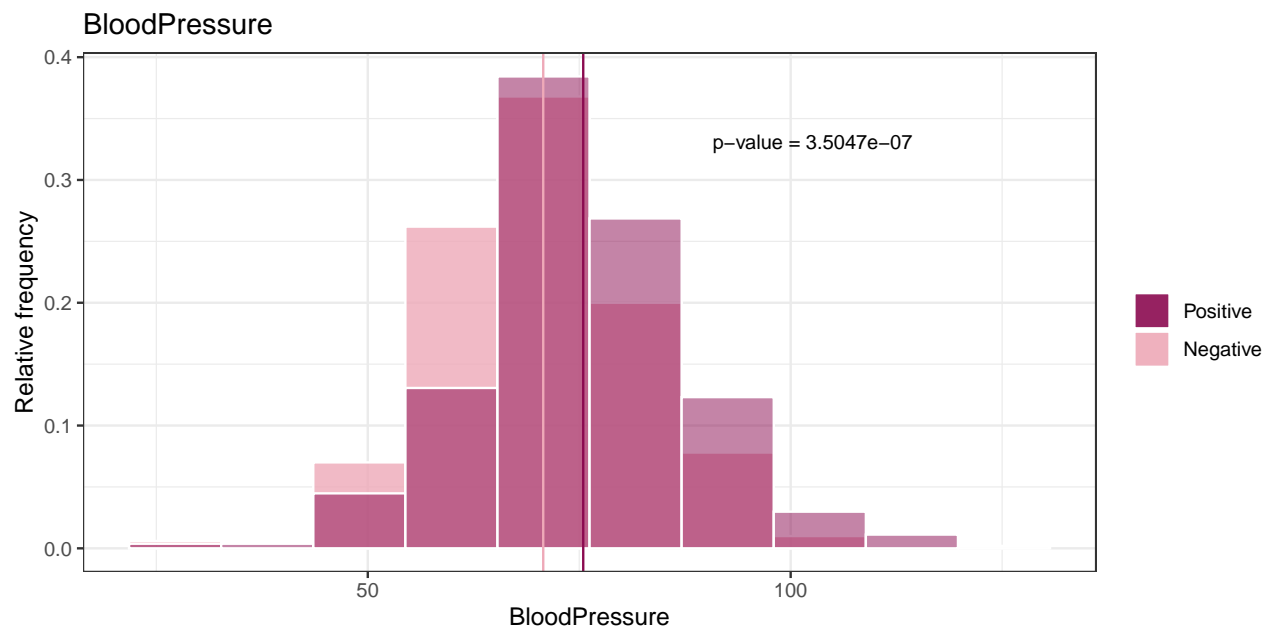
BloodPressure

Now skin thickness is interesting, because the distributions are visually to those of the blood pressure but the presence of outliers is appreciable. We will later deal with those but for now let us mantain this variable as it is. Note that the population with diabetes appear to exhibit a thicker skin.

```
histogram_by_groups(data, "SkinThickness")
```

SkinThickness
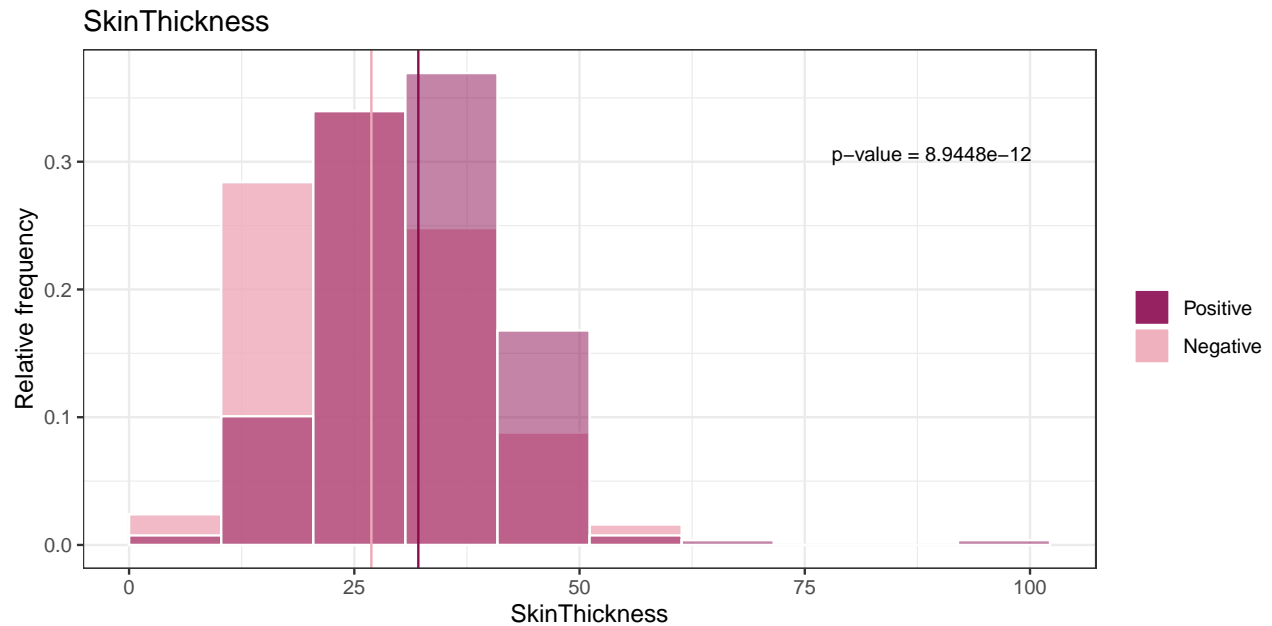
Insulin, as we may have expected from the name of the property itself, appears to be a relevant. The median of the distributions does indeed seem to differ, with the one for the diabetes population being slightly higher. It is also right-skewed, so we decide to log-transform it.

```
histogram_by_groups(data, "Insulin")
```
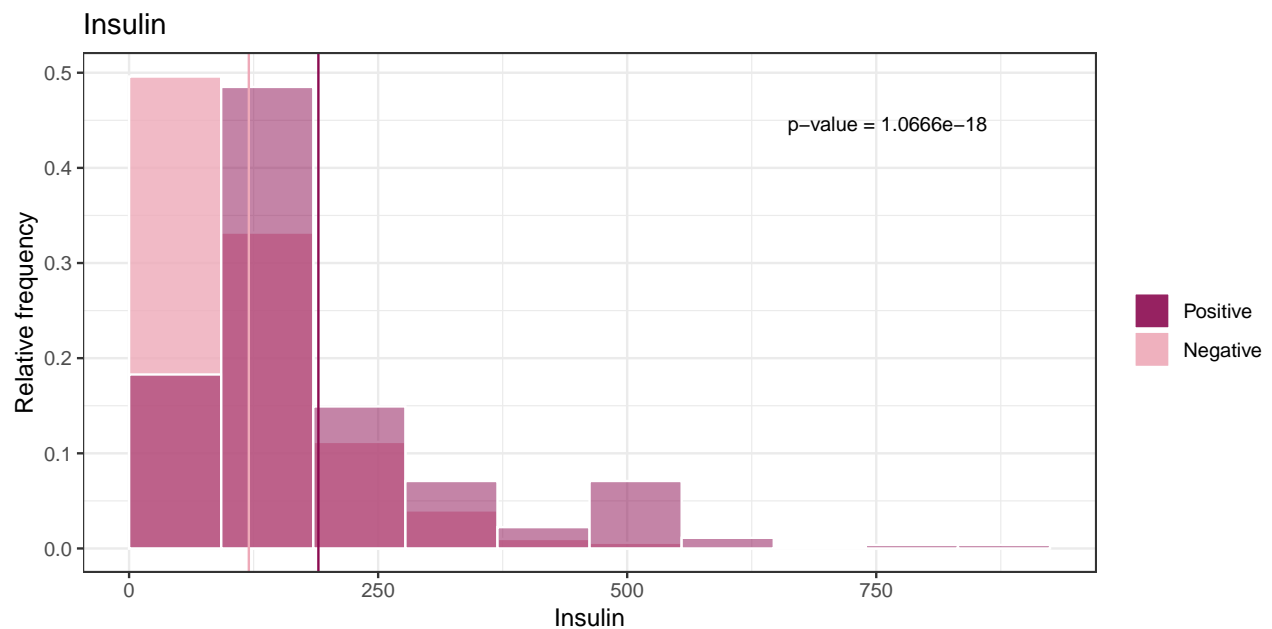
Insulin

