Clusterization and classification of hepatitis dataset

Monika Zakrzewska

Contents

1 Descriptive analysis and visualization

This report will describe a *hepatitis* dataset downloaded from the UCI Machine Learning open repository. The dataset describes patients who have undergone hepatitis. Initially, we will consider the association of variables due to survival. There are 14 categorical variables in the dataset (Class, Sex, Steroid, Antivirals, Fatigue, Malaise, Anorexia, Liver Big, Liver Firm, Spleen Palpable, Spiders, Ascites, Varices) and 6 quantitative variables (Age, Bilirubin, Alk Phosphate, SGOT, Albumin, Protime). The data (especially the quantitative variables) have missing records. To fill in the missing records in quantitative variables, the median will be used and in the categorical variables it will be the mode (the most commonly used data).

In the database, the empty records had a value of *"?"* which was converted to *NA* while loading the data into RStudio.

The question we want to answer in this analysis is whether age, gender, psychology (fatigue, malaise, anorexia) and the amount of bilirubin (a bile pigment that comes from the breakdown of red blood cells, its concentration in the blood helps to assess liver function - an increase in this concentration can cause jaundice) affected survival. This subset will be called *hepatitis1* and will be used in the rest of the document - for clusterization and classification.

Below we chnage the names of columns and recode categorical variables.

```
colnames(hepatitis) <- c("Class", "AGE", "SEX", "STEROID", "ANTIVIRALS", "FATIGUE",
                         "MALAISE", "ANOREXIA", "LIVER BIG", "LIVER FIRM",
                         "SPLEEN PALPABLE", "SPIDERS", "ASCITES", "VARICES",
                         "BILIRUBIN", "ALK_PHOSPHATE", "SGOT", "ALBUMIN",
                         "PROTIME", "HISTOLOGY")

hepatitis$SEX[hepatitis$SEX == "1"] <- "M"
hepatitis$SEX[hepatitis$SEX == "2"] <- "F"

hepatitis$Class[hepatitis$Class == "1"] <- "DIE"
hepatitis$Class[hepatitis$Class == "2"] <- "LIVE"

hepatitis[4:14][hepatitis[,c(4:14)] == 1] <- 0
hepatitis[4:14][hepatitis[,c(4:14)] == 2] <- 1
hepatitis[20][hepatitis[20] == 1] <- 0
hepatitis[20][hepatitis[20] == 2] <- 1

hepatitis1 <- data.frame(hepatitis[1:3], hepatitis[6:8], hepatitis[15])
```

The gender variables from *1* and *2* were changed to *M* and *F* respectively, in the Class variable *1* means *DIE* and *2* means *LIVE*. The qualitative data were then changed from *1* to *0* and from *2* to *1* in all data.

Initially, all the data will be analysed, but the results will only be presented for selected ones involved in the analysis (Class, age, fatigue, malaise, anorexia and bilirubin).

A general description will be given below for orientation, i.e. how many variables we have, how many complete rows, etc.

```
##                       [,1]
## rows                   155
## columns                  7
## discrete_columns         2
## continuous_columns       5
## all_missing_columns      0
## total_missing_values     9
## complete_rows          149
## total_observations    1085
```

```
## memory_usage          11216
```

It is seen that there are 155 observations (patients) with only 9 missing cells (out of 1085), so it is not a big deficiency.

Below there is a chart that shows how the missing data is distributed across the group.

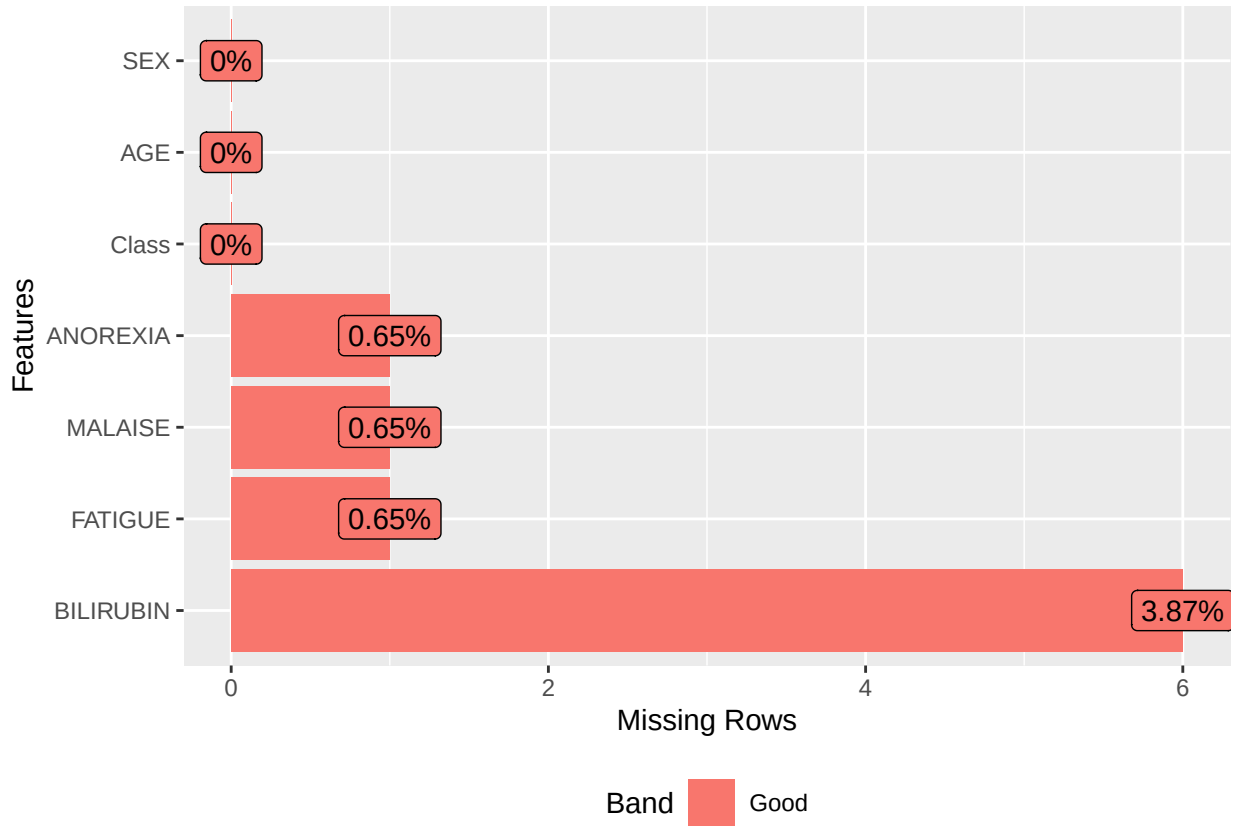


Figure 1: Missing data distribution

As it is seen, there is not much missing data, but for the accuracy of the analysis and the lack of deletion of records, they will be supplemented with an median or mode.

1.1 Quantitative variables

As can be seen above Bilirubin is the only quantitative variable to have missing data. Therefore, these missing data will be replaced by median.

```
hepatitis1 <- hepatitis1 %>%
  replace_na(list(BILIRUBIN = median(na.omit(hepatitis1$BILIRUBIN))))
```

Below are descriptive statistics and measures of position and spread for the quantitative variables considered.

Table 1: Descriptive statistics for quantitative variables

	AGE	BILIRUBIN
Mean	41.20	1.41
Std.Dev	12.57	1.19
Min	7.00	0.30
Q1	32.00	0.80
Median	39.00	1.00
Q3	50.00	1.50
Max	78.00	8.00
MAD	13.34	0.44
IQR	18.00	0.70
CV	0.30	0.84
Skewness	0.36	2.91
SE.Skewness	0.19	0.19
Kurtosis	-0.18	10.24
N.Valid	155.00	155.00
Pct.Valid	100.00	100.00

It can be seen that the average age of the individuals in the analysis under consideration a little bit over 41 years, so they are middle-aged. It can also be seen that there may be outliers in the case of bilirubin, which will be diagnosed and addressed in the following subsections.

