Clustering Gene-Expression Data

This assignment involves a bioinformatics problem specifically unsupervised clustering of gene-expression micro-array data. Two scenarios are considered:

- Clustering different tissue samples based on their gene-expression levels across multiple genes (Activity 1).
- Clustering genes according to their gene-expression levels across multiple experimental conditions (Activity 2 and Activity 3).

To run the hierarchical cluster analysis following packages need to be installed:

```
library(foreign)
library(dplyr)
library(stats)
library(graphics)
library(factoextra)
library(dbscan)
library(reshape)
library(reshape2)
library(dendextend)
library(tidyverse)
```

Activity 1: Clustering cancerous tissue samples

In this first activity, we consider the problem of clustering tissue samples based on their gene-expression levels. This is a relevant problem in bioinformatics as it can help the discovery of different subtypes of cancer. In particular, we will use a dataset from the study in Golub *et al.*, which contains 72 human samples with leukemia. The expression levels of 1868 selected genes have been measured in all these samples. The dataset thus contains 72 observations (rows) and 1868 variables (columns). It is available in the file *golub-1999 -v1_database.arff*.

1. Read the dataset directly from the ARFF file into a data frame.

```
  golub <- read.arff("/Users/Biljana/Data Mining/Ass 2/golub-1999-v1_database.arff") # read data
               from Weka Attribute-Relation File Format (ARFF) files into a data frame (golub)

  dim(golub) # retrieve dimensions of the data frame
## [1]   72 1869
  sum(is.na(golub)) # calculate the overall number of missing values in the data frame
## [1] 0

  levels(golub$Classe) # factor levels of the categorical variable Classe
## [1] "1" "2"
```

2. Set aside the rightmost column (containing the class labels) from the data, storing it separately from the remaining data frame (with the 1868 predictors).

```
golub_classe <- golub %>% select(-c(1:1868)) # allocate the rightmost variable Classe from the
                                              data frame by dropping a sequence of variables (1:1868)

golub_predictors <- golub %>% select(c(1:1868)) # store the remaining sequence of predictors
                                                 (1:1868) variables in a data frame (golub_predictors)
```

3. Use the 72 x 1868 data frame to compute a matrix containing all the pairwise Euclidean distances between observations, that is, a 72 x 72 matrix with distances between tissue samples according to their 1868 expression levels. This matrix must be of type dist, which can be achieved either by using the function dist() from the base R package stats or by coercion using the function as.dist().

```
golub_matrix <- dist(golub_predictors, method = "euclidean", diag = FALSE, upper = FALSE,
                     p = 2) # distance matrix computation of class "dist" object by using the
                            specified distance measure (euclidean) that computes the
                            dissimilarity distances between the rows of a data matrix
```

The distance measures are usually written for two vectors x and y. Euclidean distance is defined between the two vectors as (2 norm aka L2), √∑i(xi−yi)2.

```
head(as.matrix(golub_matrix)) # conversion to conventional distance matrix
```

```
##          1        2        3        4        5        6        7        8
## 1     0.00 43921.21 41846.80 31997.37 33678.22 39833.04 30194.68 27456.28
## 2  43921.21     0.00 35482.87 40790.43 38868.76 30269.22 44915.40 37786.18
## 3  41846.80 35482.87     0.00 36976.48 37225.85 22505.93 39069.55 34970.39
## 4  31997.37 40790.43 36976.48     0.00 17658.39 33297.49 28216.52 22093.27
## 5  33678.22 38868.76 37225.85 17658.39     0.00 33840.96 31694.20 25103.92
## 6  39833.04 30269.22 22505.93 33297.49 33840.96     0.00 36408.55 31175.04
```

4. Use the distance matrix as input to call the Single-Linkage clustering algorithm available from the base R package stats and plot the resulting dendrogram. Do not use any class labels to perform this step.

```
golub_sl <- hclust(golub_matrix, method = "single") # hierarchical cluster analysis with the
                                                     Single-Linkage clustering algorithm

plot(golub_sl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "red3", labels = FALSE,
     main = "Cluster Dendrogram with Single Linkage Method") # plot the dendrogram with
                                                              Single-Linkage method (see Fig 1)
```

It is an agglomerative hierarchical clustering approach on a set of dissimilarities of golub_matrix, a dissimilarity structure as produced by dist() function, by utilizing the agglomeration method, the single linkage method: "single", that is closely related to the minimal spanning tree and adopts a "friends of friends" clustering strategy. It computes minimal inter-cluster dissimilarity or in another words, computes all pairwise dissimilarities between the observations in one cluster and the observations in another cluster, and records the smallest of these dissimilarities.

5. Use the distance matrix as input to call the Complete-Linkage clustering algorithm available from the base R package stats and plot the resulting dendrogram. Do not use any class labels to perform this step.

```
golub_cl <- hclust(golub_matrix, method = "complete") # hierarchical cluster analysis with the
                                                       Complete-Linkage clustering algorithm

plot(golub_cl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "turquoise3", labels = FALSE,
     main = "Cluster Dendrogram with Complete Linkage Method") # plot the dendrogram with
                                                                Complete-Linkage method (see Fig 2)
```

It is an agglomerative hierarchical clustering approach on a set of dissimilarities of golub_matrix, a dissimilarity structure as produced by dist() function, by using the agglomeration method, the complete linkage method: "complete" that finds similar clusters. It computes maximal inter-cluster dissimilarity and generates all pairwise dissimilarities between the observations in one cluster and the observations in another cluster. At the end, it records the largest of these dissimilarities.

6. Use the distance matrix as input to call the Average-Linkage clustering algorithm available from the base R package stats and plot the resulting dendrogram. Do not use any class labels to perform this step.

```
golub_al <- hclust(golub_matrix, method = "average") # hierarchical cluster analysis with the
                                                       Average-Linkage clustering algorithm

plot(golub_al, xlab = "", sub = "", cex = 0.6, hang = -1, col = "slateblue2", labels = FALSE,
     main = "Cluster Dendrogram with Average Linkage Method") # plot the dendrogram with
                                                               Average-Linkage method (see Fig 3)
```

It is an agglomerative hierarchical clustering on a set of dissimilarities of golub_matrix, a dissimilarity structure as produced by dist() function, by using the agglomeration method (the average linkage method: "average"). The average linkage method computes mean inter-cluster dissimilarities. In another words, it computes all pairwise dissimilarities between the observations in one cluster and the observations in another cluster, and records the average of these dissimilarities.

7. Use the distance matrix as input to call Ward's clustering algorithm available from the base R package stats and plot the resulting dendrogram. Do not use any class labels to perform this step.

```
golub_wl <- hclust(golub_matrix, method = "ward.D2") # hierarchical cluster analysis with
                                                       Ward's 2 clustering algorithm

plot(golub_wl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "maroon3", labels = FALSE,
     main = "Cluster Dendrogram with Ward 2 Linkage Method") # plot the dendrogram with
                                                               Ward's 2 Linkage method (see Fig 4)
```

It is an agglomerative hierarchical clustering approach on a set of dissimilarities of golub_matrix, a dissimilarity structure as produced by dist() function, by using the agglomeration method (the Ward's linkage method: "Ward.D2"). The Ward's error sum of squares hierarchical clustering method was first published by Joe Ward (Ward, 1963). Two algorithms are implementing Ward's clustering method: Ward 1 described by Murtagh Fionn (Murtagh, 1985) and Ward 2 described in Kaufman & Rousseeuw, 1990. When applied to the same distance matrix, they produce different results. It was shown that Ward 2 algorithm preserves the Ward's criterion from 1963 (Murtagh & Legendre, 2014) where the dissimilarities are squared before cluster updating.

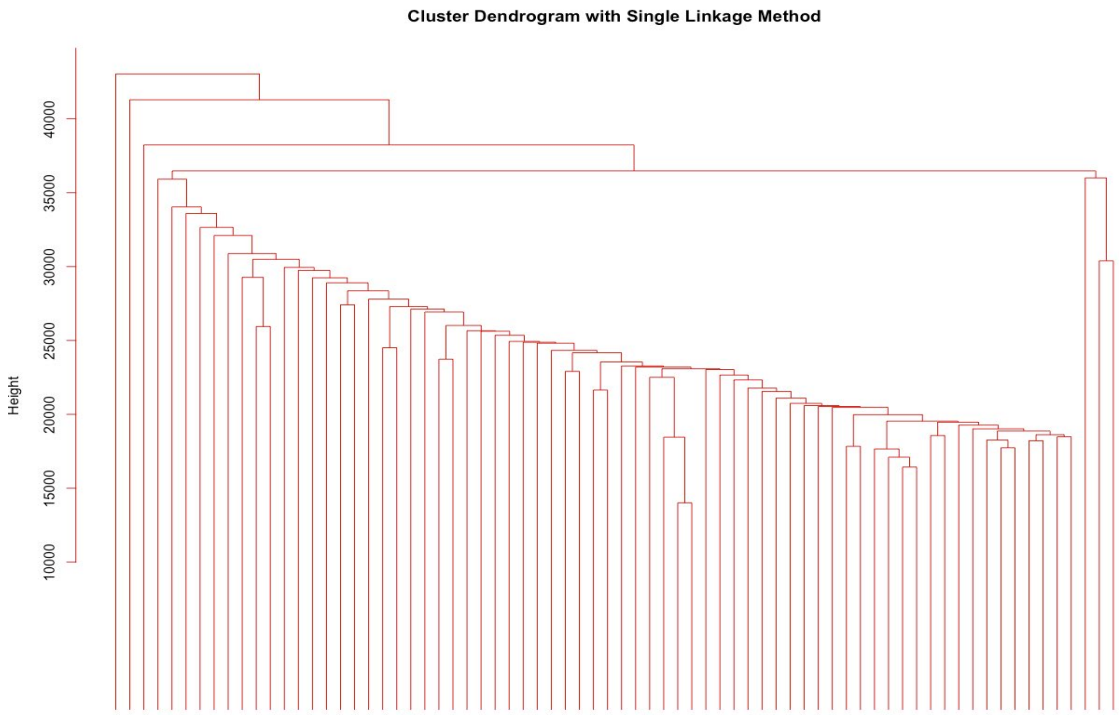

Fig 1: Agglomerative hierarchical cluster dendrogram of type single on Euclidiean dissimilarity distance matrix of the entire dataset “golub-1999-v1_database” without class labels.

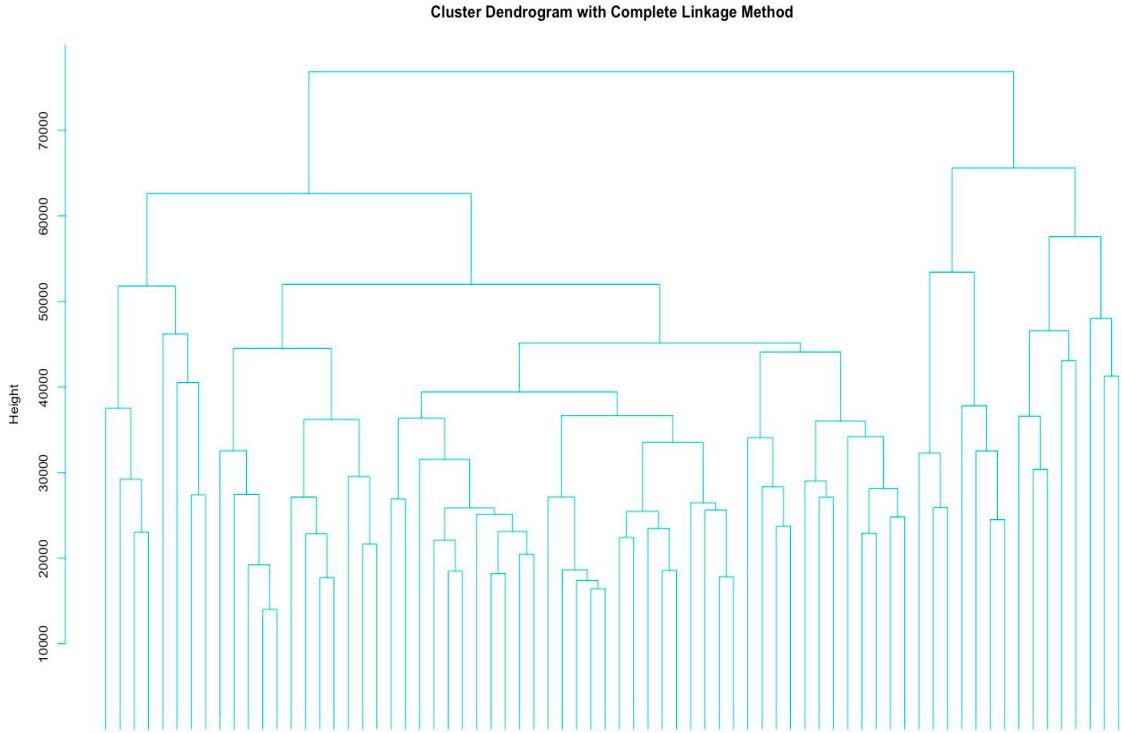

Fig 2: Agglomerative hierarchical cluster dendrogram of type complete on Euclidiean dissimilarity distance matrix of the entire dataset “golub-1999-v1_database” without class labels.