It does appear that the transformation impproves the the symmetrization of the data, although some left-skewness appears. We will later see if outlier detection get to target those values or not.

```
data$LogInsulin <- log(data$Insulin + 1)
histogram_by_groups(data, "LogInsulin")
```

LogInsulin
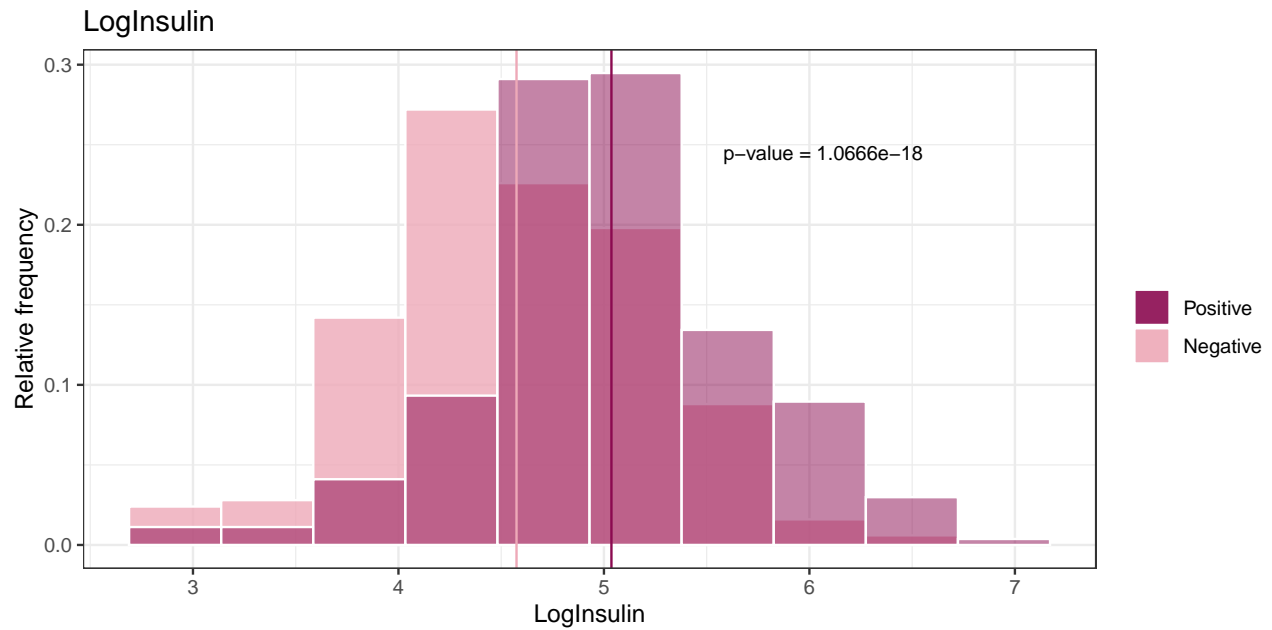
Body Mass Index (BMI) again exhibits this tendency of leaning towards a more right-skewed distribution. It does also follow the tendency of being slightly higher for people with diabetes. As such we log-transform to try and get a more normalized variable.

```
histogram_by_groups(data, "BMI")
```