Table 2: Descriptive statistics for patients that survived and for patients that died

	Survived		Died	
	AGE	BILIRUBIN	AGE	BILIRUBIN
Mean	39.80	1.14	46.59	2.45
Std.Dev	12.83	0.71	9.94	1.91
Min	7.00	0.30	30.00	0.40
Q1	30.00	0.70	38.50	1.00
Median	38.00	1.00	46.50	1.80
Q3	50.00	1.20	55.00	3.40
Max	78.00	4.60	70.00	8.00
MAD	11.86	0.44	11.86	1.26
IQR	20.00	0.50	15.75	2.15
CV	0.32	0.62	0.21	0.78
Skewness	0.51	2.79	0.29	1.35
SE.Skewness	0.22	0.22	0.41	0.41
Kurtosis	-0.02	9.00	-0.72	1.27

N.Valid	123.00	123.00	32.00	32.00
Pct.Valid	100.00	100.00	100.00	100.00

The mean age for the survivor class is lower than for the deceased class, which may suggest that more people who were older died. The median age for the survivors is more than 8 years less than the median for those who died. Median bilirubin levels were more than 2 times higher for patients who died.

- Box plots for the data considered by class:

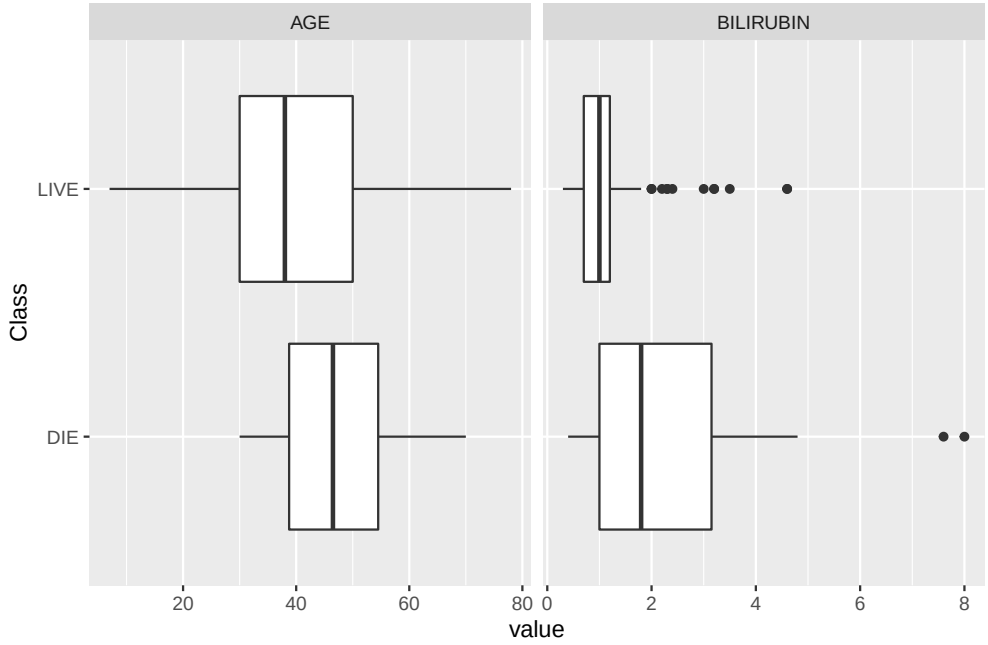


Figure 2: Box plots for the data considered by class

In both cases, bilirubin has outlier observations. Class "LIVE" have more of them. To replace those outliers, median value will be assigned in a place of the outlier.

```
sapply(hepatitis1$BILIRUBIN, function(x) x %in% boxplot(hepatitis1$BILIRUBIN, plot = FALSE)$out)

hepatitis1$BILIRUBIN[hepatitis1$BILIRUBIN %in% boxplot(hepatitis1$BILIRUBIN, plot = FALSE)$out] <- media
```

- Distributions of quantitative variables:

The distribution graph for bilirubin looks like this due to the fact that there were quite a few outlier observations that were converted into a median. The distribution for age gently resembles a normal distribution, whereas the distribution for bilirubin does not.

- Densities:

The density of the AGE variable also resembles a normal distribution (as does the histogram), so one can check that the data come from a normal distribution. This will also be checked for bilirubin (but it is already clear from the histogram and density plot that the sample is not from a normal distribution).

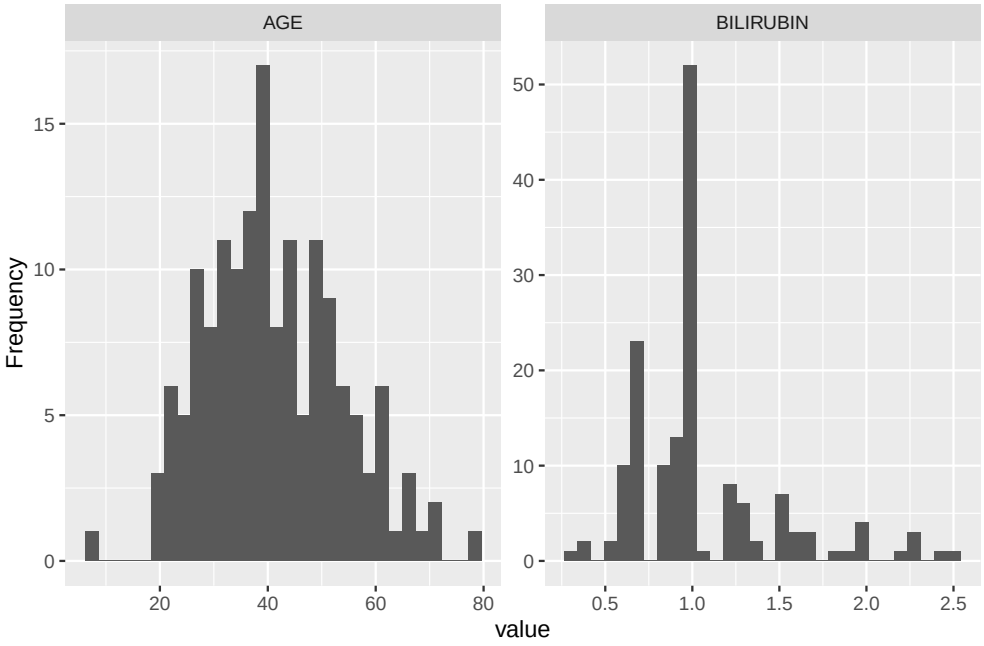


Figure 3: Distributions of quantitative variables

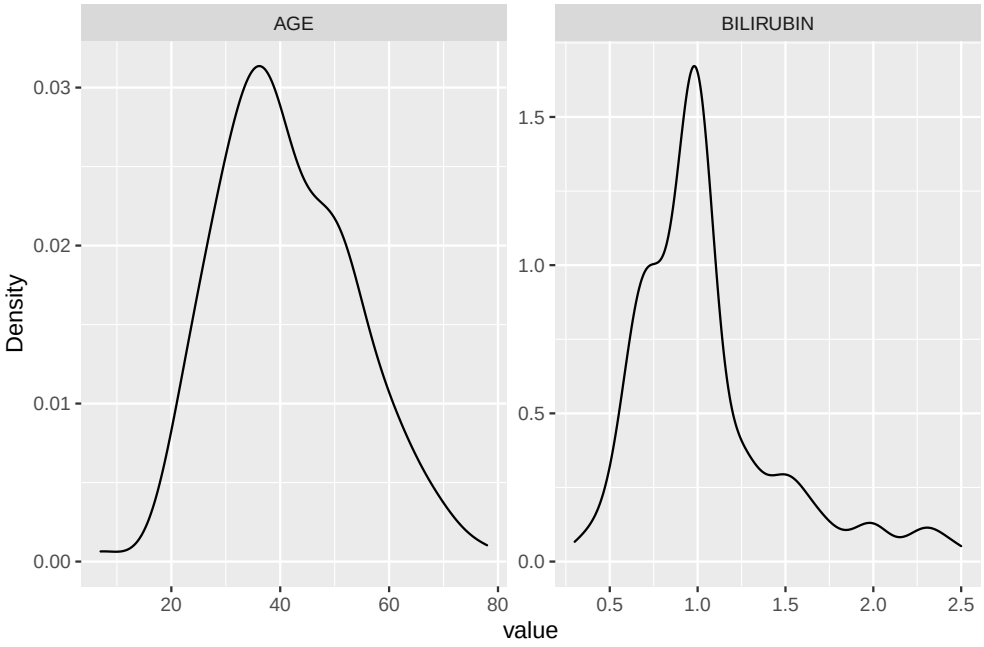


Figure 4: Density plots of quantitative variables

Table 3: Shapiro-Wilk test results for quantitative variables

	W.statistics	p.value
AGE	0.98	0.08
BILIRUBIN	0.86	0.00

The p-value for age is greater than the standard significance level ($\alpha = 0.05$), so there is no basis to reject the null hypothesis that the sample comes from a normal distribution. For bilirubin the p-value is less than the standard significance level, so we can reject the null hypothesis in favour of the alternative hypothesis, that the sample is not from a normal distribution.