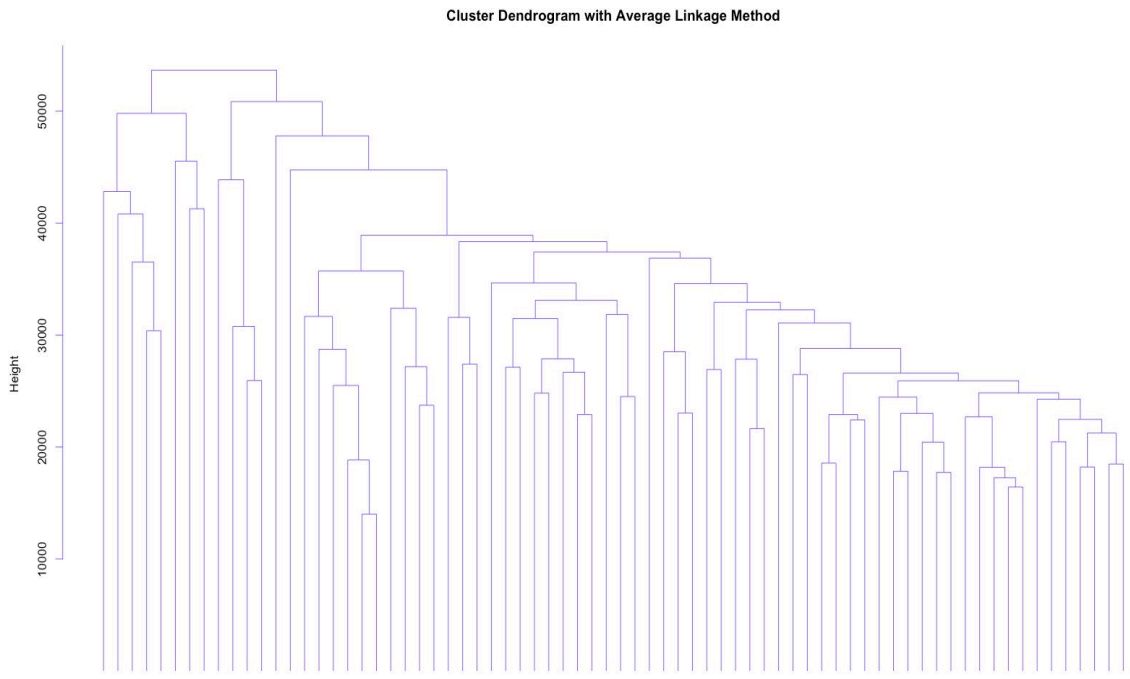

Fig 3: Agglomerative hierarchical cluster dendrogram of type average on Euclidean dissimilarity distance matrix of the entire dataset “golub-1999-v1_database” without class labels.

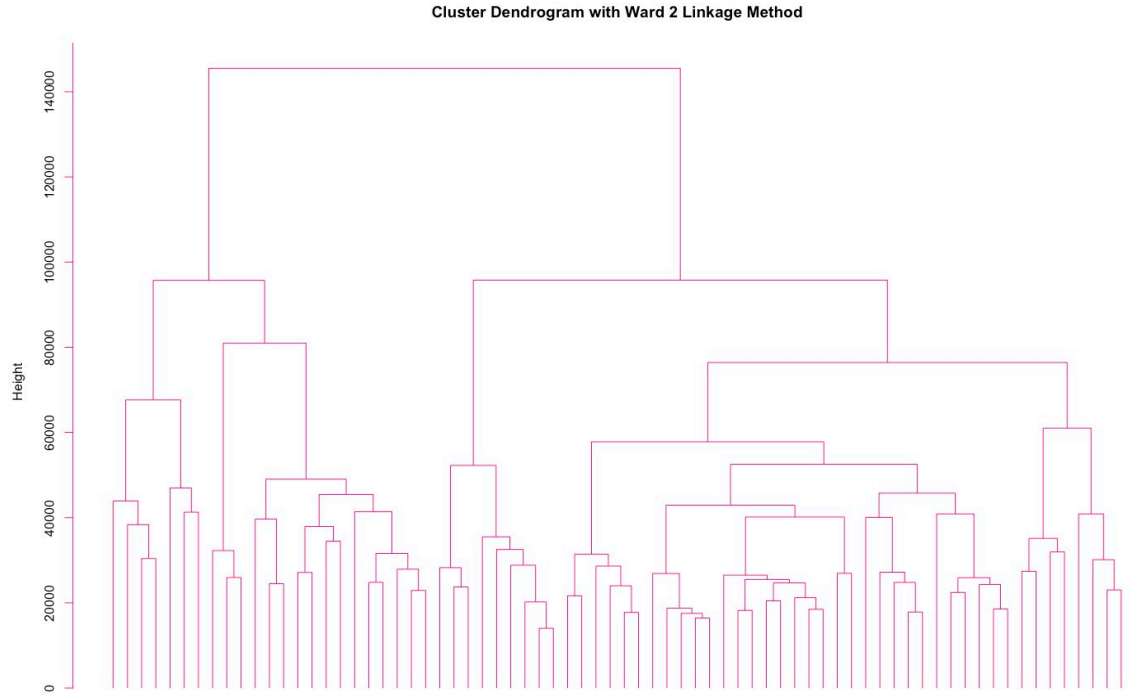

Fig 4: Agglomerative hierarchical cluster dendrogram of type Ward on Euclidean dissimilarity distance matrix of the entire dataset “golub-1999-v1_database” without class labels.

8. Compare the dendrograms plotted in Items 4 to 7. Visually, the dendrograms suggest that some clustering algorithm(s) generate more clear clusters than the others. In your opinion, which algorithm(s) may we be referring to and why? In particular, in which aspects do the results produced by this/these algorithm(s) look more clear?

Perform Item 9 below only for this/those algorithm(s).

```
# Plot dendrograms with Single, Complete, Average and Ward's 2 Linkage method together:

  opar <- par(mfrow = c(2, 2))
  plot(golub_sl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "red3", labels = FALSE,
       main = "Cluster Dendrogram with Single Linkage Method")
  plot(golub_cl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "turquoise3", labels = FALSE,
       main = "Cluster Dendrogram with Complete Linkage Method")
  plot(golub_al, xlab = "", sub = "", cex = 0.6, hang = -1, col = "slateblue2", labels = FALSE,
       main = "Cluster Dendrogram with Average Linkage Method")
  plot(golub_wl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "maroon3", labels = FALSE,
       main = "Cluster Dendrogram with Ward 2 Linkage Method")
  par(opar)
```

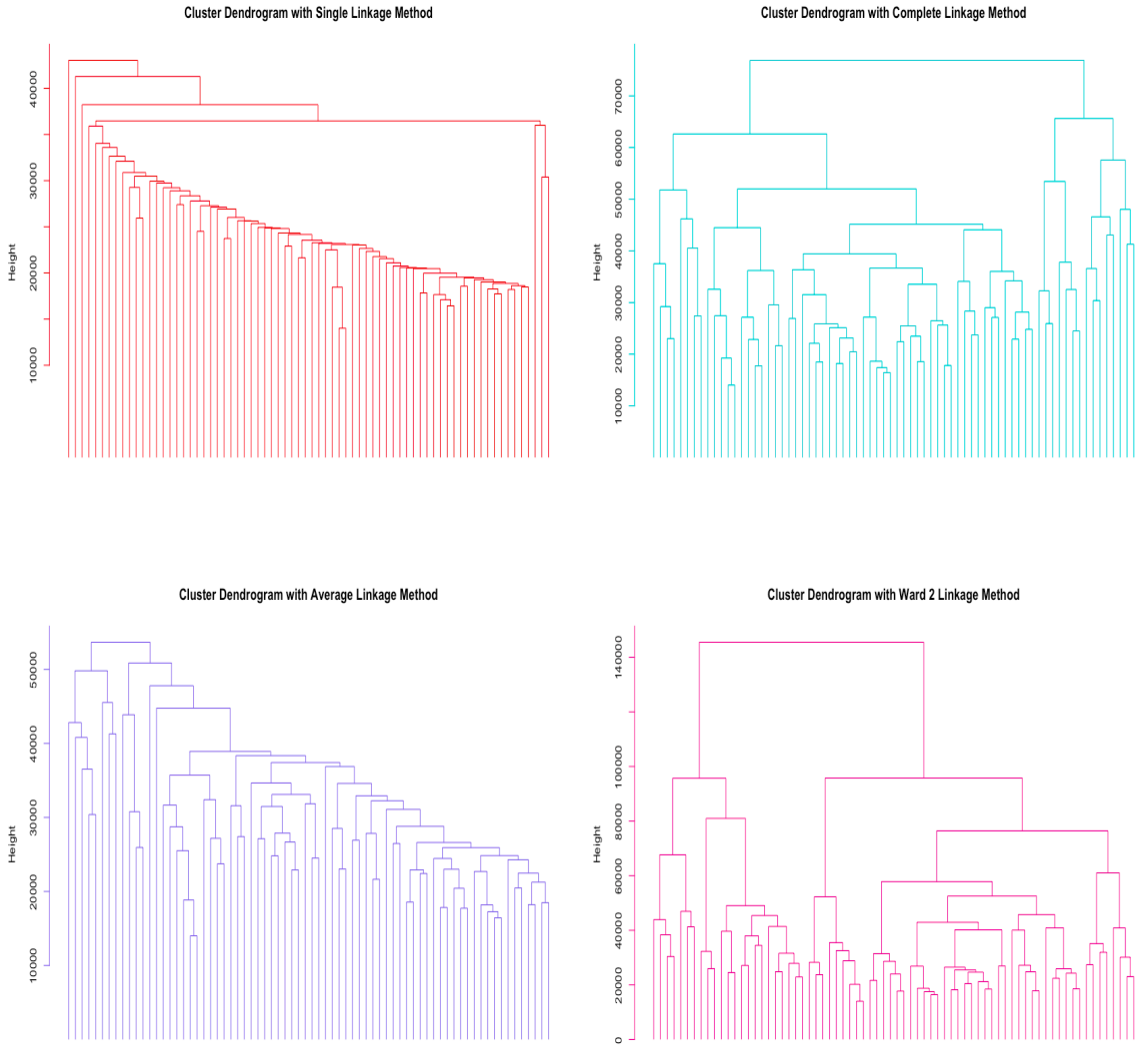

Fig 5: Computed four cluster dendrograms of hierarchical cluster analysis with Single, Complete, Average and Ward's 2 methods on the entire dataset "golub-1999-v1_database" without class labels.

The observed four dendrograms are compact visualizations of final dissimilarity matrices measured with four different agglomerative methods of hierarchical clustering. Visually, the clustering solutions that depend on Complete-Linkage and Ward's 2 Linkage methods are preferred the most because they yielded more balanced dendrograms with clearly visible clusters. In addition, these dendrograms better demonstrate the general structure of the dataset labeled with the putative class label (variable Classe) that has two factor levels "1" and "2".

Complete-Linkage or the "maximum" method measures the dissimilarities between the merged pair and the others as the maximum of the pair dissimilarities in each case. The process is repeated until all objects are being clustered. There are 72 objects, thus, there are 71 sequential steps (n-1) until final tree is obtained where all objects are in a single cluster. However, by analyzing the structure of the dendrogram, we can notice that there are 14 pair of objects of class 1 and 8 pair of objects of class 2 that are very similar at the initial step, and the others 19 objects of class 1 and 8 objects of class 2 separate themselves from the others (Sarstedt & Mooi, 2014). The inter-object dissimilarity is less than the level all objects clustered because of maximum method applied. Height at 60 000 is the point where all objects are as similar as they are dissimilar to one another, therefore it is a convenient point to cut the tree leaving the cluster of objects which can be used to create a categorical variable with subcategories of class 1 and class 2 that can be used to better generalize the structure of the dataset. As well, by cutting off the dendrogram at various heights, different numbers of clusters will appear that will give some partial ordering information about the expression levels of the genes. As a result, we can build expression matrix of orderings of genes.