BMI
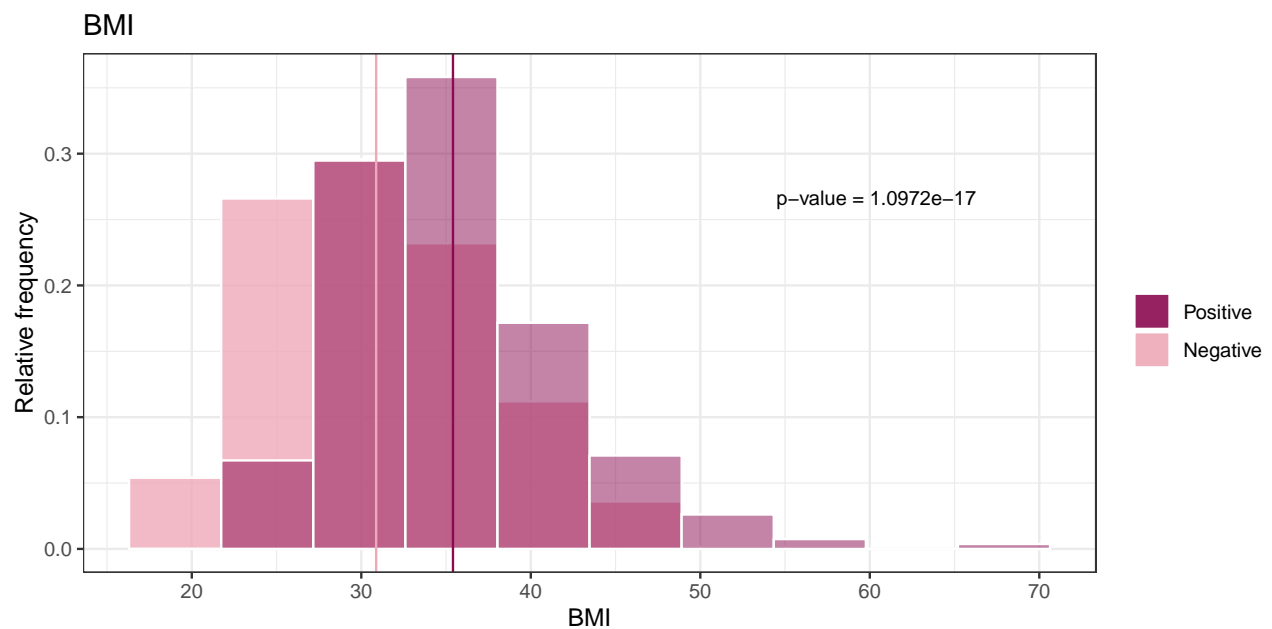
As we may visually judge, it is the case that the log transformation centers the data and provides a more normal-like distribution. There is some presence of seemingly outlying points to the right.

```
data$LogBMI <- log(data$BMI + 1)
histogram_by_groups(data, "LogBMI")
```

LogBMI
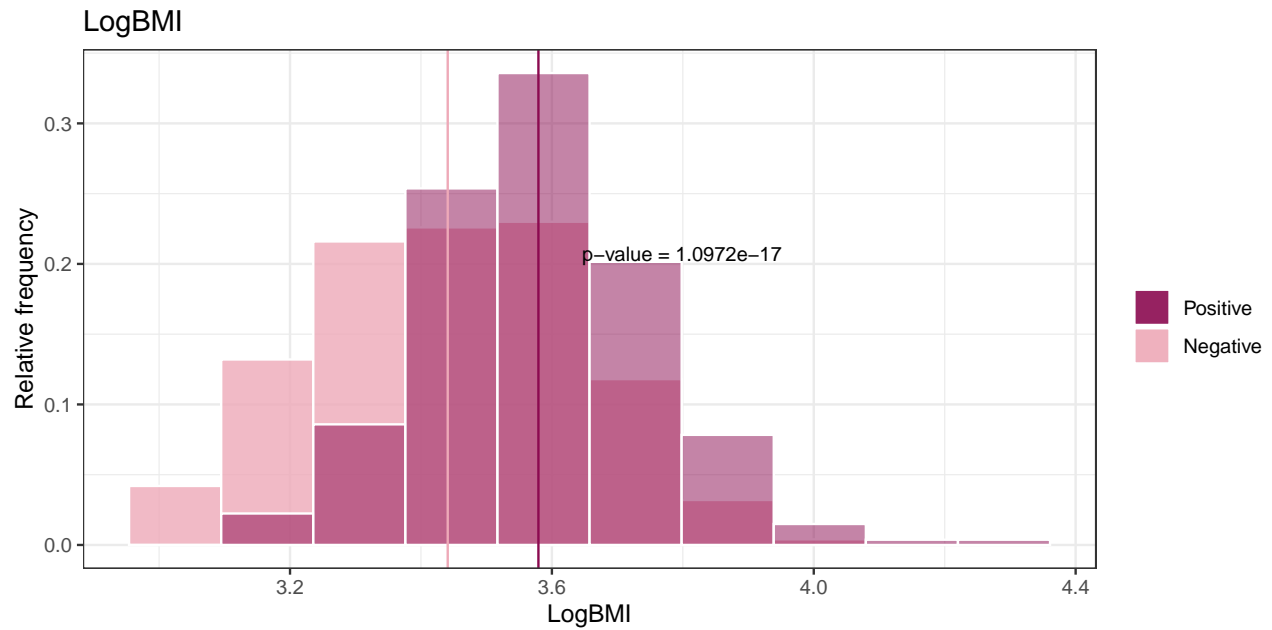
The Diabetes Pedigree Function (DPF) is the again a flagrant right-skewed. It again has higher values for the positive set. This is to be expected from the definition of this very function as a risk indication for diabetes. Let us try to log-transform it.

```
histogram_by_groups(data, "DiabetesPedigreeFunction", "DPF")
```

DPF
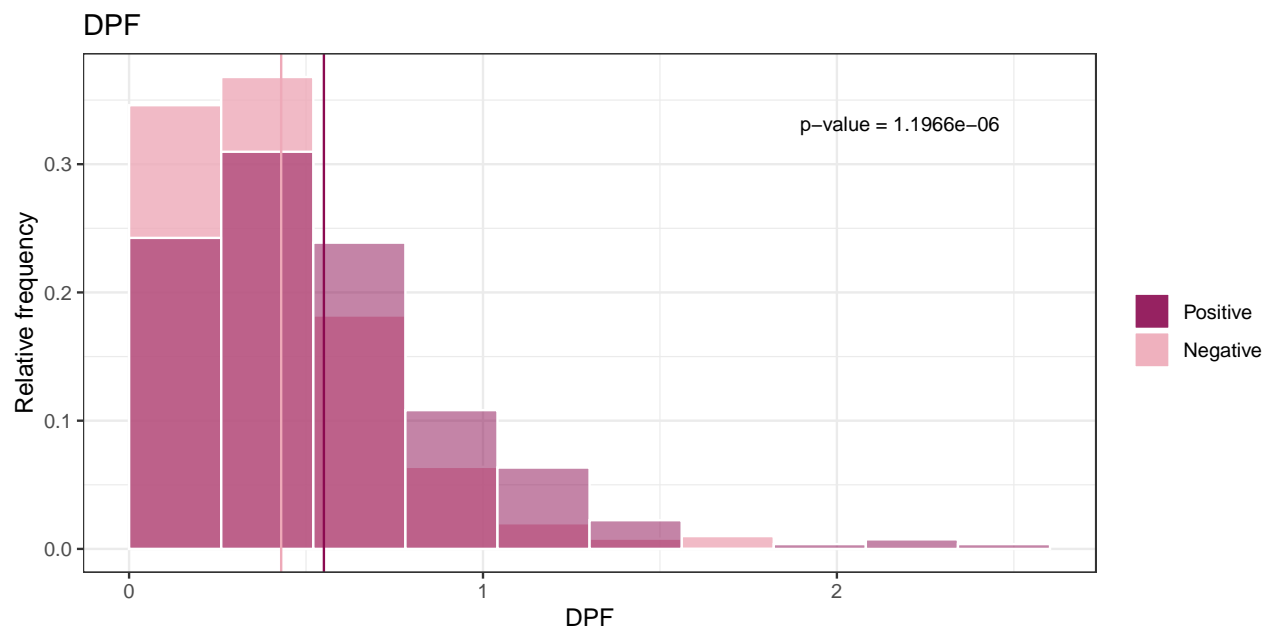
The improvement is highly noticeable. We retain thus this transformed variable.

```
data$LogDPF <- log(data$DiabetesPedigreeFunction + 1)
histogram_by_groups(data, "LogDPF")
```