- QQ-plot:

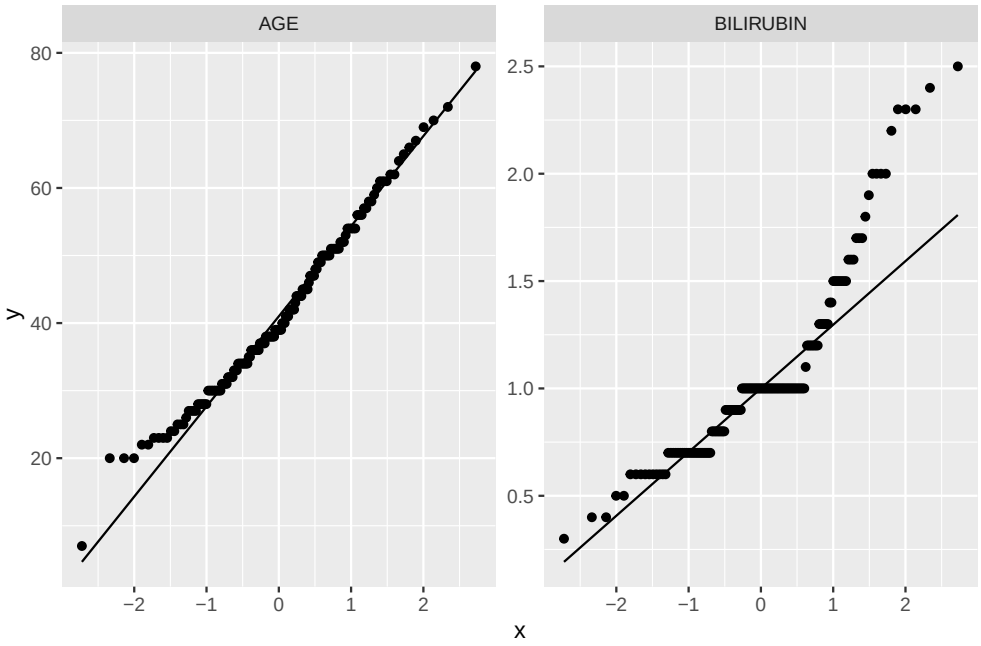


Figure 5: QQ plots of quantitative variables

In the qq plots, it is clear that the ages arrange themselves into a straight line, which may suggest that the sample comes from a normal distribution. The bilirubin values do not arrange themselves into a straight line, indeed they have 'levels' that may have arisen when the outliers were turned into the median.

- Correlations:

From the graph, it can be seen that a medium ($\rho \in (0.4, 0.7)$) correlation occurs between malaise and fatigue and between malaise and anorexia. Strong and very strong correlations do not occur between these data. The rest of the variables have weak or very weak correlations between them.

- Data spread:

A scatterplot shows that there is no apparent trend or correlation between age and bilirubin levels.

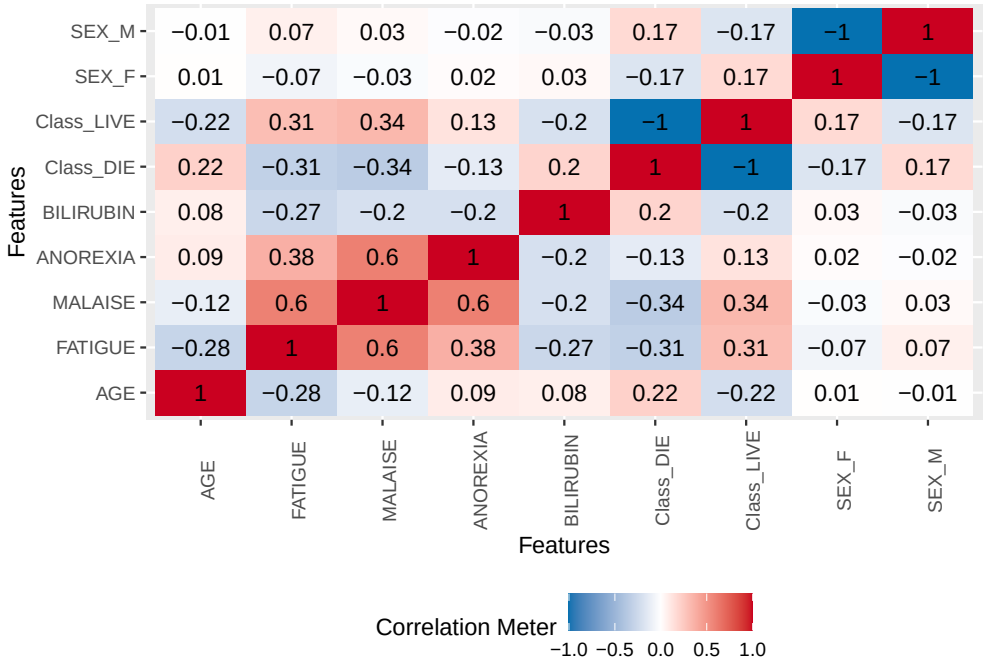


Figure 6: Correlations plot

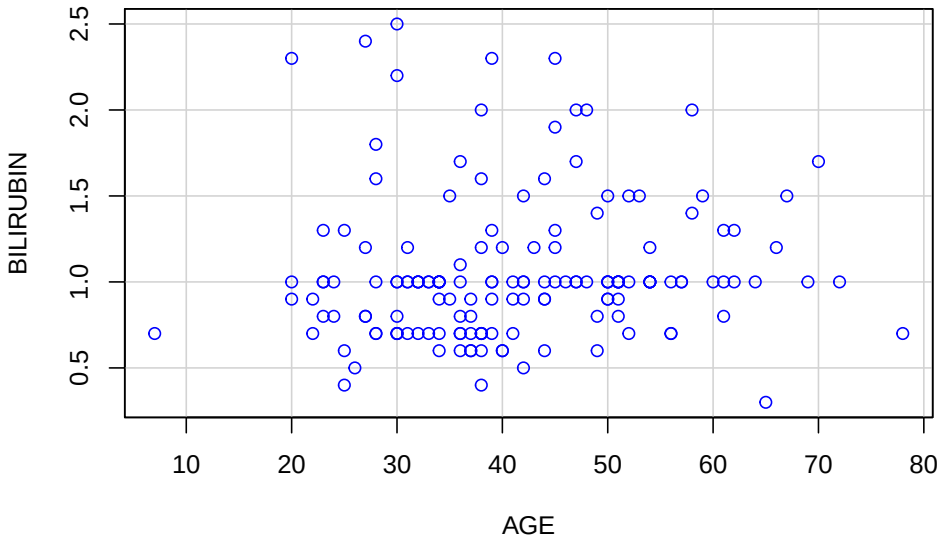


Figure 7: Scatterplot of AGE and BILIRUBIN

1.2 Qualitative variables

To replace missing data, mode will be used.

```
mode <- function(x){
  distinct_values <- unique(x)
  distinct_tabulate <- tabulate(match(x, distinct_values))
  distinct_values[which.max(distinct_tabulate)]
}

hepatitis1 <- hepatitis1 %>%
  mutate(across(everything(), ~replace_na(.x, mode(.x))))
```

Below there are frequency tables for the variables considered as categorical.