The clustering solution that depends on Ward's 2 algorithm or Ward's minimum variance method is based on criterion that the dissimilarities between the merged pair and the other object at each step are measured as the optimal value of an objective function that is the error sum of squares which minimizes the "total" within cluster variance. However, it assumes that the initial cluster distance between objects is defined as squared Euclidean distance. This method measures how much the sum of squares will increase when the pair will merge. Squaring of dissimilarities at each step resulted into dendrogram that has very similar morphology with the visual output of Complete-Linkage method, so we say that trees are almost wreath product invariant. Ward's 2 method very efficiently minimizes the Ward's clustering criterion and produces the Ward's method. As well, it captures the general structure of the dataset very intuitively.

The choice of the clustering solution to quantify dissimilarities between two clusters is important in analyzing the data. The computed dendrograms depend strongly on the type of linkage used and on the choice of dissimilarity measure. It is interesting to notice that the genes that are grouped together using one method are not necessarily grouped together using another method. It is especially visible when we compare Single-Linkage dendrogram with the others. The Single-Linkage or "minimum" method forms a cluster when only one pair of dissimilarities (not all) is less than a particular level, thus at every further step it selects the minimum of the two distances and two objects can be merged when there is a single close link between them. This pattern leads to formation of clusters that are heterogeneous internally and arbitrarily shaped externally but that is in contradiction with the general aim of clustering to obtain homogeneous clusters. In another words, the Single-Linkage dendrogram produced unbalanced groups with long thin clusters (James, Witten, Hastie, & Tibshirani, 2013).

The Average-Linkage algorithm generates clustering solutions that are somewhere in between Single-Linkage and Complete-Linkage algorithms because it computes the average dissimilarities at each step. If the dissimilarity between A and B is 50 000, and between A and C is 60 000, then Complete-Linkage method selects 60 000, Single-Linkage method selects 50 000 and the Average method selects 55 000.

9. Redraw the dendrogram(s) for the selected algorithm(s) in Item 8, now using the class labels that you stored separately in Item 2 to label the observations (as disposed along the horizontal axis of the dendrogram). Do some prominent clusters in the dendrogram(s) correspond approximately to the classes (that is, the two subtypes of leukemia)?

```
Classe <- golub_classe$Classe # assigning variable as a vector

# Plot cluster dendrogram with Complete-Linkage algorithm used along with class labels:
  dendrogram_cl <- as.dendrogram(golub_cl)
  par(mar = c(12,4,1,1))
  dendrogram_cl %>%
    set("labels_col", value = c("blue", "maroon3"), k = 2) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("blue", "maroon3"), k = 2) %>%
    place_labels(paste(golub_classe$Classe, sep = "_")) %>%
    plot( main = "Cluster Dendrogram with Complete Linkage Method")
  dendrogram_cl_rect <- rect.dendrogram(dendrogram_cl, k = 2, lty = 5,
                        lwd = 0, x = 1, col = rgb(0.1, 0.2, 0.4, 0.1))
```

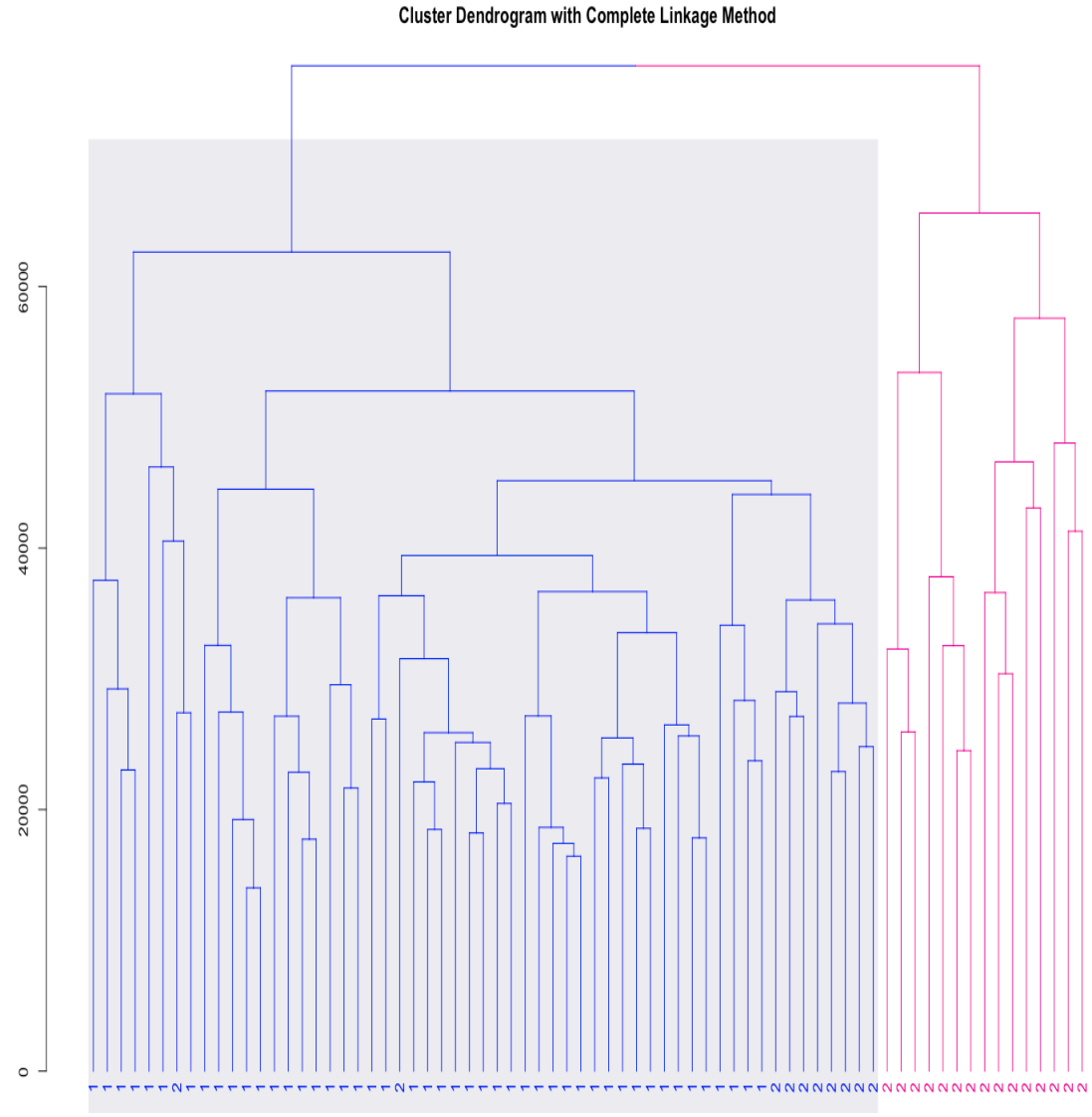

Fig 6: Cluster dendrogram with Complete Linkage algorithm applied on Euclidean distance matrix of the entire dataset "golub-1999-v1_database" with class labels ("1" and "2").

```r
# Plot cluster dendrogram with Ward's 2 Linkage algorithm used along with class labels:
  dendrogram_wl <- as.dendrogram(golub_wl)
  par(mar = c(12,4,1,1))
  dendrogram_wl %>%
    set("labels_col", value = c("blue", "maroon3"), k = 2) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("blue", "maroon3"), k = 2) %>%
    place_labels(paste(golub_classe$Classe, sep = "_")) %>%
    plot( main = "Cluster Dendrogram with Ward 2 Linkage Method")
  dendrogram_wl_rect <- rect.dendrogram(dendrogram_wl, k = 2, lty = 5, lwd = 0, x = 1,
                                        col = rgb(0.1, 0.2, 0.4, 0.1))
```

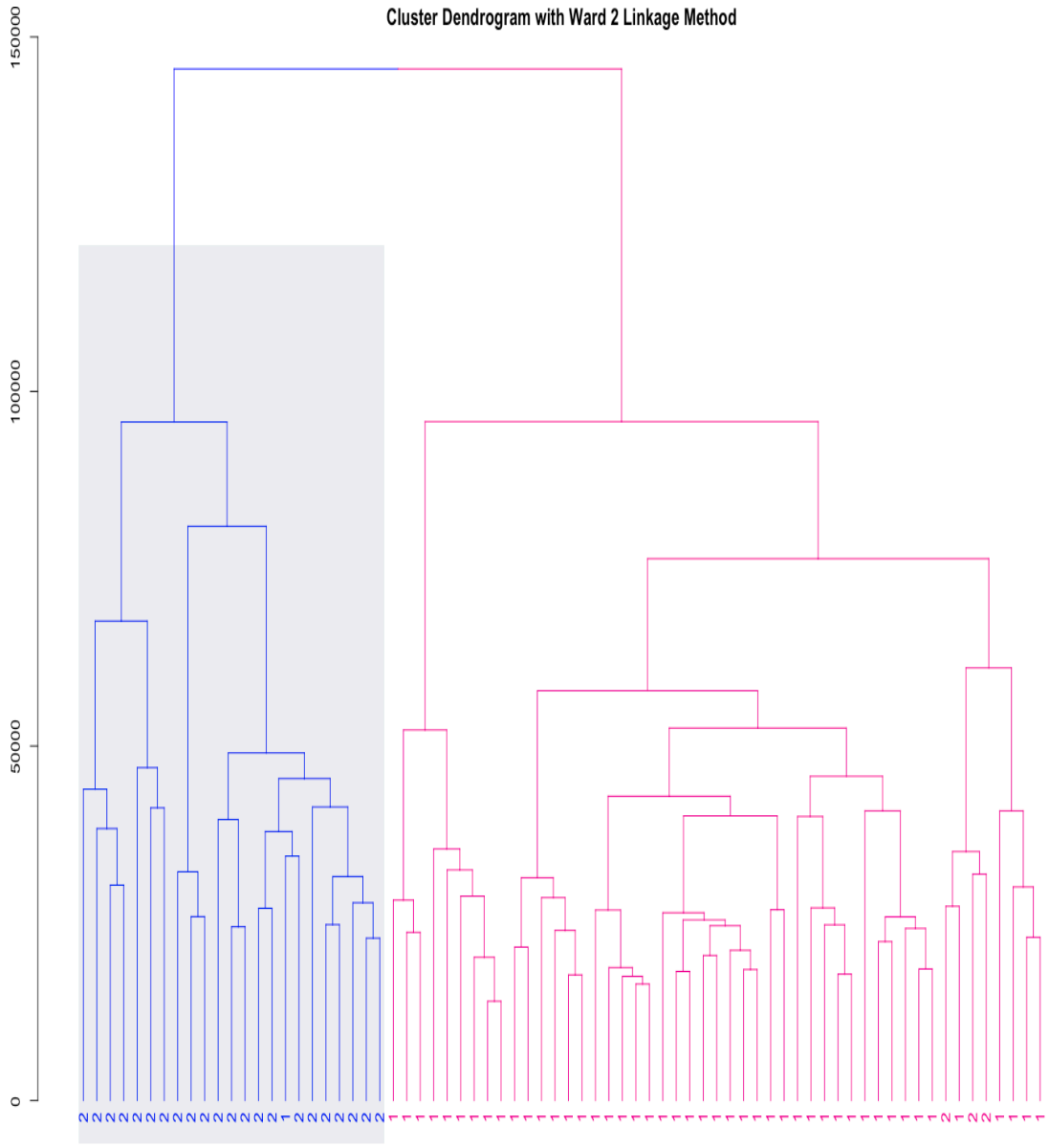

Fig 7: Cluster dendrogram with Ward's 2 Linkage algorithm applied on Euclidean distance matrix of the entire dataset "golub-1999-v1_database" with class labels ("1" and "2").