LogDPF
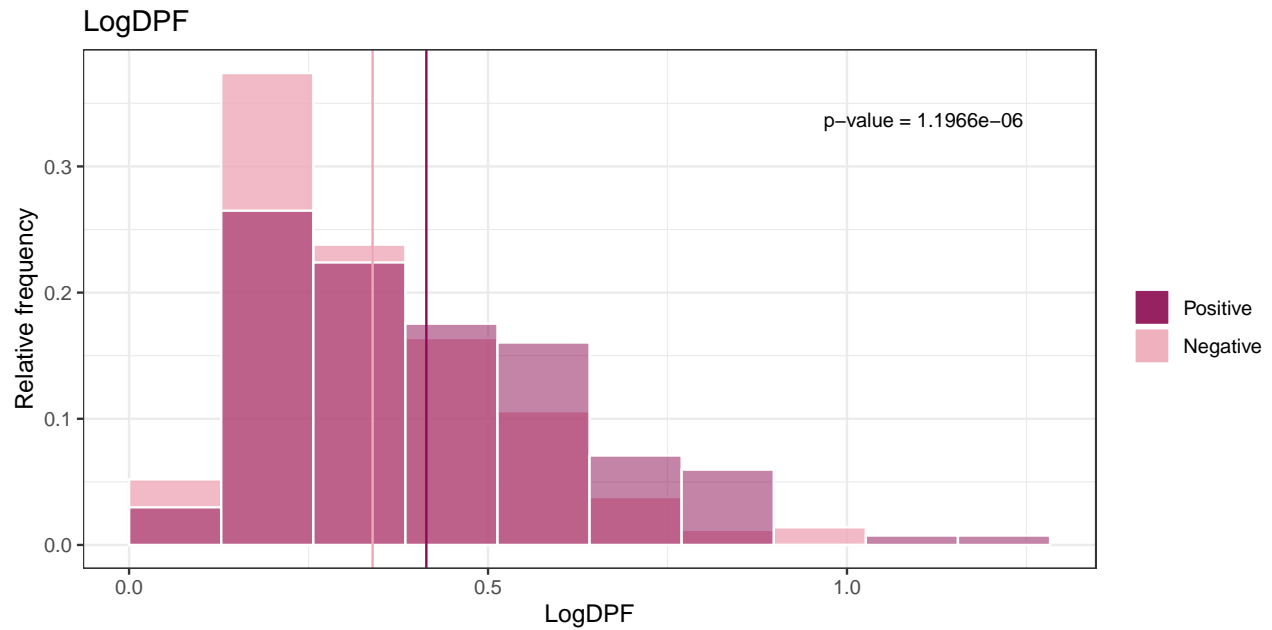
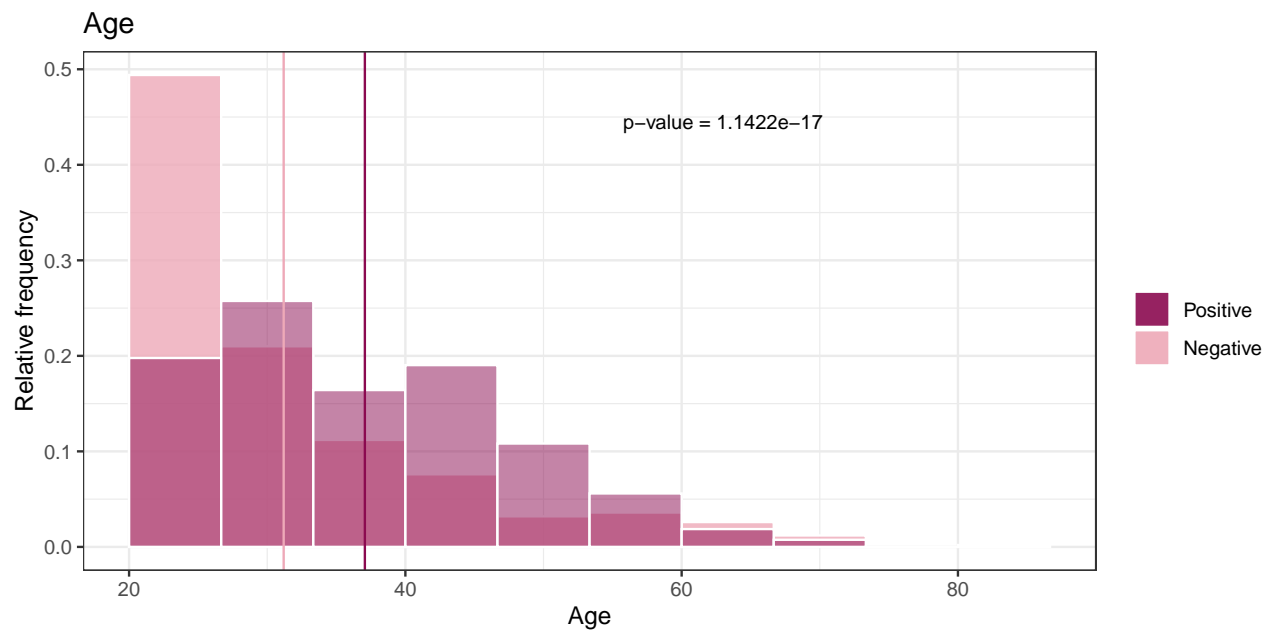We may note that young people, as with other illnesses have less tendency to suffer diabetes than the elders. The age is expected to exhibit a right-skewed distribution, as is indeed the case. In an attempt to improve the symmetry, we once again use logarithms to transform the variable.

```
histogram_by_groups(data, "Age")
```