Table 4: Frequency table of categorical variables by class

label	variable	Class	
		DIE	**LIVE**
SEX	F	0 (0%)	16 (100.00%)
	M	32 (23.02%)	107 (76.98%)
FATIGUE	0	30 (29.70%)	71 (70.30%)
	1	2 (3.70%)	52 (96.30%)
MALAISE	0	23 (37.70%)	38 (62.30%)
	1	9 (9.57%)	85 (90.43%)
ANOREXIA	0	10 (31.25%)	22 (68.75%)
	1	22 (17.89%)	101 (82.11%)

From the table itself, it can be seen that gender is unlikely to influence the class due to the fact that among women, 0% have died, and on top of that, in this group the subgroups differ strongly from each other (they are not equal).

Below there is a graph of what survival looks like depending on whether there was malaise. Nearly 40% of people who did not have malaise died, and of those patients who had malaise, more than 90% survived, which may suggest that people with good wellbeing were more likely to die than those with malaise.

Now, when the data exploration is over, let's get to the clusterization.

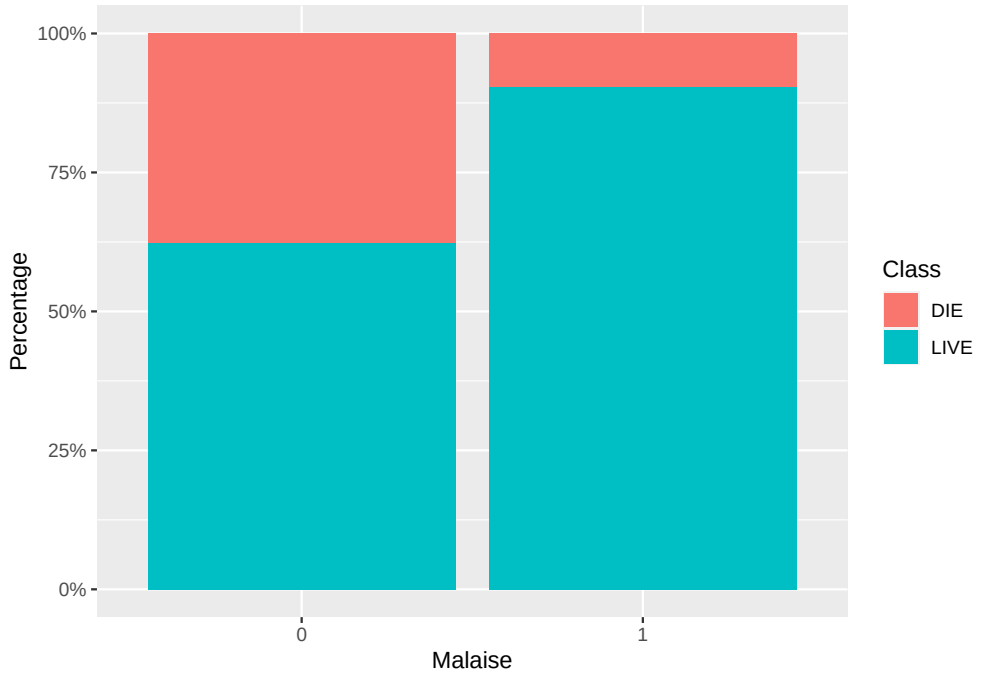


Figure 8: Barplot of Malaise by group and class

2 Clusterization

2.1 Methods

The clustering methods/algorithms that will be used are:

- **PAM** - a generalised version of the k-means algorithm. It can be used for mixed type data (qualitative and quantitative variables) and is more robust to outlier observations than the k-means metdoa. As in the k-means method, the number of clusters must be specified at the beginning.
- **AGNES** - belongs to agglomerative methods, which means that each observation is initially treated as a separate cluster. In subsequent stages, groups that are similar to each other are combined into larger and larger groups until an all-encompassing cluster is formed.
- **DIANA** - initially we have one cluster, then split into two, etc. until each element is in a separate leaf (i.e. the reverse method to AGNES).
- **K-prototypes** - it is a mix of a popular methods k-means and k-modes. It is userd for mixed type of data. It uses the Euclidian distance for numeric variables and simple matching coefficient for categorical variables in the dataset.

2.2 How to find the optimal number of clusters?

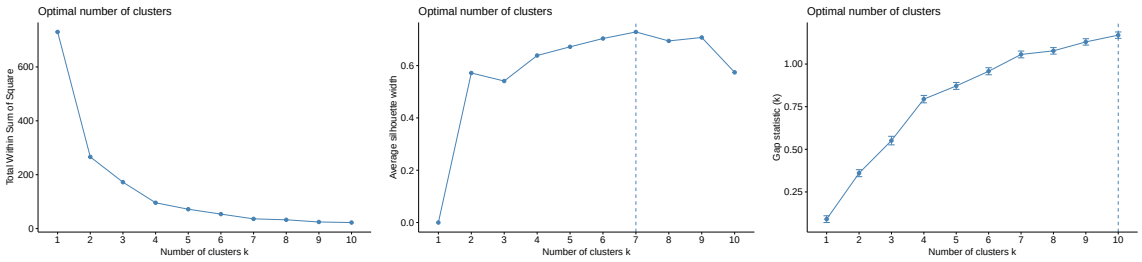
1. Elbow method,
2. Mean values of Silhouette index (the bigger value the better),
3. Gap value,
4. McClain index (the lower value the better, only for k-prototypes model).

2.3 PAM

First method will be PAM (Partition Around Medoids). We will find dissimmilarity matrix with Gower distance but without class labels.

```
hepatitis.no.labels <- as.data.frame(hepatitis1[,2:7])
dist3 <- daisy(hepatitis.no.labels,
                    metric = "gower")
dist_mat <- as.matrix(dist3)
```

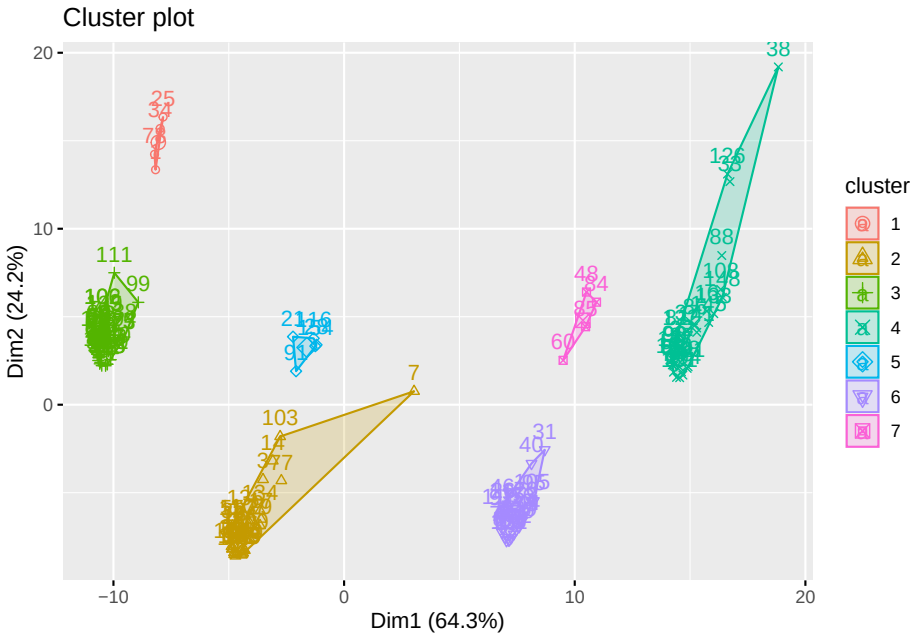
Below there are plots with within sum of squares, silhouette and gap for each number of clusters. It can help with deciding how many clusters should be in the model.