The clustering solutions that depend on Complete-Linkage and Ward's 2 Linkage methods effectively generalize the structure of the dataset by clearly showing a correlation between the classes and the cluster formations. However, focus is on Ward's 2 dendrogram (Fig 7) because it captured the objects more precisely than Complete-Linkage

dendrogram (Fig 6). Namely, 46 objects out of 47 objects belonging to the ground truth class label of subtype ALL(class ”1”) were grouped in one cluster (cluster 1) and 42 out of 45 objects of subtype AML(class ”2”) were grouped in another cluster (cluster 2). It is obviously certain that the prominent clusters in the Ward’s 2 dendrogram correspond to the classes of the two subtypes of leukemia. In addition, we can indicate that those three objects that are members of class “2” and that are being grouped within cluster 1 are apparently closer to objects in cluster 1 suggesting that they might be considered as subgroups of cluster 1. The same holds true for one objects of class “1” within cluster 2. In conclusion, unsupervised machine learning method (clustering) is very effective tool in revealing the structure of the dataset in correlation with the class labels known as ground truth.

10. Repeat the analysis, now using normalised data. The 1868 predictors have not been normalised before computing the distance matrix in Item 3. Normalisation is a non-trivial aspect in unsupervised clustering, as there is no ground truth to assess whether or not it improves performance. On the one hand, it may, prevent variables with wider value ranges to dominate distance computations, but on the other hand it, may distort clusters by removing natural differences in variance that help characterise them as clusters. Normalisation thus becomes an aspect of Exploratory Data Analysis when it comes to clustering: the analyst will usually generate and try to interpret results both with normalised and non-normalised versions of the data. The type of normalisation depends on the application in hand. Here, we are computing Euclidean distance between rows of the dataset, so the type of normalisation that applies is typically the so-called z-score normalisation of columns, where each column is rescaled to have zero mean and standard deviation of 1. In this item, you are first asked to normalise the data this way before computing the distance matrix in Item 3.Then, repeat Items 4 to 9. Does normalisation improve or worsen the results in this dataset?

```
# Step 3 - Normalize the data and compute the distance matrix with Euclidean method of measure
           the distances between the rows of a data matrix:

  golub_scaled <- scale(golub_predictors) # standardize the variables to have zero mean and
                                           standard deviation 1
  round(sd(golub_scaled[,45], 0)) # check the function on random selected variables
## [1] 1

  round(mean(golub_scaled[,1345], 0)) # check the function on random selected variables
## [1] 0

  golub_matrix_scaled <- dist(golub_scaled, method = "euclidean", diag = FALSE, upper = FALSE,
                              p = 2) # compute distance dissimilarity matrix with Euclidean
                                       distance measure that computes the distances between the
                                      rows of the normalized matrix as an object of class "dist"

# Step 4 to 8 - Hierarchical clustering with Single, Complete, Average and Ward’s 2 Linkage method
                on normalized variables:

  golub_sl_scaled <- hclust(golub_matrix_scaled, method = "single")
  golub_cl_scaled <- hclust(golub_matrix_scaled, method = "complete")
  golub_al_scaled <- hclust(golub_matrix_scaled, method = "average")
  golub_wl_scaled <- hclust(golub_matrix_scaled, method = "ward.D2")

# Plot dendrograms with Single, Complete, Average and Ward’s 2 linkage methods:

  plot(golub_sl_scaled, xlab = "", sub = "", cex = 0.6, hang = -1, col = "red3", labels = FALSE,
       main = "Single Linkage Method on Normalized Data")
  plot(golub_cl_scaled, xlab = "", sub = "", cex = 0.6, hang = -1, col = “turquoise3”,
       labels = FALSE, main = "Complete Linkage Method on Normalized Data")
  plot(golub_al_scaled, xlab = "", sub = "", cex = 0.6, hang = -1, col = “slateblue2”,
       labels = FALSE, main = "Average Linkage Method on Normalized Data")
  plot(golub_wl_scaled, xlab = "", sub = "", cex = 0.6, hang = -1, col = “maroon3”,
       labels = FALSE, main = "Ward 2 Linkage Method on Normalized Data")
```

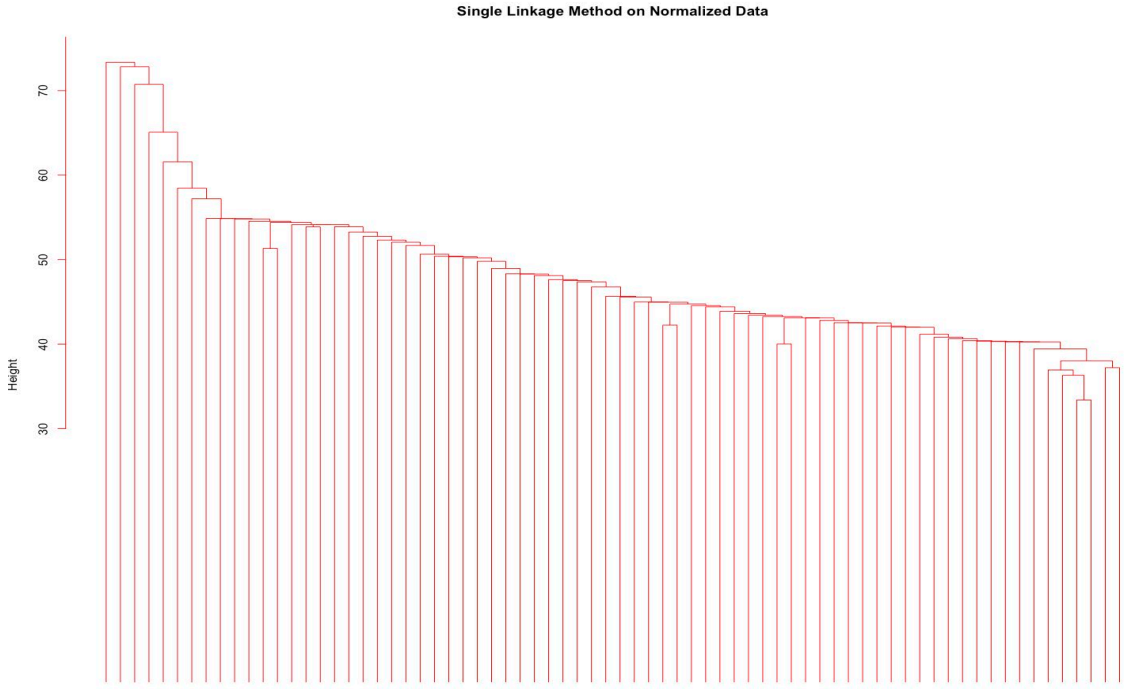

Fig 8: Cluster dendrogram with Single-Linkage algorithm applied on Euclidean distance matrix normalized with standard deviation 1 and mean zero of the entire dataset "golub-1999- v1_database" without class labels.

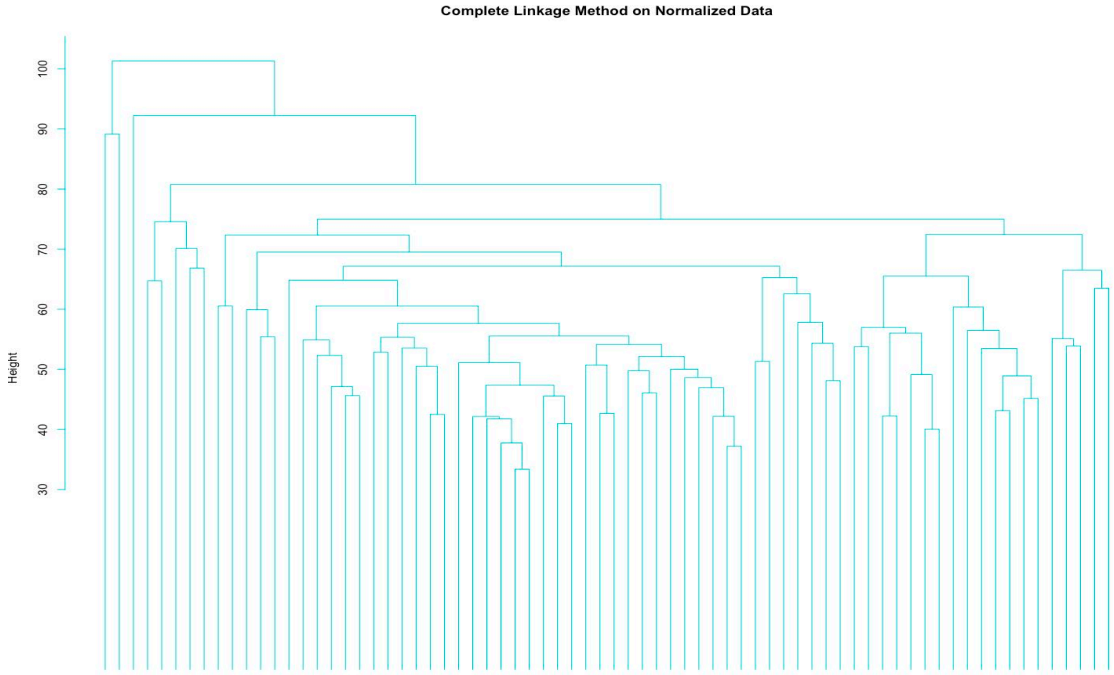

Fig 9: Cluster dendrogram with Complete-Linkage algorithm applied on Euclidean distance matrix normalized with standard deviation 1 and mean zero of the entire dataset "golub-1999-v1_database" without class labels.

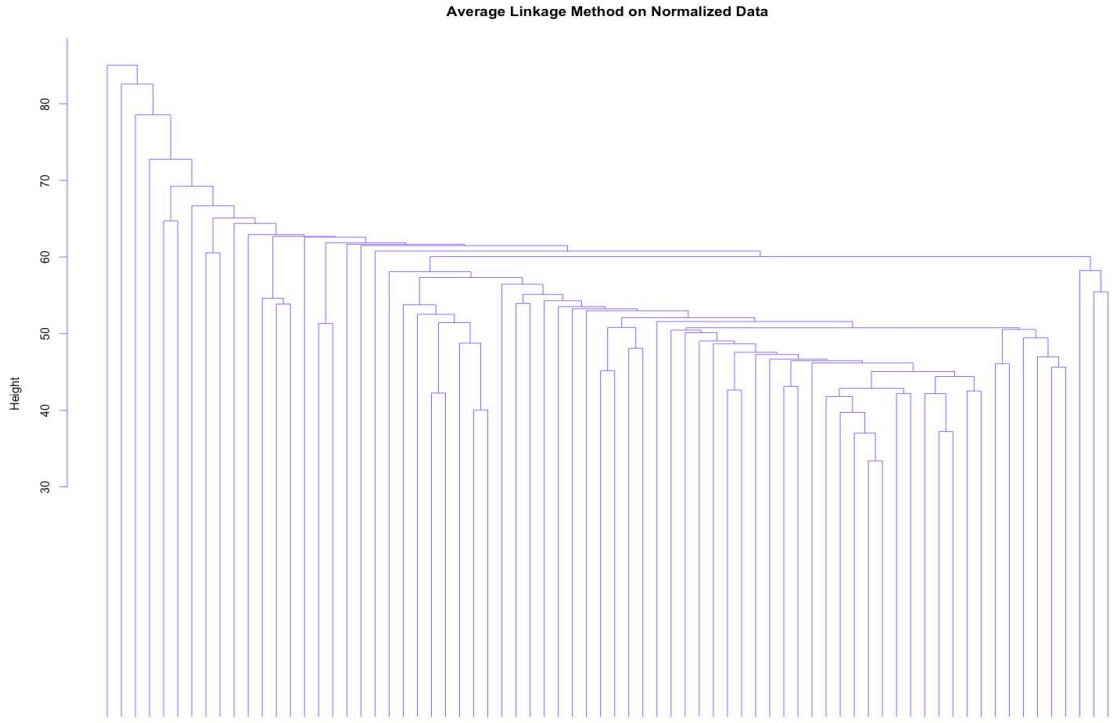

Fig 10: Cluster dendrogram with Average-Linkage algorithm applied on Euclidean distance matrix normalized with standard deviation 1 and mean zero of the entire dataset "golub-1999-v1_database" without class labels.