Age

The transformation does help although not by much. This is a common problem of the age variable. We keep the transformation anyway as it does seem to help with the symmetry for the positive group.

```
data$LogAge <- log(data$Age + 1)
histogram_by_groups(data, "LogAge")
```
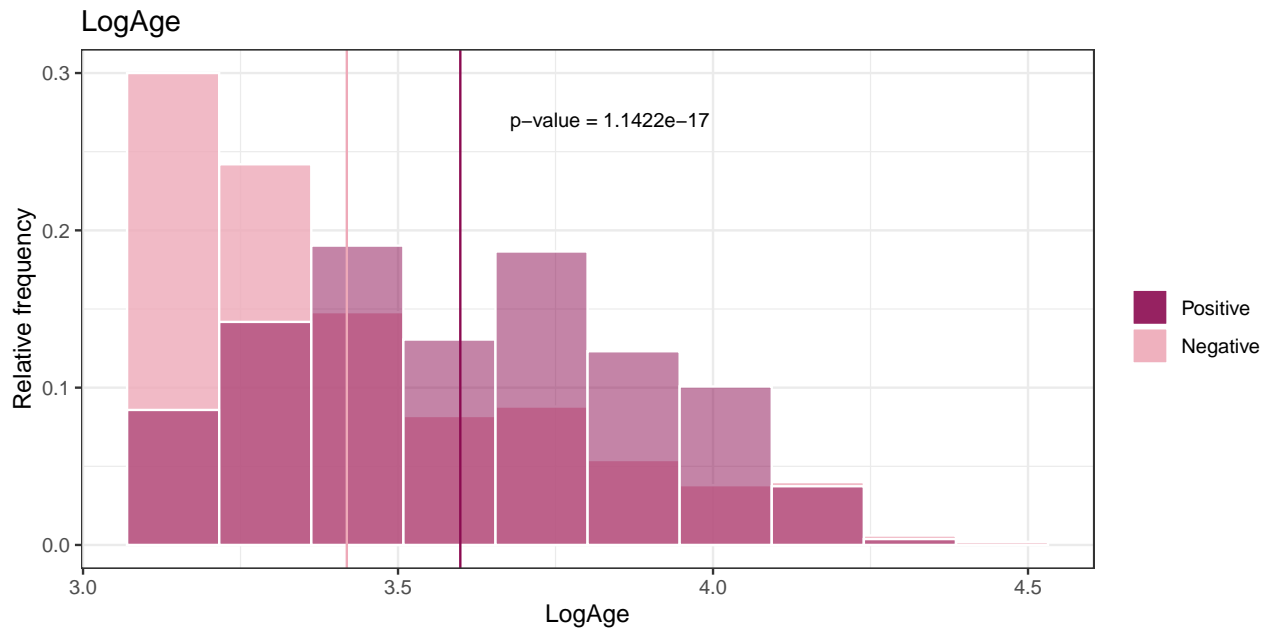
```
# Some convenience
df    <- subset(data, select = -c(Pregnancies, Insulin, BMI, DiabetesPedigreeFunction, Age))
df0       <-  df[df$Outcome == "Negative", ]
df1       <-  df[df$Outcome == "Positive", ]
xnames  <- names(df)[! names(df) %in% c("Outcome")]
```

Now, let's take a look at some multivariate plots. We'll begin by inspecting the Parallel Coordinate Plot:

```
par(las = 2)
parcoord(df[xnames], col = c("pink2", "deeppink4"))
legend("topright", legend = c("Negative", "Positive"),
       col = c("pink2", "deeppink4"), lty = 1, lwd = 2)
```
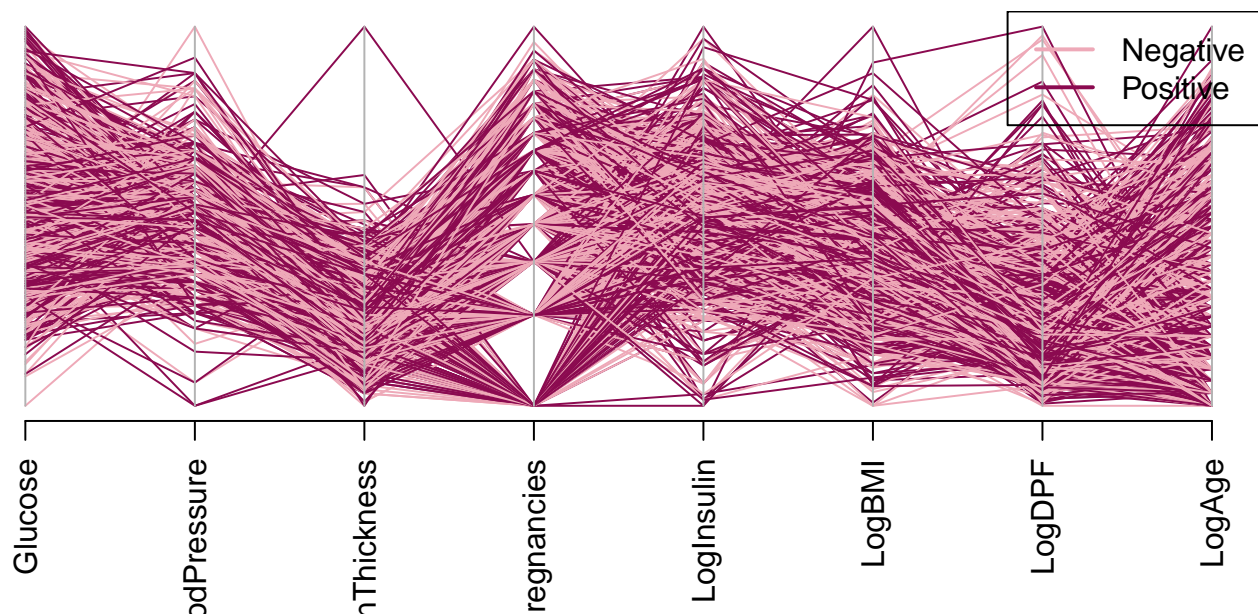

It seems that, overall, the positive lines are over negative ones. This is most notable on the Glucose and log transformed BMI. They are two of the most significant features according to the p-value.