Here, the best k would be probably $k = 7$.

```
pam_fit <- pam(dist3, k=7, diss = TRUE)
```

And below is a diagram of what the graph looks like with a 7 cluster split.



Here the clusters clearly doesn't overlap and are well separated. It is also possible to check Calinski-Harabasz index (the bigger the better).

```
calinhara(dist3, pam_fit$cluster)
```

```
## [1] 399.711
```

2.4 AGNES

Here, there is no need to specify k value before start of clusterization, but there have to be specified the linkage method (single, complete, average, Ward). To select the best metod, one need to look at the agglomeration coefficient, which measures the amount of cluster structure found (values closer to 1 suggest stronger cluster structures).

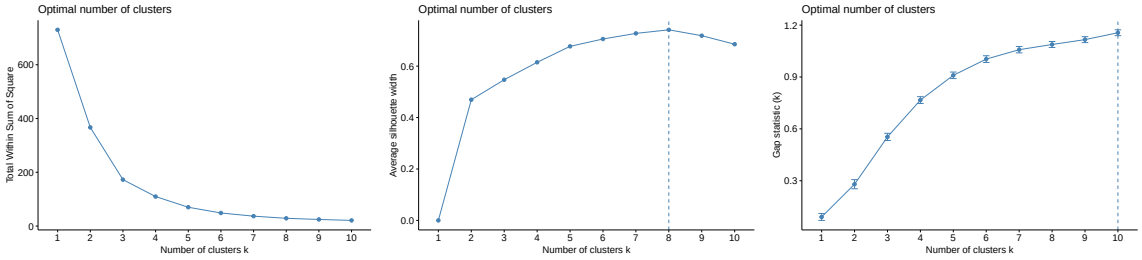
```
m <- c( "single", "average", "complete", "ward")
names(m) <- c( "single", "average", "complete", "ward")
# function to compute aglomeration coefficient
ac <- function(x) {
  agnes(hepatitis.no.labels, method = x, metric = "euclid")$ac
}
map_dbl(m, ac)
```

```
##    single   average  complete      ward
## 0.9341169 0.9665915 0.9854284 0.9942396
```

From the agglomeration coefficient, it is seen that the best is Ward *linkage method*.

```
agnes.ward <- agnes(hepatitis.no.labels, method = "ward", metric="euclid")
```

The optimal number of clusters can be checked using three methods. From the elbow method (top graph), the optimal number of clusters is 5. From the graph of the average silhouette index value (graph in the middle), it can be read that the best number of clusters is 8, and for the gap statistics method (bottom graph) it is 10. So the number taken here will be 8.



Below is a dendogram and clusters plot.

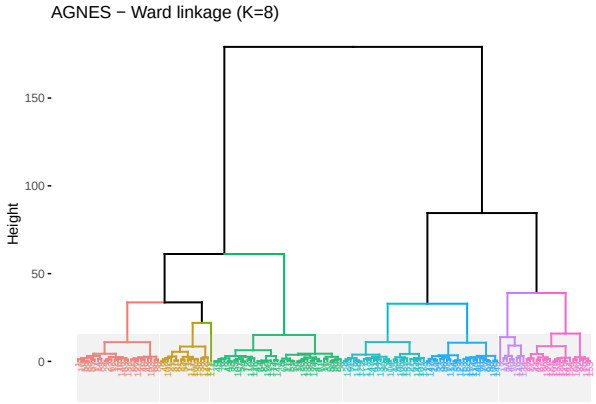


Figure 9: Dendogram and cluster plot for AGNES

Here the cluster plot shows that the clusters are overlapping and clusterization is not too good.

Below there is Calinski-Harabasz index.

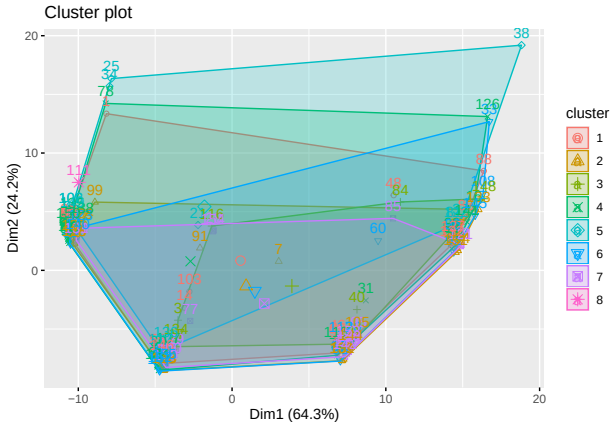


Figure 10: Dendogram and cluster plot for AGNES

```
calinhara(dist3, clust.agnes)
```

```
## [1] 1.813996
```

2.5 DIANA

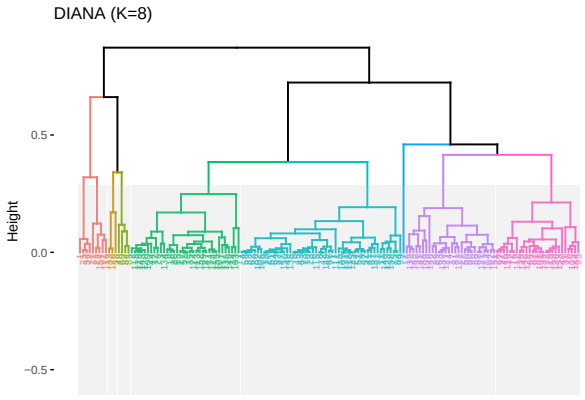
Here, dissimilarity matrix needs to be used - similar to the PAM algorithm.

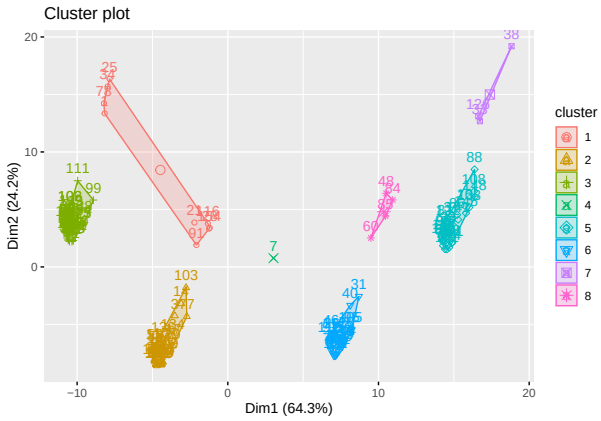
```
diana.mod <- diana(dist3)
```

The divisive coefficient is

```
## [1] 0.9728111
```

The divisive coefficient is quite high. Below there is the dendogram with the division into eight clusters (it is possible to take the same number of clusters as in AGNES).





The clusters are very separated from each other.

Below there is Calinski-Harabasz index.

```
calinhara(dist3, clust.diana.mod)
```