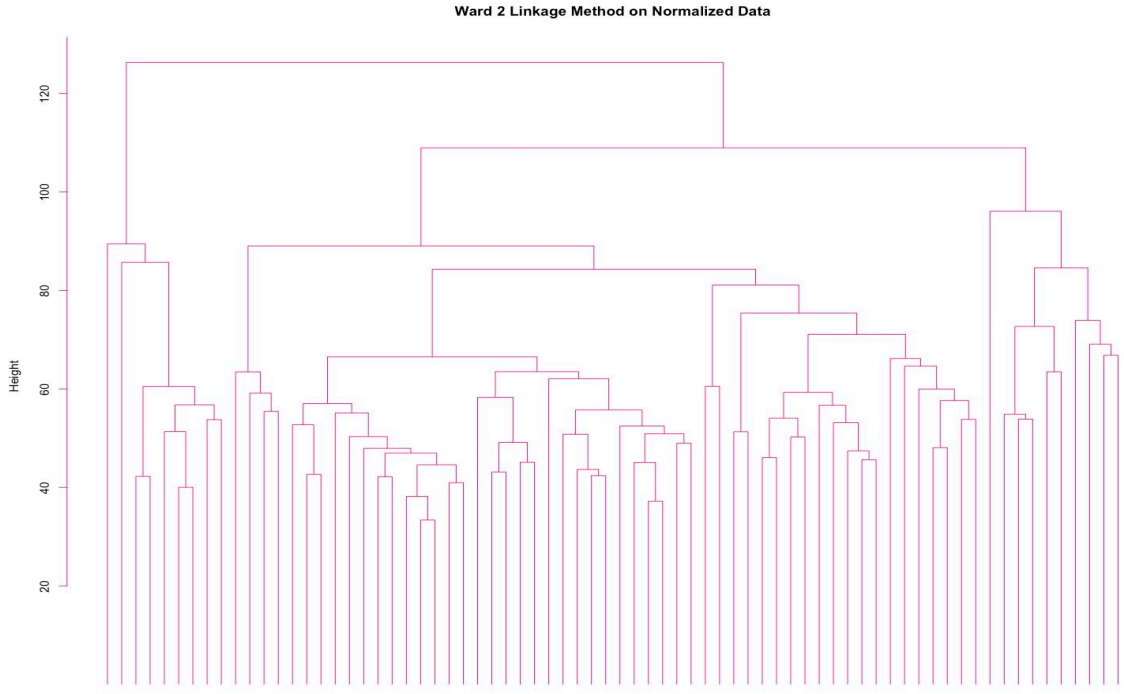

Fig 11: Cluster dendrogram with Ward's 2 Linkage algorithm applied on Euclidean distance matrix normalized with standard deviation 1 and mean zero of the entire dataset "golub-1999-v1_database" without class labels.

```r
# Step 9 - Plot cluster dendrograms with Complete and Ward's 2 Linkage algorithms applied with
          class labels that generate the most clear clusters:

# Dendrogram with Complete-Linkage algorithm on normalized dataset:
  dendrogram_cl_scaled_golub <- as.dendrogram(golub_cl_scaled)
  par(mar = c(12,4,1,1))
  dendrogram_cl_scaled_golub %>%
    set("labels_col", value = c("blue", "maroon3"), k = 2) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("blue", "maroon3"), k = 2) %>%
    place_labels(paste(golub_classe$Classe, sep = "_")) %>%
    plot( main = "Complete Linkage Method on Normalized Data")
  dendrogram_cl_rect_scaled <- rect.dendrogram(dendrogram_cl_scaled_golub, k = 2, lty = 5,
                                               lwd = 0, x = 1, col = rgb(0.1, 0.2, 0.4, 0.1))
```

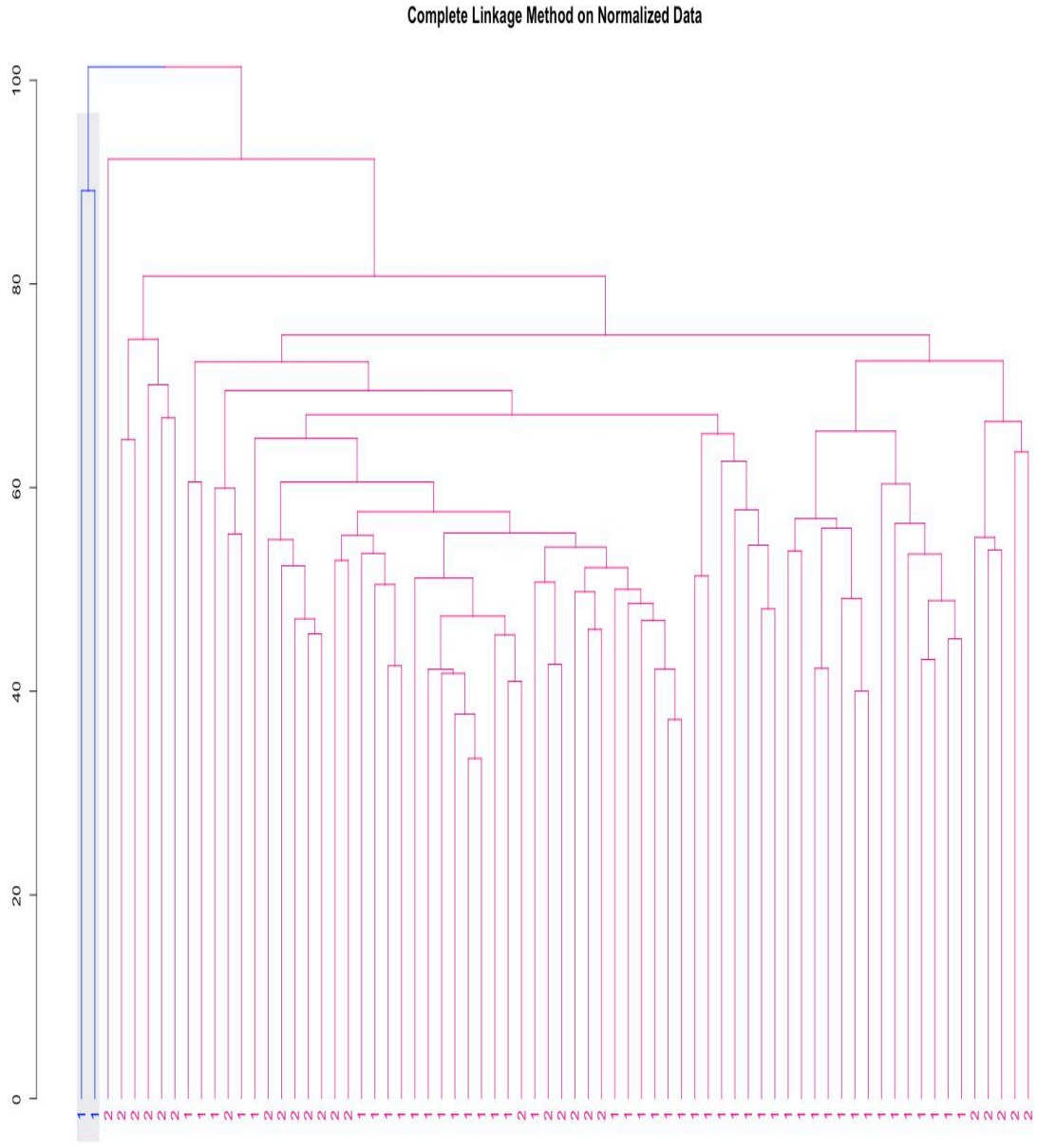

Fig 12: Cluster dendrogram with Complete-Linkage algorithm applied on normalized Euclidean distance matrix of the entire dataset "golub-1999-v1_database" with class labels ("1" and "2").

```r
# Dendrograms with Ward's 2 Linkage algorithm on normalized dataset:
  dendrogram_wl_scaled_golub <- as.dendrogram(golub_wl_scaled)
  par(mar = c(12,4,1,1))
  dendrogram_wl_scaled_golub %>%
    set("labels_col", value = c("blue", "maroon3"), k = 2) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("blue", "maroon3"), k = 2) %>%
    place_labels(paste(golub_classe$Classe, sep = "_")) %>%
    plot( main = "Ward's 2 Linkage Method on Normalized Data")
  dendrogram_wl_rectScaled <- rect.dendrogram(dendrogram_wl_scaled_golub, k = 2, lty = 5, lwd = 0,
                                              x = 1, col = rgb(0.1, 0.2, 0.4, 0.1))
```

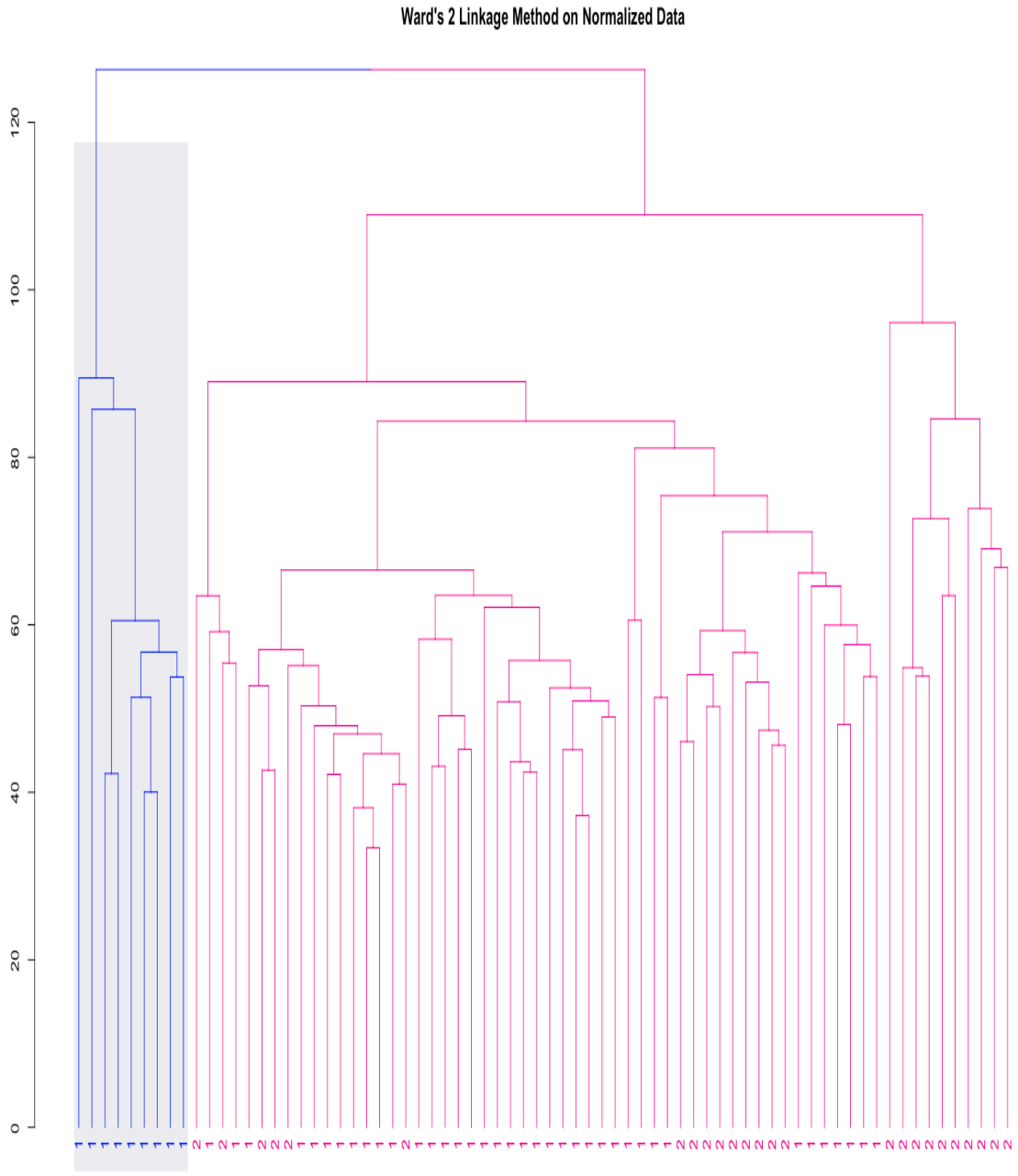

Fig 13: Cluster dendrogram with Ward's 2 Linkage algorithm applied on normalized Euclidean distance matrix of the entire dataset "golub-1999-v1_database" with class labels ("1" and "2").

In general, normalization has to improve the results not to worsen them. From a visual point of view, normalization improves the results but it did not really solve the problem.