The Andrew's plot is the following:

```
andrewsplot(as.matrix(df[xnames]), df$Outcome, style = "cart")
legend("topright", legend = c("Negative", "Positive"),
       col = c("black", "red"), lty = 1, lwd = 2)
```
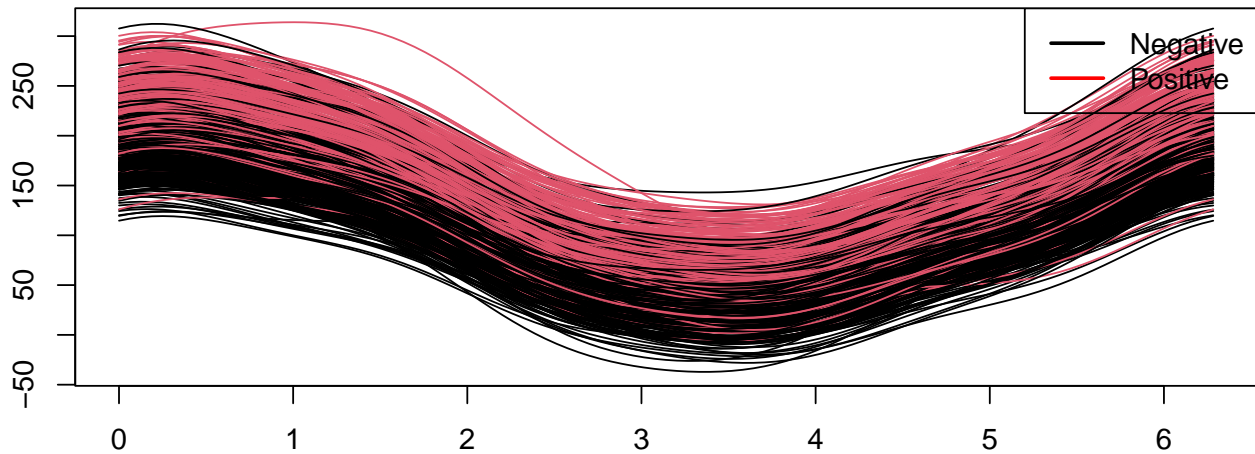
Andrews' Curves

Again, we see that the two groups are different. The group of people who have diabetes tend to have more volatile curves, reaching higher and lower values along the curve.

Multivariate characteristics and outlier identification

We are goig to use a multivariate approach to identifying the outliers in our data. In the process, we will need to compute the mean and covariance.

We start then by finding the mean vector and the covariance matrix. In order to reduce the sensitivity to outliers in this computation, we use a robust estimation using the "Fast MCD" (Minimum Covariance Determinant) estimator. We set main parameter, `alpha`, which determines the percentage of the data to use, at 0.85.

```
our_corrplot <- function(cov_mat) {
    colnames(cov_mat) <- c("Log\nPregnancies",
                           "Glucose",
                           "Blood\nPressure",
                           "Skin\nThickness",
                           "LogInsulin",
                           "LogBMI",
                           "LogDPF",
                           "LogAge")
    corrplot.mixed(cov2cor(cov_mat), lower = "number", upper = "color",
        diag = "n", tl.col = "black", tl.cex = 0.65,
        lower.col = "black")
}

mcd_est <- CovMcd(df[xnames], alpha = 0.85, nsamp = "deterministic")

## [1] "The mean vector is:"

##          Glucose  BloodPressure  SkinThickness  LogPregnancies      LogInsulin
##      120.4547983     71.9860918     28.3810848       1.3402737       4.7230964
```
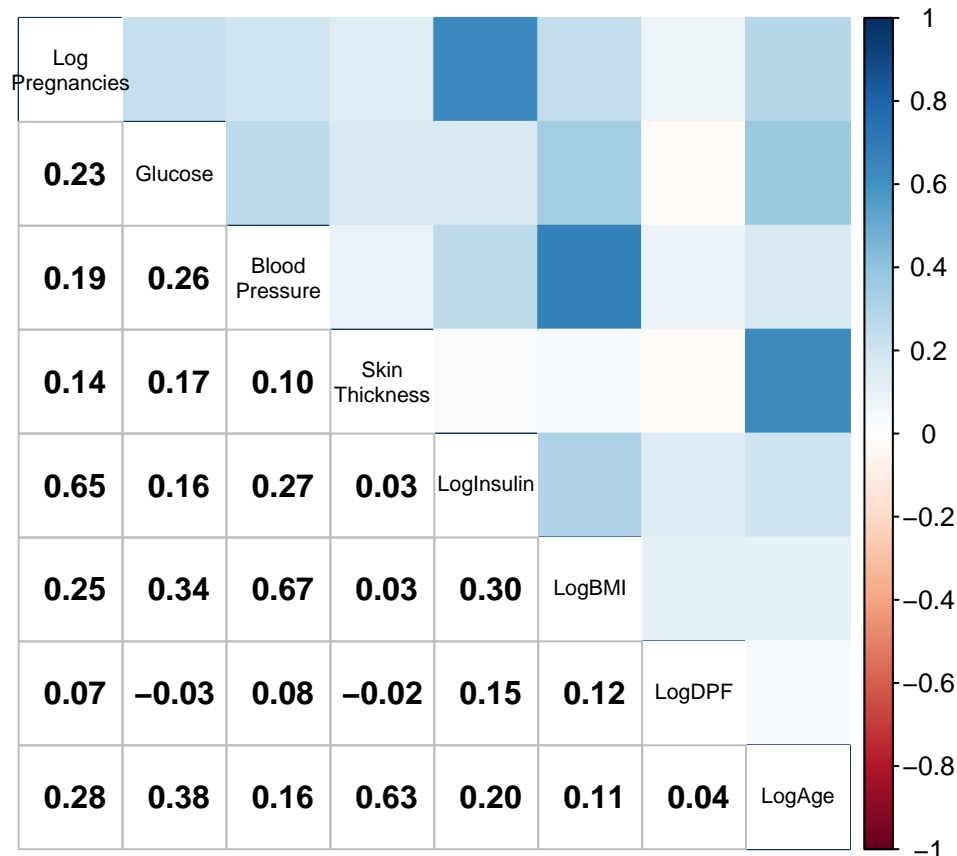
11

```
##          LogBMI          LogDPF          LogAge
##       3.4853699       0.3514095       3.4677968

## [1] "The covariance matrix is:"

##                  Glucose BloodPressure SkinThickness LogPregnancies LogInsulin
## Glucose         949.5595       85.7752       60.9537         3.2722    14.2651
## BloodPressure    85.7752      142.7851       32.1056         1.6069     1.3763
## SkinThickness    60.9537       32.1056      105.0266         0.7861     1.9485
## LogPregnancies    3.2722        1.6069        0.7861         0.6162     0.0143
## LogInsulin       14.2651        1.3763        1.9485         0.0143     0.5123
## LogBMI            1.5692        0.8434        1.4231         0.0055     0.0444
## LogDPF            0.4139       -0.0651        0.1557        -0.0035     0.0200
## LogAge            2.7784        1.4462        0.5345         0.1587     0.0461
##                LogBMI  LogDPF LogAge
## Glucose        1.5692  0.4139 2.7784
## BloodPressure  0.8434 -0.0651 1.4462
## SkinThickness  1.4231  0.1557 0.5345
## LogPregnancies 0.0055 -0.0035 0.1587
## LogInsulin     0.0444  0.0200 0.0461
## LogBMI         0.0427  0.0046 0.0072
## LogDPF         0.0046  0.0353 0.0022
## LogAge         0.0072  0.0022 0.1019
```

```
our_corrplot(mcd_est$cov)
```