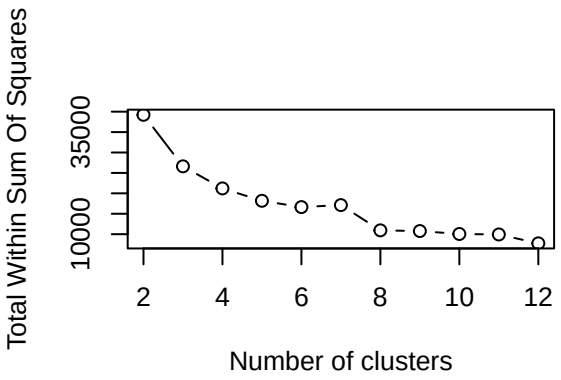
```
## [1] 393.6119
```

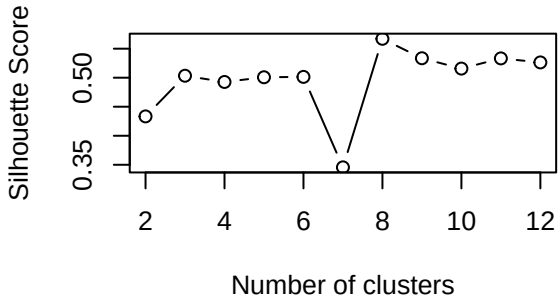
2.6 K-prototypes

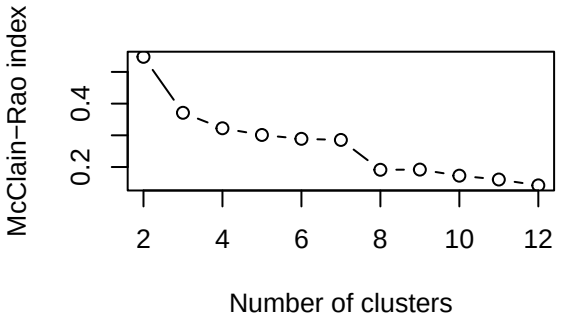
Here, data taken will not be "processed", but taken without the labels. To find the best value of k, Silhouette index, elbow method and McClain index will be shown below in the plots.

```
k_proto_no_clus <- function(df){

  ind <- numeric(12)
  ind_s <- numeric(12)
  ind_m <- numeric(12)

  for (i in 1:12){
    k_pro <- kproto(df, k = i)

    if (i == 1){
      ind[i] <- 0
      ind_s[i] <- 0
      ind_m[i] <- 0
    }

    else{
      ind[i] <- k_pro$tot.withinss
      ind_s[i] <- validation_kproto(method = "silhouette", object = k_pro)
      ind_m[i] <- validation_kproto(method = "mcclain", object = k_pro)
    }
  }

  par(mfrow=c(2,2))

  plot(2:12, ind[2:12], type = "b", ylab = "Total Within Sum Of Squares", xlab = "Number of clusters")
  plot(2:12, ind_s[2:12], type = "b", ylab = "Silhouette Score", xlab = "Number of clusters")
  plot(2:12, ind_m[2:12], type = "b", ylab = "McClain-Rao index", xlab = "Number of clusters")

}
```

```
k_proto_no_clus(hepatitis.no.labels)
```







From the elbow method and silhouette score, the best number of cluster should be 8.

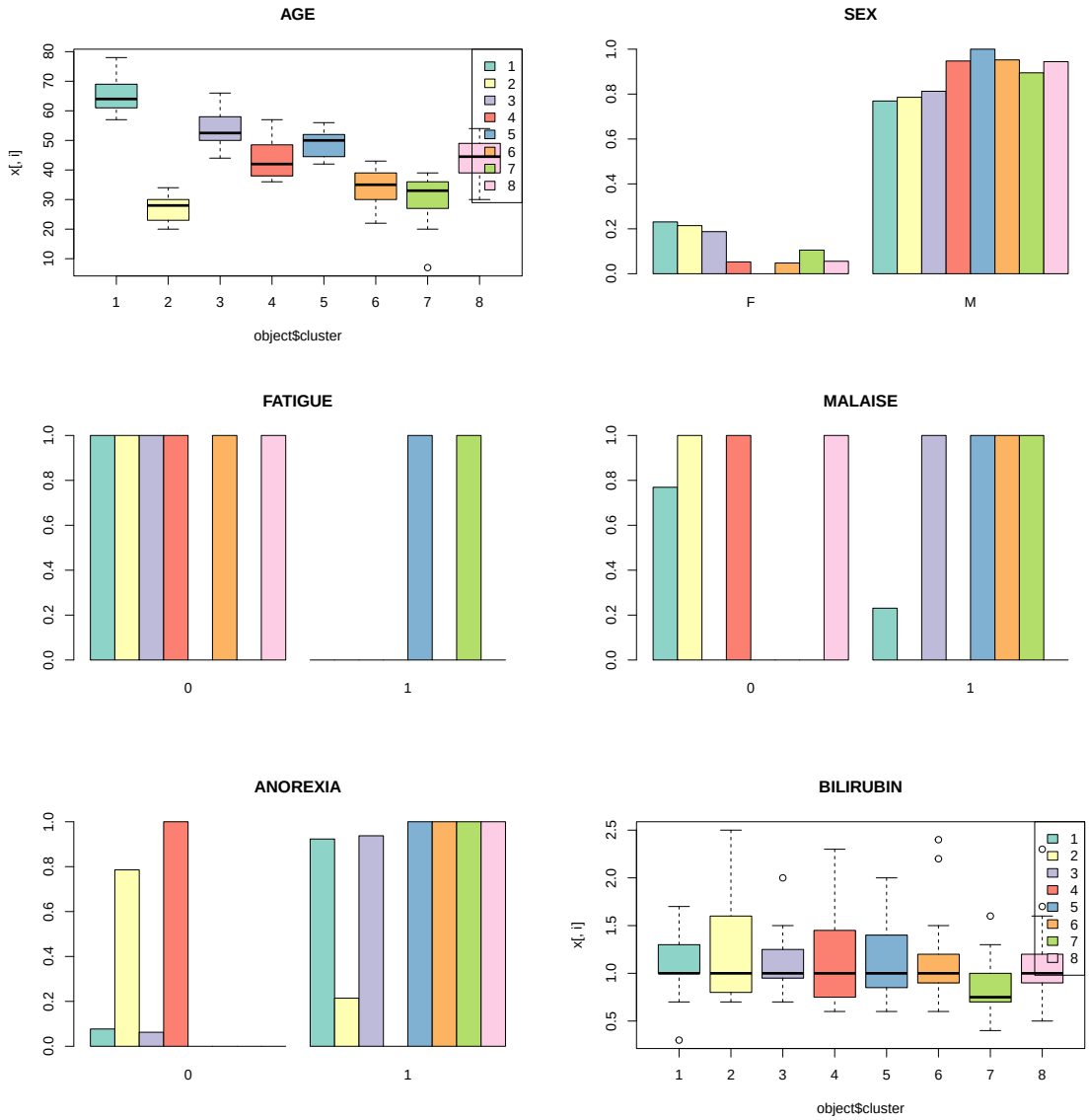
```
k_proto <- kproto(hepatitis.no.labels, k = 8)
```

Below there are centers of all clusters for every variable.

```
k_proto$centers
```

```
##        AGE SEX FATIGUE MALAISE ANOREXIA BILIRUBIN
## 1 65.15385   M       0       0        1 1.0615385
## 2 27.07143   M       0       0        0 1.2642857
## 3 53.68750   M       0       1        1 1.1250000
## 4 43.78947   M       0       0        0 1.1578947
## 5 48.62500   M       1       1        1 1.0937500
## 6 34.09524   M       0       1        1 1.1047619
## 7 31.28947   M       1       1        1 0.8210526
## 8 43.66667   M       0       0        1 1.1055556
```

Below it is shown in a plots how the clusters look for every variable.



This method has the lowest value of Silhouette score from all the methods presented here.

Below there is Calinski-Harabasz index.

```
calinhara(dist3, k_proto$cluster)
```

```
## [1] 120.1169
```

By looking only at the Calinski-Harabasz index, the best algorithm for this data is PAM.

3 Classification

In this part of the report, the objective is to find a good classifier that classifies patients from our subset into two groups - survival (*LIVE*) and death (*DIE*).

3.1 Methods

1. Logistic Regression (LR) - an algorithm otherwise known as a generalised linear model, used to predict the probability of belonging to a class. Logistic regression allows direct assessment of the significance of variables.
2. AdaBoost - this algorithm combine multiple "weak classifiers" into a single "strong classifier".
3. Classification trees - an algorithm that represents choices and their results in the form of a tree. The nodes in the graph represent the event or choice, and the edges of the graph represent the rules or decision conditions.