The resulted dendrograms on normalized dataset display similar morphology to the counterparts on non-normalized dataset, however, the focus is mainly on Complete-Linkage and Ward's 2 as the best. Both dendrograms yielded two major clusters. We can notice objects that belong to either of the two classes known, as class labels according to the ground truth are not perfectly captured as two prominent clusters in both dendrograms. One common pattern is that objects that belong to label of subtype ALL(class "1") are clearly separated in one group visually significantly reduced. On the other hand, objects that belong to label of subtype AML(class "2") are mixed with objects of class "1" that might suggests close relationships between them and might be considered as subgroups of a bigger group. Further studies need to be performed in order to determine the relationship between the dendrogram evaluation criteria to subgroup differentiation of the genetic structure. In addition, performances of these hierarchical clustering methods need to be compared with model-based clustering methods for the analyzed genetic data. However, accession association studies should be performed in order to find the relationship between the cluster groups and various accessions of genes based on localization in chromatin, protein activity, levels of expression at different experimental conditions and etc. (Odong *et al*., 2011).

As well, we can observe that the unit of height is reduced drastically to 100 and 120 units for Complete-Linkage and Ward's 2 Linkage respectively. Expressing a variable in smaller units leads to a larger range for that variable that has a bigger impact on the resulting dendrogram. In general, with normalization, attempt is made to give all variables an equal weight or equal importance by converting each variable to a unitless measure or relative distance, in hope of obtaining more intuitive dendrogram that will more objectively generalize the real structure of the dataset. However, we have to take into consideration that some variables are more important than the others, and we cannot apply equal weights to all of them, but the assignment of weights should be based on subject-matter knowledge (Abrahamowicz, 1985). Therefore, the dilemma of normalization of data prior quantifying the degree of dissimilarity appears inevitable at present and it is left up to a choice of the user.

In conclusion, the clustering solutions that depends on Complete and Ward's 2 algorithms using non-normalized dataset yielded better results with prominent clusters that correspond approximately to the classes known as ground truth compared to those on normalized dataset. Hierarchical cluster analysis on normalized dataset failed because two sub types of leukemia are characterized by complex gene activity of groups of genes that cannot be normalized and treated with equal weights. However, the two subtypes of leukemia can be revealed completely as clusters in a completely unsupervised way.

Activity 2: Clustering genes (Part A)

In this second activity, we consider the problem of clustering genes according to their gene-expression levels across different conditions in a controlled experiment. The goal is to identify genes that show similar expression patterns over a wide range of experimental conditions. This is a relevant problem in bioinformatics as it can, for example, help identify genes that share the same regulatory mechanisms or functions in an organism. In particular, we will use a dataset YeastGalactose from the study in Yeung, Medvedovic, and Bumgarner (2003), which is composed of the gene expression levels of a subset of 205 selected genes of the yeast *Saccharomyces cerevisiae* from 20 different measurements (experimental conditions). The dataset thus contains 205 observations (rows) and 20 variables (columns). It is available in the file *yeast.arff*.

11. Read the dataset directly from the ARFF file into a data frame.

```
  yeast <- read.arff("/Users/Biljana/Data Mining/Ass 2/yeast.arff") # read data from Weka
                          Attribute-Relation File Format (ARFF) files into a dataframe (yeast)
  dim(yeast) # retrieve dimensions of the data frame
## [1] 205  21
```

12. Set aside the rightmost column (containing the class labels) from the data, storing it separately from the remaining data frame (with the 20 predictors).

```
  yeast_classe <- yeast %>% select(-c(1:20)) # allocate the variable "Classe" in a data frame
                                             by dropping a sequence of variables (1:20)

  yeast_predictors <- yeast %>% select(c(1:20)) # storing the remaining sequence of predictor
                                             variables (1:20) in a data frame

  summary(yeast_predictors) # summarize the data set
##       atr1                atr2               atr3                atr4
##  Min.   :-2.740e-01   Min.   :-0.7280   Min.   :-0.61675   Min.   :-0.54025
##  1st Qu.:-2.750e-02   1st Qu.:-0.3365   1st Qu.:-0.25650   1st Qu.:-0.35450
##  Median : 1.225e-02   Median :-0.0695   Median :-0.13075   Median :-0.02650
##  Mean   :-7.317e-05   Mean   :-0.0868   Mean   :-0.07019   Mean   :-0.08093
##  3rd Qu.: 4.850e-02   3rd Qu.: 0.1762   3rd Qu.: 0.06050   3rd Qu.: 0.09750
##  Max.   : 1.628e-01   Max.   : 0.4830   Max.   : 0.88050   Max.   : 0.89900
##       atr5               atr6                atr7               atr8
##  Min.   :-0.6965   Min.   :-0.317000   Min.   :-0.24400   Min.   :-0.52850
##  1st Qu.:-0.3678   1st Qu.:-0.074000   1st Qu.:-0.14950   1st Qu.:-0.35625
##  Median :-0.2700   Median :-0.019500   Median :-0.05775   Median :-0.07825
##  Mean   :-0.1879   Mean   :-0.005391   Mean   :-0.06180   Mean   :-0.09335
##  3rd Qu.:-0.1308   3rd Qu.: 0.051000   3rd Qu.: 0.02500   3rd Qu.: 0.07375
##  Max.   : 0.9970   Max.   : 0.427500   Max.   : 0.22725   Max.   : 0.85800
##       atr9              atr10               atr11              atr12
##  Min.   :-0.55175   Min.   :-0.72800   Min.   :-0.77175   Min.   :-0.38125
##  1st Qu.:-0.37550   1st Qu.:-0.09500   1st Qu.:-0.53625   1st Qu.:-0.17125
##  Median :-0.07050   Median :-0.03325   Median :-0.05275   Median :-0.08850
##  Mean   :-0.09617   Mean   :-0.02264   Mean   :-0.15987   Mean   :-0.03901
##  3rd Qu.: 0.05175   3rd Qu.: 0.05175   3rd Qu.: 0.11400   3rd Qu.: 0.03000
##  Max.   : 1.03525   Max.   : 0.52250   Max.   : 0.79800   Max.   : 0.84650
##      atr13              atr14              atr15              atr16
##  Min.   :-0.71575   Min.   :-0.39925   Min.   :-0.7105   Min.   :-0.43675
##  1st Qu.:-0.49625   1st Qu.:-0.25875   1st Qu.:-0.4768   1st Qu.:-0.21625
##  Median :-0.06025   Median :-0.05500   Median :-0.1643   Median :-0.03500
##  Mean   :-0.13306   Mean   :-0.05991   Mean   :-0.1716   Mean   :-0.03582
##  3rd Qu.: 0.11550   3rd Qu.: 0.04000   3rd Qu.: 0.0665   3rd Qu.: 0.09000
##  Max.   : 0.88700   Max.   : 0.88225   Max.   : 1.0582   Max.   : 0.76750
##      atr17              atr18              atr19              atr20
##  Min.   :-0.50825   Min.   :-0.31100   Min.   :-0.43100   Min.   :-1.0032
##  1st Qu.:-0.34475   1st Qu.:-0.18725   1st Qu.:-0.28450   1st Qu.:-0.6905
##  Median :-0.03950   Median :-0.02725   Median :-0.06575   Median : 0.0175
##  Mean   :-0.09630   Mean   :-0.01968   Mean   :-0.07326   Mean   :-0.1282
##  3rd Qu.: 0.08325   3rd Qu.: 0.05725   3rd Qu.: 0.05850   3rd Qu.: 0.3640
##  Max.   : 0.68550   Max.   : 0.89750   Max.   : 0.81300   Max.   : 0.9960
```

Distribution spread range of the first variable ("atr1") is very small in comparison to the rest of the variables.

13. Use the 205 X 20 data frame to compute a matrix containing all the pairwise Pearson-based dissimilarities between observations, that is, a 205 X 205 matrix with dissimilarities between genes according to their 20 expression measurements. Important Note: It is well known that co-regulated genes are better characterized by similar trends in their gene expression profiles, rather than similar expression levels in terms of their absolute values. In other words, the similarity between genes in terms of their expression profiles for different measurements is better captured by a correlation measure, such as Pearson correlation (James, Witten, Hastie, & Tibshirani, 2013), which is the most widely adopted similarity measure for practical applications of gene clustering. For this reason, in this activity we will use Pearson correlation instead of Euclidean distance. However, recall that Pearson is a similarity measure that ranges from -1(lowest similarity) to +1(highest similarity). After computing the 205 X 205 Pearson similarity matrix, you have to convert it to a dissimilarity matrix whose values range from 0 (lowest dissimilarity) to +1(highest dissimilarity). Once you have this Pearson-based dissimilarity matrix, you can coerce it into type dist as required by the hierarchical clustering methods in the base R package stats.

```
  yeast_predictors_cm <- cor(t(yeast_predictors), method = "pearson") # compute Pearson's
                                                          correlation (similarity) matrix
  yeast_predictors_dd <- as.dist((1 - yeast_predictors_cm)/2) # compute Pearson's based
                            dissimilarity distance matrix as object type "dist" in range 0-1
  head(as.matrix(yeast_predictors_dd)) # display of 6 rows and 6 columns as conventional
                                                                    distance matrix
##           1         2         3           4           5         6
## 1 0.00000000 0.05640425 0.03532003 0.032371276 0.015575199 0.08488286
## 2 0.05640425 0.00000000 0.02759233 0.028914431 0.030166474 0.05537128
## 3 0.03532003 0.02759233 0.00000000 0.020914672 0.018072443 0.06864080
## 4 0.03237128 0.02891443 0.02091467 0.000000000 0.009913172 0.04848900
## 5 0.01557520 0.03016647 0.01807244 0.009913172 0.000000000 0.04274468
## 6 0.08488286 0.05537128 0.06864080 0.048489001 0.042744679 0.00000000
```

Pearson's correlation coefficient has its range in the interval [-1 to +1]. We can transform it to [0, +1] by mappings (Batagelj *et al.*, 1995): (1-r)/2, sqrt(1 - r^2), abs(1-r) and etc. By applying these transformations to dissimilarity distance matrix we expect that the properties of the distance matrix will be preserved.

14. Repeat the clustering analysis in Items 4 to 9 of Activity 1, now using the dissimilarity matrix for the YeastGalactose data computed in Item 13 (and, when applicable, the class labels that you stored separately in Item 12 to label observations as disposed along the horizontal axis of the relevant dendrograms).

```
# Step 4-8 of Activity 1 (see Fig 14, Fig 15, Fig 16 and Fig 17):
   yeast_sl <- hclust(yeast_predictors_dd, method = "single") # hierachical cluster analysis with
                                                          Single-Linkage method
   plot(yeast_sl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "red2", labels = FALSE,
        main = "Single Linkage Method with Correlation Based Distance") # plot the dendrogram
   yeast_al <- hclust(yeast_predictors_dd, method = "average") # hierachical cluster analysis with
                                                          Average-Linkage method
   plot(yeast_al, xlab = "", sub = "", cex = 0.6, hang = -1, col = "slateblue2", labels = FALSE,
        main = "Average Linkage Method with Correlation Based Distance") # plot the dendrogram
   yeast_cl <- hclust(yeast_predictors_dd, method = "complete") # hierachical cluster analysis
                                                          with Complete-Linkage method
   plot(yeast_cl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "turquoise3", labels = FALSE,
       main = "Complete Linkage Method with Correlation Based Distance") # plot the dendrogram
   yeast_wl <- hclust(yeast_predictors_dd, method = "ward.D2") # hierachical cluster analysis with
                                                          Ward's 2 Linkage method
   plot(yeast_wl, xlab = "", sub = "", cex = 0.6, hang = -1, col = "maroon3", labels = FALSE,
       main = "Ward 2 Linkage Method with Correlation Based Distance") # plot the dendrogram
```

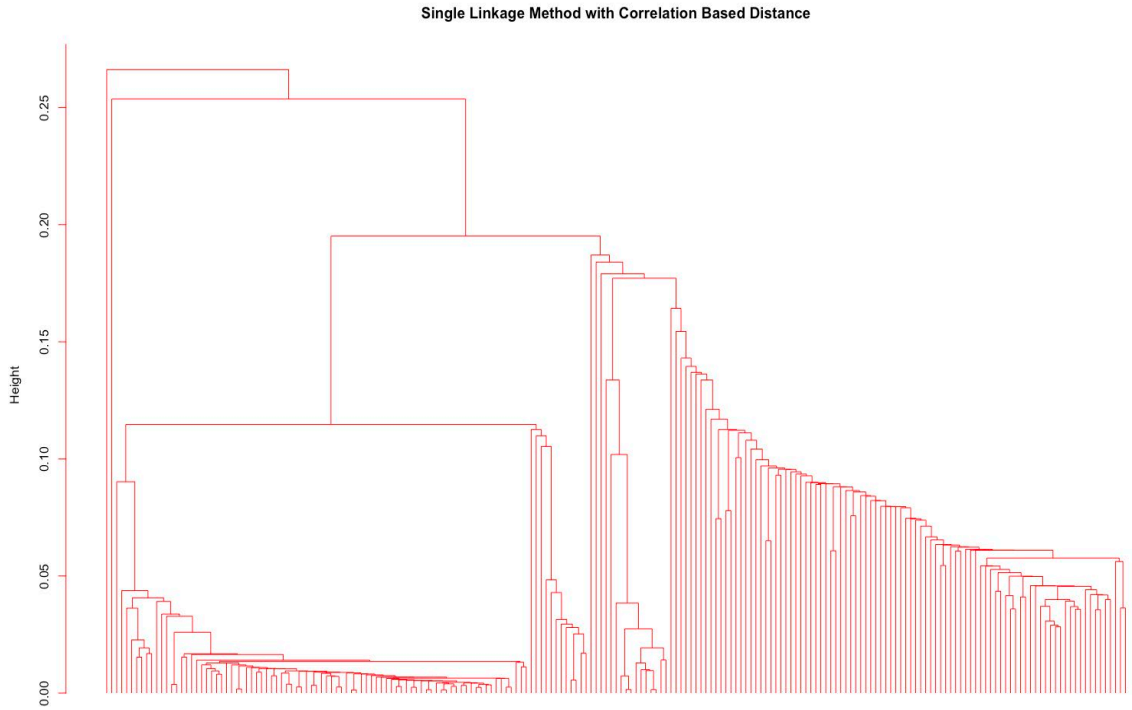

Fig 14: Cluster dendrogram with Single-Linkage algorithm applied on Pearson correlation based distance matrix of the entire dataset “YeastGalactose” without class labels.