It is interesting to seggregate by the class of the Outcome variable. This way, we can both, get the mean vector and covariance matrix for each of the classes, and the outliers for both sets.

```
mcd_neg <- CovMcd(df0[xnames], alpha = 0.85, nsamp = "deterministic")
mcd_pos <- CovMcd(df1[xnames], alpha = 0.85, nsamp = "deterministic")
```

[1] "### Negative class ###"

[1] "The mean vector is:"

```
##        Glucose  BloodPressure  SkinThickness LogPregnancies    LogInsulin
##    108.1916300     70.0242291     26.6365639      1.2186616     4.5363303
##         LogBMI          LogDPF         LogAge
##      3.4390348       0.3232806      3.3844990
```
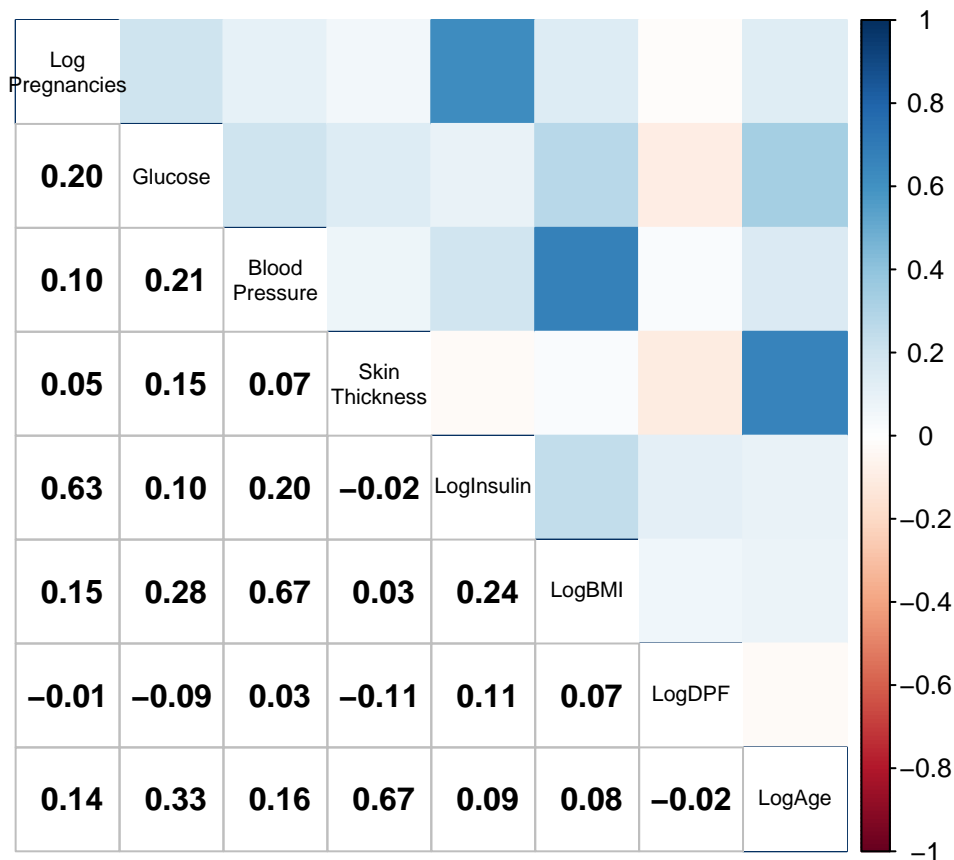
[1] "The covariance matrix is:"

```
##                  Glucose BloodPressure SkinThickness LogPregnancies LogInsulin
## Glucose          535.7768       53.3925       24.9768         0.9621    10.0637
## BloodPressure     53.3925      132.1672       25.3349         1.2742     0.7898
## SkinThickness     24.9768       25.3349      110.6200         0.5693     1.4467
## LogPregnancies     0.9621        1.2742        0.5693         0.5732    -0.0112
## LogInsulin        10.0637        0.7898        1.4467        -0.0112     0.4821
## LogBMI             0.7120        0.6712        1.4900         0.0041     0.0355
## LogDPF            -0.0496       -0.1801        0.0490        -0.0139     0.0135
## LogAge             0.9492        1.1183        0.4847         0.1485     0.0188
##                  LogBMI   LogDPF   LogAge
## Glucose          0.7120  -0.0496   0.9492
## BloodPressure    0.6712  -0.1801   1.1183
## SkinThickness    1.4900   0.0490   0.4847
## LogPregnancies   0.0041  -0.0139   0.1485
## LogInsulin       0.0355   0.0135   0.0188
## LogBMI           0.0447   0.0025   0.0053
## LogDPF           0.0025   0.0294  -0.0013
## LogAge           0.0053  -0.0013   0.0866
```

```
our_corrplot(mcd_neg$cov)
```

Correlation matrix (lower triangle):