3.2 Comparison of the classification error using train and test split

In this section of the report, different classification methods will be compared. 3 methods have been used, of which only the most accurate one (i.e. the one for which the classification error was the smallest) will be described. The data has been divided into test and training data, which will be used for all classification models.

```
n <- nrow(hepatitis1)
learning.indx <- sample(1:n,0.7*n)

learning.set <- hepatitis1[learning.indx,]
test.set <- hepatitis1[-learning.indx,]
```

First, models need to be created and fitted to the data.

```
# Classification tree

model <- Class~. # all of the variables for creation of the classifiers
n.test <- dim(test.set)[1]

# We build the decision tree (with default parameters)
tree.default <- rpart(model, data=learning.set)

pred.labels.test <- predict(tree.default, newdata=test.set, type = "class")

# determination of projected a posteriori probabilities
pred.probs.test <- predict(tree.default, newdata=test.set, type = "prob")

# Evaluation of classification accuracy

# confusion matrix
conf.mat.test <- table(pred.labels.test, test.set$Class)
error.rate.test <- (n.test - sum(diag(conf.mat.test))) / n.test

sens.tree <- conf.mat.test[2,2]/(conf.mat.test[2,2] + conf.mat.test[2,1])
spec.tree <- conf.mat.test[1,1]/(conf.mat.test[1,1] + conf.mat.test[1,2])
```

```
# Logistic regression

hepatitis1$Class <- factor(hepatitis1$Class)
```

```
learning.set$Class <- factor(learning.set$Class)
test.set$Class <- factor(test.set$Class)

model.logit <- glm(Class~., data = learning.set, family = binomial(link="logit"))

pred.prob <- predict(model.logit, test.set, type = "response")

prob2labels <- function(probs,cutoff)
{
  klasy <- rep("DIE",length(probs))
  klasy[probs>cutoff] <- "LIVE"
  return(as.factor(klasy))
}

# actual and predicted labels
lr.labels <- prob2labels(probs=pred.prob,cutoff=0.35)
real.labels <- test.set$Class

# classification error
conf.matrix <- table(lr.labels, real.labels)
classif.error <- (n.test - sum(diag(conf.matrix)))/n.test

sens.lr <- conf.matrix[2,2]/(conf.matrix[2,2] + conf.matrix[2,1])
spec.lr <- conf.matrix[1,1]/(conf.matrix[1,1] + conf.matrix[1,2])
```

```
# AdaBoost

model.adaboost <- boosting(Class~., data = learning.set, boos = TRUE, mfinal = 50)

pred.adaboost <- predict.boosting(model.adaboost,newdata = test.set)

# Evaluation of classification accuracy

# confusion matrix
conf.mat.test.ada <- pred.adaboost$confusion
error.rate.test.ada <- pred.adaboost$error

sens.ada <- conf.mat.test.ada[2,2]/(conf.mat.test.ada[2,2] + conf.mat.test.ada[2,1])
spec.ada <- conf.mat.test.ada[1,1]/(conf.mat.test.ada[1,1] + conf.mat.test.ada[1,2])
```

The table below shows the classification error, sensitivity and specificity values for the methods:

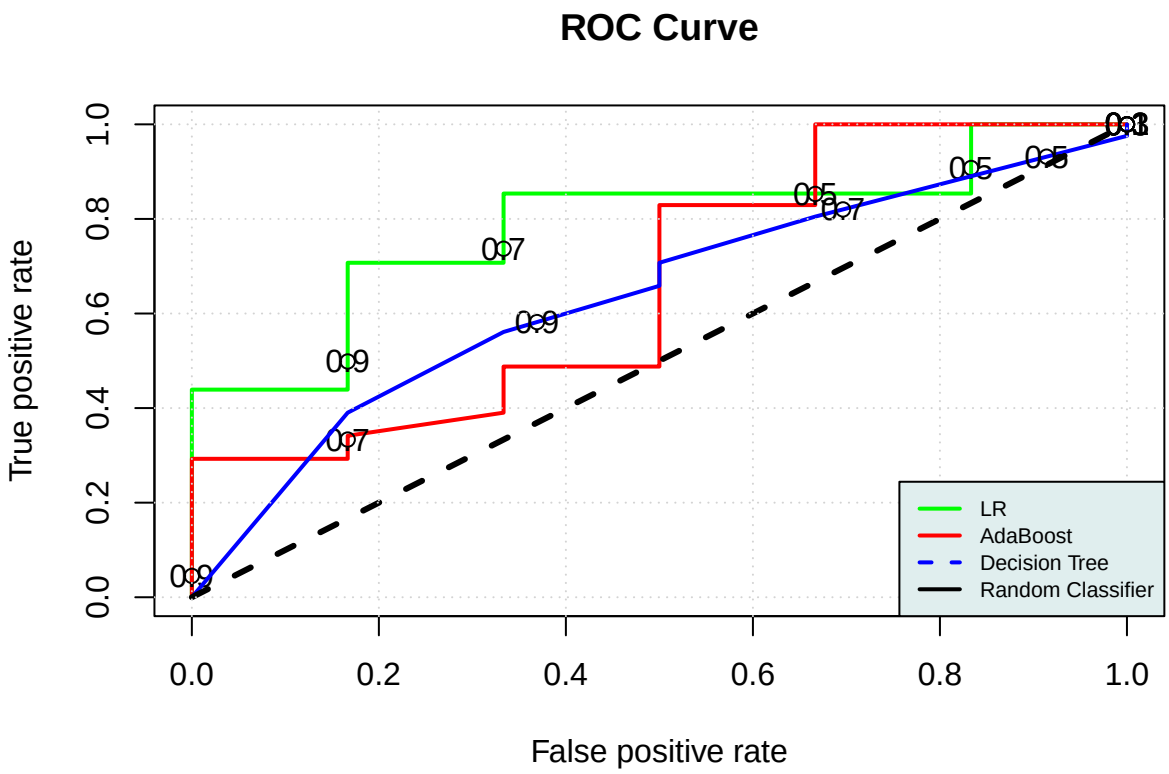
- LR,
- AdaBoost,
- classification trees.

Table 5: Classification error, sensitivity and specificity values

	LR	AdaBoost	tree
Classification error	0.128	0.234	0.255
Sensivity	0.889	0.895	0.892
Specificity	0.500	0.222	0.200

It is also possible to see what the ROC curve looks like and how much the area under the curve (AUC) is.

```
true.labels <- test.set$Class

pred.ROCR.logit <- prediction(pred.prob, true.labels)
perf.ROCR.logit <- performance(pred.ROCR.logit, "tpr", "fpr")

pred.ROCR.ada <- prediction(pred.adaboost$prob[,2], true.labels)
perf.ROCR.ada <- performance(pred.ROCR.ada, "tpr", "fpr")

tree.preds <- predict(tree.default, test.set, type="prob")[, 2]
pred.ROCR.tree <- prediction(tree.preds, true.labels)
perf.ROCR.tree <- performance(pred.ROCR.tree, "tpr", "fpr")

plot(perf.ROCR.logit, print.cutoffs.at=seq(0.1, 1, 0.2), colorize=FALSE, col = "green", lwd=2)
plot(perf.ROCR.ada, print.cutoffs.at=seq(0.1, 1, 0.2), colorize=FALSE, col="red", add=T, lwd=2)
plot(perf.ROCR.tree, print.cutoffs.at=seq(0.1, 1, 0.2), colorize=FALSE, col="blue", add=T, lwd=2)

# add the ROC curve for the random classifier
lines(c(0,1), c(0,1), lwd=3, lty=2)
grid()
title("ROC Curve")
legend("bottomright",lty=c(1,1,2), lwd=2, col=c("green","red","blue", "black"),
       legend=c("LR", "AdaBoost", "Decision Tree","Random Classifier"), bg="azure2", cex=0.7)
```




```
AUC.logit <- performance(pred.ROCR.logit, "auc")@y.values[[1]]
AUC.ada  <- performance(pred.ROCR.ada, "auc")@y.values[[1]]
AUC.tree  <- performance(pred.ROCR.tree, "auc")@y.values[[1]]

df.auc <- data.frame(AUC.logit, AUC.ada, AUC.tree)
colnames(df.auc) <- c("LR", "AdaBoost", "tree")
rownames(df.auc) <- "AUC"

df.auc %>%
  kable(booktabs = TRUE, digits = c(3,3,3), align = "c", caption = "AUC values") %>%
  kableExtra::column_spec(c(2:4), width = "2.5cm") %>%
  kableExtra::column_spec(1, width = "1.5cm") %>%
  kableExtra::kable_styling(latex_options = "HOLD_position")
```

Table 6: AUC values

	LR	AdaBoost	tree
AUC	0.785	0.663	0.636

For this dataset, the best classificator is logistic regression.