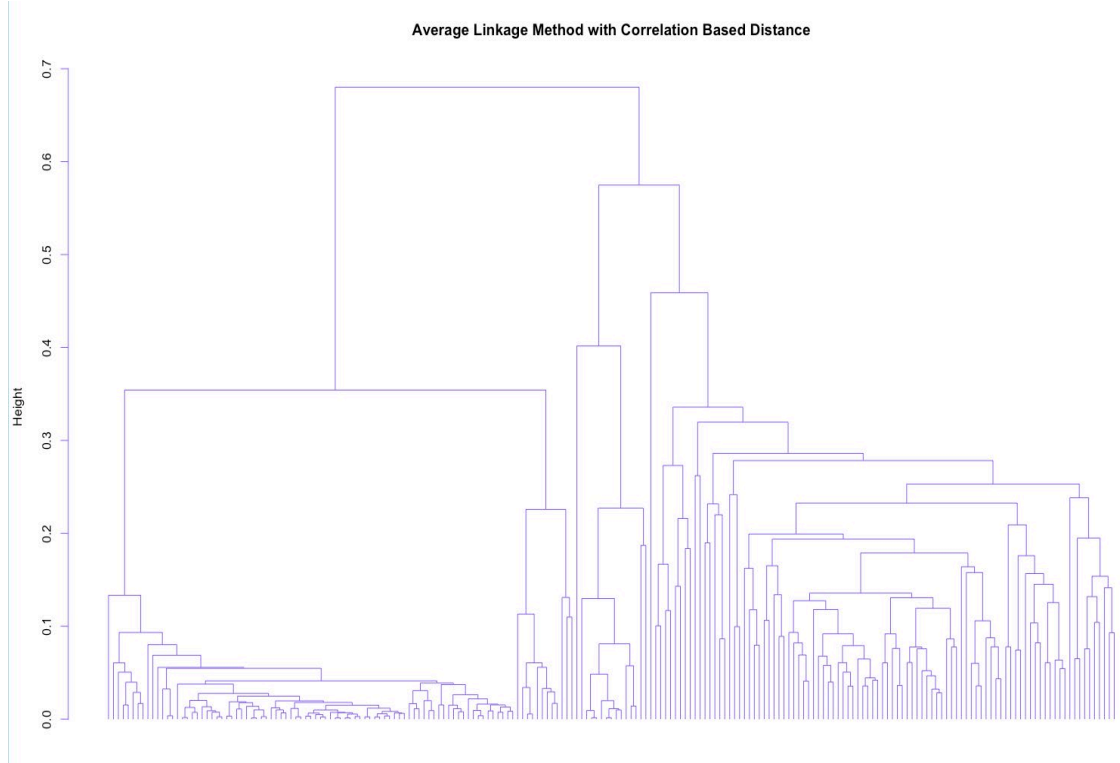

Fig 15: Cluster dendrogram with Average-Linkage algorithm applied on Pearson correlation based distance matrix of the entire dataset “YeastGalactose” without class labels.

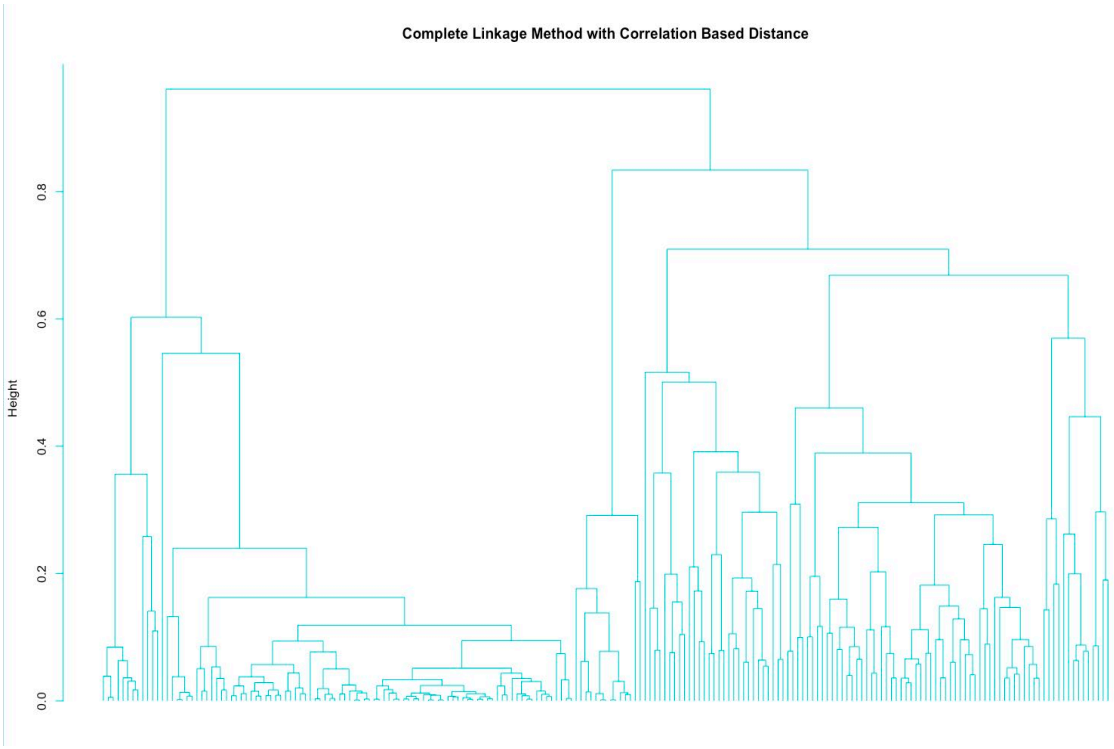

Fig 16: Cluster dendrogram with Complete-Linkage algorithm applied on Pearson correlation based distance matrix of the entire dataset “YeastGalactose” without class labels.

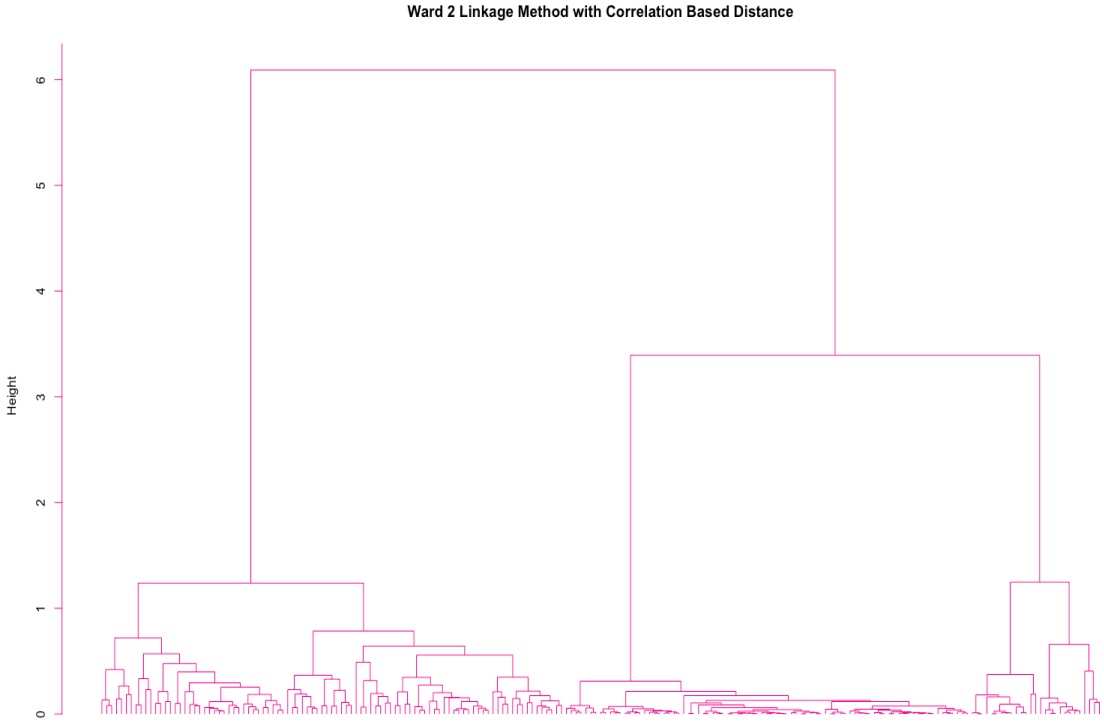

Fig 17: Cluster dendrogram with Ward 2 Linkage algorithm applied on Pearson correlation based distance matrix of the entire dataset “YeastGalactose” without class labels. The aggregation of single objects is dropped at the bottom.

YeastGalactose Dataset Hierarchical Clustering Best Dendrograms:

```
# Step 9 of Activity 1 (see Fig 18 and Fig 19):

# Plot cluster dendrogram with Complete-Linkage algorithm and class labels:
  dendrogram_cl_yeast <- as.dendrogram(yeast_cl) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("black", "blue", "green", "violet"), k = 4)
    colours_to_use_yeast_cl <- as.numeric(yeast_classe$Classe)
    colours_to_use_yeast_cl <- colours_to_use_yeast_cl[order.dendrogram(dendrogram_cl_yeast)]
    labels_colors(dendrogram_cl_yeast) <- colours_to_use_yeast_cl
    dend_list_yeast_cl <- as.character(yeast_classe$Classe)
    labels(dendrogram_cl_yeast) <- dend_list_yeast_cl[order.dendrogram(dendrogram_cl_yeast)]
    plot(dendrogram_cl_yeast, main = "Complete Linkage Method with Correlation Based Distance",
         ylab = "Height")
  dendrogram_cl_rect_yeast <- rect.dendrogram(dendrogram_cl_yeast, k = 4, lty = 5,
                                              lwd = 0, x = 1, col = rgb(0.1, 0.2, 0.4, 0.1))
  legend("topright",
         legend = c("Claster1","Cluster2","Cluster3","Claster4"),
         col = c("black", "red", "green", "blue"),
         title = "Cluster Labels",
         pch = c(20,20), bty = "n", pt.cex = 1.5, cex = 0.8,
         text.col = c("black"), horiz = F, inset = c(0,0.1))


# Plot cluster dendrogram with Ward's 2 Linkage algorithm and class labels:
  dendrogram_wl_yeast <- as.dendrogram(yeast_wl) %>%
    set("branches_lty", 1) %>%
    set("branches_k_color", value = c("green", "black", "blue", "red"), k = 4)
    colours_to_use_yeast_wl <- as.numeric(yeast_classe$Classe)
    colours_to_use_yeast_wl <- colours_to_use_yeast_wl[order.dendrogram(dendrogram_wl_yeast)]
    labels_colors(dendrogram_wl_yeast) <- colours_to_use_yeast_wl
    dend_list_yeast_wl <- as.character(yeast_classe$Classe)
    labels(dendrogram_wl_yeast) <- dend_list_yeast_wl[order.dendrogram(dendrogram_wl_yeast)]
    plot(dendrogram_wl_yeast, main = "Ward 2 Linkage Method with Correlation Based Distance",
         ylab = "Height")
  dendrogram_wl_rect_yeast <- rect.dendrogram(dendrogram_wl_yeast, k = 4, lty = 5, lwd = 0,
                                               x = 1, col = rgb(0.1, 0.2, 0.4, 0.1))
  legend("topright",
         legend = c("Claster1","Cluster2","Cluster3","Claster4"),
         col = c("black", "red", "green", "blue"),
         title = "Cluster Labels",
         pch = c(20,20), bty = "n", pt.cex = 1.5, cex = 0.8,
         text.col = c("black"), horiz = F, inset = c(0,0.1))
```