	LogPregnancies	Glucose	BloodPressure	SkinThickness	LogInsulin	LogBMI	LogDPF	LogAge
Glucose	0.20							
BloodPressure	0.10	0.21						
SkinThickness	0.05	0.15	0.07					
LogInsulin	0.63	0.10	0.20	−0.02				
LogBMI	0.15	0.28	0.67	0.03	0.24			
LogDPF	−0.01	−0.09	0.03	−0.11	0.11	0.07		
LogAge	0.14	0.33	0.16	0.67	0.09	0.08	−0.02	

```
## [1] "### Positive class ###"

## [1] "The mean vector is:"

##          Glucose   BloodPressure  SkinThickness LogPregnancies     LogInsulin
##      142.2148438      75.5351562     31.7304688      1.5230344      5.0556806
##           LogBMI          LogDPF         LogAge
##        3.5763371       0.4027727      3.5985651

## [1] "The covariance matrix is:"

##                 Glucose BloodPressure SkinThickness LogPregnancies LogInsulin
## Glucose         931.9767       34.5571       21.1150        -1.0132    11.1108
## BloodPressure    34.5571      152.7967       19.9239         0.9369     0.7668
## SkinThickness    21.1150       19.9239       86.7544        -0.2417     1.2014
## LogPregnancies   -1.0132        0.9369       -0.2417         0.6889    -0.0464
## LogInsulin       11.1108        0.7668        1.2014        -0.0464     0.4275
## LogBMI            0.2897        0.7166        0.9532        -0.0222     0.0159
## LogDPF           -0.0499       -0.1220        0.0606        -0.0064     0.0091
## LogAge            0.7751        1.1984       -0.2209         0.1289     0.0199
##                 LogBMI  LogDPF   LogAge
## Glucose         0.2897 -0.0499   0.7751
## BloodPressure   0.7166 -0.1220   1.1984
## SkinThickness   0.9532  0.0606  -0.2209
## LogPregnancies -0.0222 -0.0064   0.1289
## LogInsulin      0.0159  0.0091   0.0199
## LogBMI          0.0294  0.0026  -0.0075
## LogDPF          0.0026  0.0430  -0.0047
```
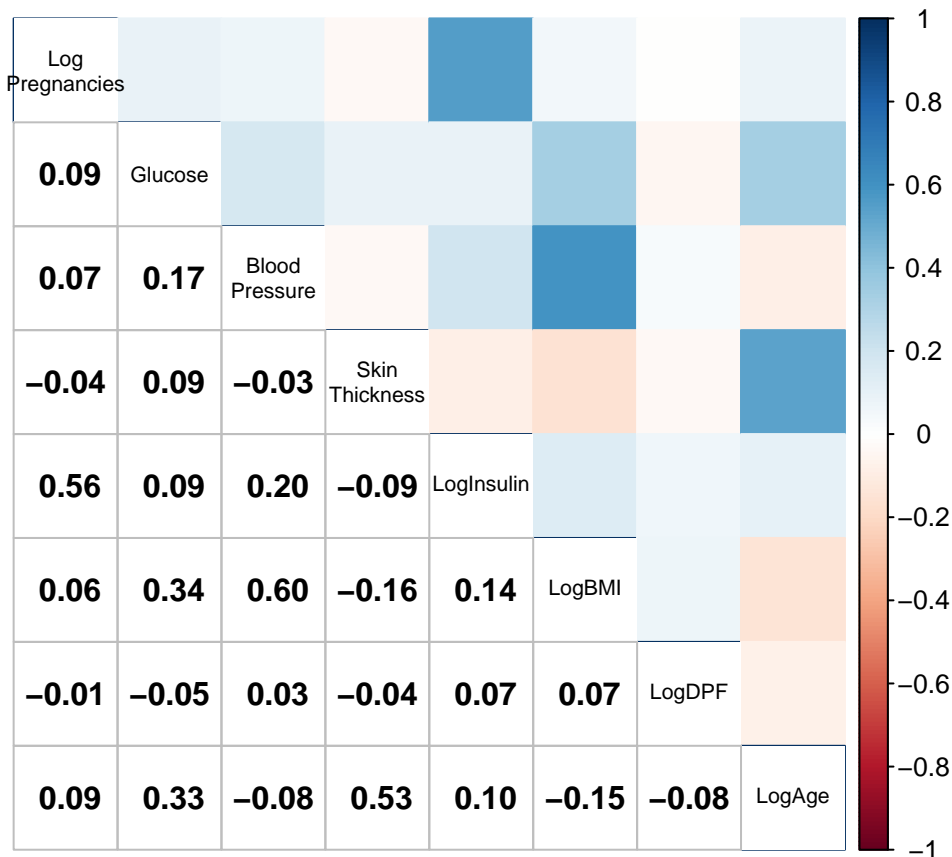
```
## LogAge          -0.0075 -0.0047  0.0854
```
```
our_corrplot(mcd_pos$cov)
```


Let us focus first on the mean vectors. The here observed particularities are nothing new, as they were already present on above histograms:

- Higher number of pregnancies seems to increase the chances on developing diabetes.

- The glucose and insulin levels of the positive population is higher in contrast to the negative one.

Looking now at the representations for the covariance matrices, the major changes are that the correlation between skin thickness and diabetes pedigree function is lower in the group who do not have diabetes than in the one having the disease. Moreover, the correlation between BMI [3] and age is positive for the sane group while negative on the positive one.

We now search for outliers. The idea is to use Mahalanobis distance. As we suppose that our data $X \sim \mathcal{N}(\mu, \Sigma)$, we have $D_M(x,\mu)^2 \sim \chi_p^2$, with D_M the Mahalanobis distance. Hence we may set our outlier criteria as

$$D_M(x,\mu)^2 > \chi_{p,0.95^{1/n}}^2 \ , \tag{1}$$

the 0.95^{th} quantile of the χ_p^2 distribution. We then drop these outliers.

```
p    <- length(xnames)
n0   <- nrow(df0)
n1   <- nrow(df1)
df0_clean <- df0[mcd_neg$mah < qchisq(0.95^(1 / n0), p), ]
df1_clean <- df1[mcd_pos$mah < qchisq(0.95^(1 / n1), p), ]
df_clean  <- rbind(df0_clean, df1_clean)
```

---

[3]Remember this is the logarithm.

```
## [1] "The negative set contained 6 outliers."
```

```
## [1] "The positive set contained 4 outliers."
```

As a final summary of this section, we perform a plot in which the histograms of different populations are observed, as well as scatterplots of pairs of variables and correlations.

```
ggpairs(df_clean, aes(color = Outcome), legend = 1,
        columns = xnames,
        diag = list(continuous = "barDiag")
        ) +
    theme(legend.position = "bottom") +
    scale_fill_manual(values = c("pink", "deeppink4")) +
    scale_color_manual(values = c("pink", "deeppink4")) + labs(fill = "Outcome")
```