3.3 Description of the selected model

As can be seen, the smallest classification error comes out for logistic regression (almost 13 of misassignments). Sensitivity shows that over 89 of the participants were correctly classified into the class of patients who died, and specificity - 50 of the observations were correctly classified into the survival class when they actually survived.

Initially, the model is created on the training data, then the variables relevant to the model are selected - in our case this is age, sex and fatigue (leaving variables irrelevant to the model does not affect the result, although it may lead to overfitting). The model was reduced to the 3 variables mentioned earlier. Below are the output of the reduced model.

```
##
## Call:
## glm(formula = Class ~ AGE + SEX + FATIGUE, family = binomial(link = "logit"),
##     data = learning.set)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.67266   0.00002   0.23093   0.79775   1.42353
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   20.43808 1708.22619   0.012  0.99045
## AGE           -0.04541    0.02298  -1.976  0.04813 *
## SEXM         -18.00322 1708.22571  -0.011  0.99159
## FATIGUE1       2.83373    1.05573   2.684  0.00727 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 119.22  on 107  degrees of freedom
## Residual deviance:  89.49  on 104  degrees of freedom
## AIC: 97.49
##
## Number of Fisher Scoring iterations: 17
```

Only age and fatigue are statistically significant. Below there are odds ratio and their confidence intervals.

```
##      (Intercept)              AGE             SEXM         FATIGUE1
## "751877083.0288" "        0.9556" "        0.0000" "       17.0088"
```

The coefficient estimation of the *AGE* variable says that if age increases by 1 year, the probability of death will be 4.5% lower, if fatigue is present, the probability of death will increase 16 times.

Below they find a scatter plot and histogram of the predicted logarithms of the odds ratio

The scatter and histogram of the residuals is as follows:

From the scatter plot, it can be seen that most of the residuals are "in one line" - between 0 and 1, which can also be seen in the histogram.

Next, there will be predicted the probability of survival under the condition of the phenomena in question. The *predict* function will be used for this. Below are scatter plots and a histogram of these probabilities.

Next, in order to appropriately assign labels for the test set, a suitable cut-off point had to be found, which is approximately 0.35. Using the *ROCR* package, the *performace*, *which.max* and *slot* functions, a check was made to see when the highest accuracy occurred and this point was used.

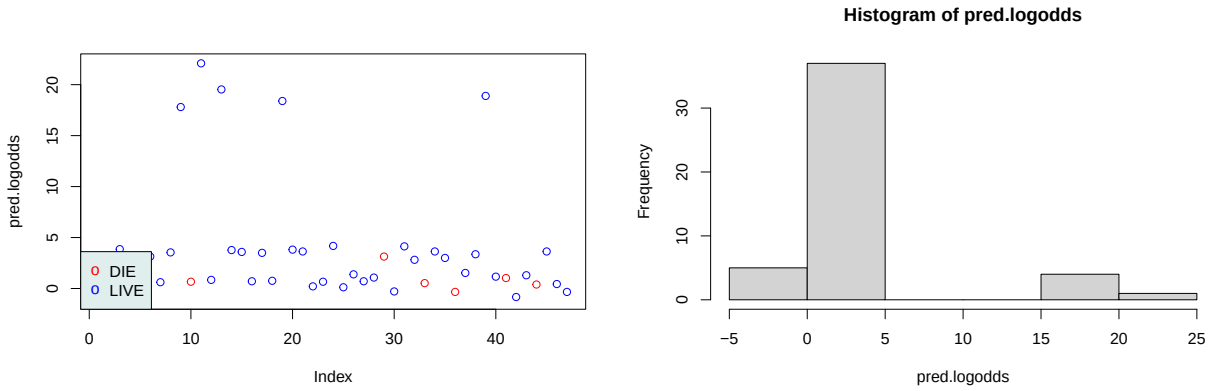


Figure 11: Scatter plot and histogram of predicted odds ratio logarithms

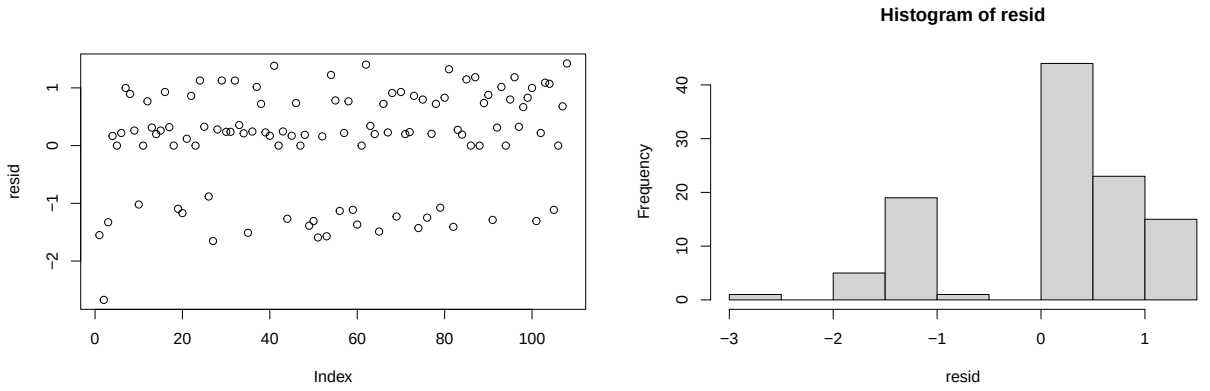


Figure 12: Scatter plot and histogram of residuals

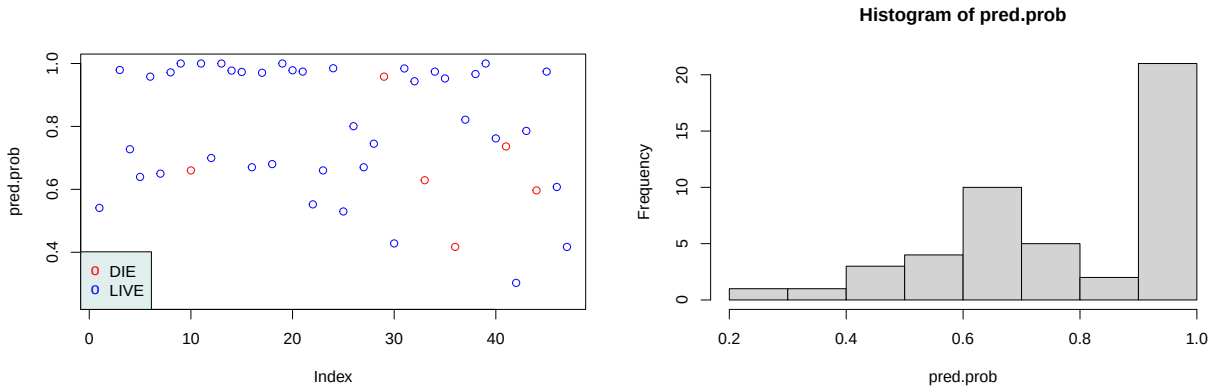


Figure 13: Scatter plor and histofram of survival probability

```
acc.perf <- performance(pred.ROCR.logit, measure = "acc")
ind <- which.max(slot(acc.perf, "y.values")[[1]] )
acc <- slot(acc.perf, "y.values")[[1]][ind]
cutoff <- slot(acc.perf, "x.values")[[1]][ind]
```

Once the appropriate cut-off point was found, the confusion table for the logistic regression model can be shown.

Table 7: Confusion matrix

	DIE	LIVE
DIE	1	1
LIVE	5	40

As it is seen, only 7 out of 47 observations was badly classified, so the classifier is pretty accurate.