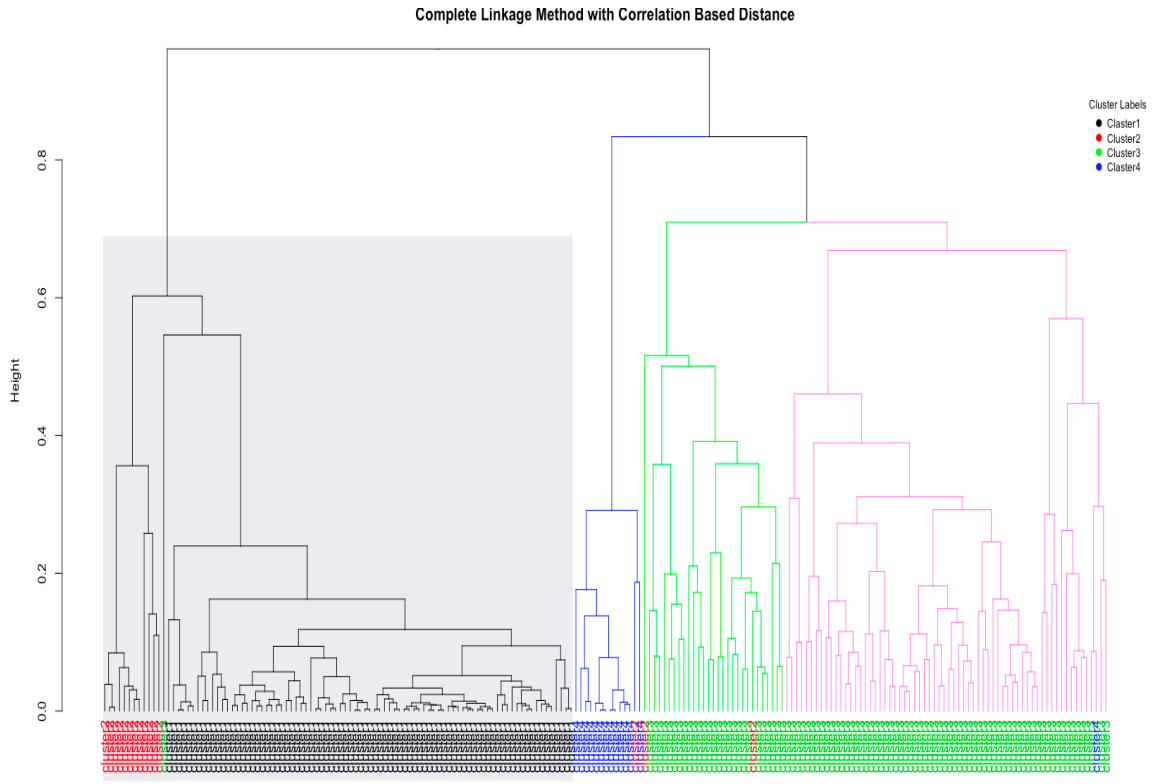

Fig 18: Cluster dendrogram with Complete-Linkage algorithm applied on Pearson correlation based distance matrix of the entire dataset "YeastGalactose" with class labels ("Claster1", "Claster2", "Clsuter3" and "Cluster 4").

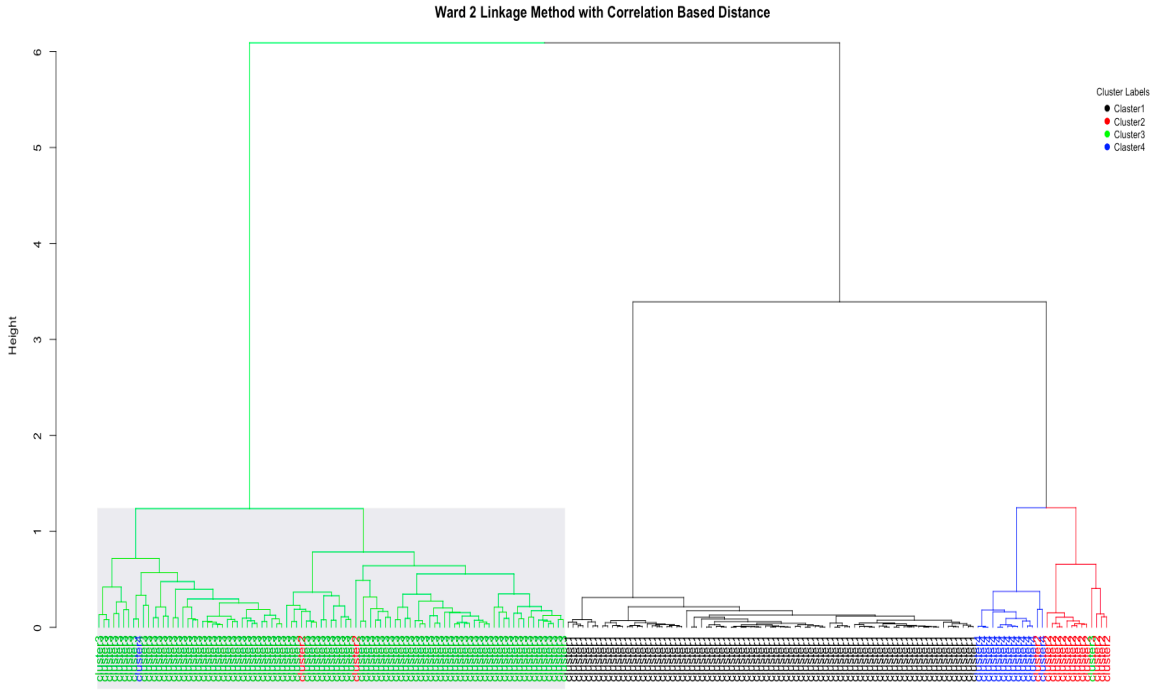

Fig 19: Cluster dendrogram with Ward's 2 algorithm applied on Pearson correlation based distance matrix of the entire dataset "YeastGalactose" with class labels ("Claster1", "Claster2", "Clsuter3" and "Cluster 4").

The clustering solutions that depend on Complete-Linkage and Ward's 2 Linkage methods are preferred the most because they yielded more balanced dendrograms with clearly defined clusters. Both dendrograms contain four clusters as four branches that occur at about the same horizontal distance (Brentari *et al.*, 2016). The outliers are fused rather arbitrarily at higher distance. However, focus is on Ward's 2 dendrogram because it captured the objects more precisely than Complete-Linkage dendrogram. The resulted Ward's 2 dendrogram (Fig 19) suggested to group the objects into four prominent clusters with number of objects within the clusters perfectly equivalent to the number of objects that belong to each of the four classes known, as class labels (("Cluster1", "Cluster2", "Cluster3" and "Cluster 4") according to the ground truth except for five objects. The same schema of differentiation of objects is observed at Complete-Linkage method (Fig 18) with the difference that "Cluster3" is splitting its members into two equally sized clusters, along with fusing the objects from "Cluster2" and "Cluster1" into one larger cluster.

In conclusion, the four functional categories of co-regulated genes in the data can be successfully revealed as clusters in a completely unsupervised way.

In order to interpret the results of clustering properly, we need to perform clustering with different parameters (with/without standardization, with different types of linkage) and to search for patterns that are constantly repeating among the full set of results (James *et al.*, 2013).

In addition, we need to perform clustering on subsets of data to gain sense about robustness of the clusters obtained (Friedman *et al.*, 2001). Finally, these results should not be taken as final truth about the structure of the data set. It is just a beginning for the development of scientific hypothesis and further analysis on independent data set.

Activity 3: Clustering genes (Part B)

Unlike the leukemia data in the first activity, which is very high-dimensional, the YeastGalactose dataset has only moderate dimensionality (20 dimensions), so density-based clustering algorithms may work in this scenario. In this activity we will experiment with the HDBSCAN* algorithm. Important note: One problem with the HDBSCAN* implementation that we are familiar with, available in the package dbscan, is that the version currently available (when this assignment was prepared) says that "Euclidean distance is required" (see ?dbscan::hdbscan). So, although the theoretical HDBSCAN* model works with any distance, in principle we should not run HDBSCAN* directly with Pearson using this package.

Apart from the possible existence of other R implementations of the algorithm that could be used instead, we will stick with the package dbscan here by using a mathematical workaround. Specifically, it can be shown that there is a relation between Pearson similarity and Euclidean distance when the observations are normalised as unit vectors, that is, when the rows of the data matrix are rescaled so that each row is a vector with magnitude one (i.e., length = 1). Clustering the normalised data with Euclidean distance is expected to provide results that are similar to those that would be obtained by clustering the original data with Pearson similarity.

15. Rescale the data frame in a row-wise fashion so that each rescaled row has magnitude 1. You can achieve this by dividing each element of a row by the magnitude of the row.

According to summary results of the dataset, we noticed that variable 1 ("atr 1") has smaller range than the rest of the variables. If we apply Euclidean distance formula directly, it might be dwarfed by other variables that have larger range of distribution. Therefore, we normalize all the variables to lie between 0 and 1 with min-max normalization where x is the actual value of the variable and minimum and maximum are taken over all instances in the dataset (Witten and Frank, 2005).

```r
# Apply min-max normalization to all variables:
  normalize <- function(x) {
    return((x - min(x)) / (max(x) - min(x)))
  }

  yeast_predictors_norm = t(apply(yeast_predictors, 1, normalize)) # apply the function
                                                                    to the data frame
# Apply the Euclidean distance normalization so that each row is a unit vector with magnitude one:
  magnitude_unit = function(x){
    x/sqrt(sum(x^2))
  }
  yeast_predictors_scaled = t(apply(yeast_predictors_norm, 1, magnitude_unit)) # apply the
                                       function to the previously rescaled dataset

  sqrt(sum(yeast_predictors_scaled[100,]^2)) == 1 # check the function with random chosen rows
## [1] TRUE
  sqrt(sum(yeast_predictors_scaled[180,]^2)) == 1 # check the function with random chosen rows
## [1] TRUE
  sqrt(sum(yeast_predictors_scaled[90,]^2)) == 1 # check the function with random chosen rows
## [1] TRUE

# Another way of checking to see if all rows add up to 1:
  result <- sqrt(rowSums(yeast_predictors_scaled^2))
   if (mean(result) == 1) {
    print("All rows have a magnitude of 1.")
  } else {
    print("Error! Rows are not standardised.")
  }
## [1] "All rows have a magnitude of 1."
```

16. Run HDBSCAN* (with Euclidean distance) on the rescaled version of the data frame obtained in Item 15. You can (optionally) try different values for the parameter MinPts, but MinPts = 5 is required. Plot the resulting HDBSCAN* dendrograms with and without the class labels along the horizontal axis, just like in Items 4–9 (Activity 1) and Item 14 (Activity 2).

```r
  yeast_predictors_hdbs <- hdbscan(yeast_predictors_scaled, minPts = 5) # using single parameter
                                       minPts = 5, HDBSCAN finds 4 clusters and 24 noise points
  yeast_predictors_hdbs # print results of HDBSCAN algorithm
## HDBSCAN clustering for 205 objects.
## Parameters: minPts = 5
## The clustering contains 4 cluster(s) and 24 noise points.
##
##  0  1  2  3  4
## 24 80 12  8 81
##
## Available fields: cluster, minPts, cluster_scores, membership_prob,
##                   outlier_scores, hc


# Condense the complicated cluster hierarchy into the simplified cluster tree that shows
  cluster-wide changes over an infinite number of eps thresholds:
  plot(yeast_predictors_hdbs, gradient = c("purple", "blue", "green", "yellow"), show_flat = T)
```

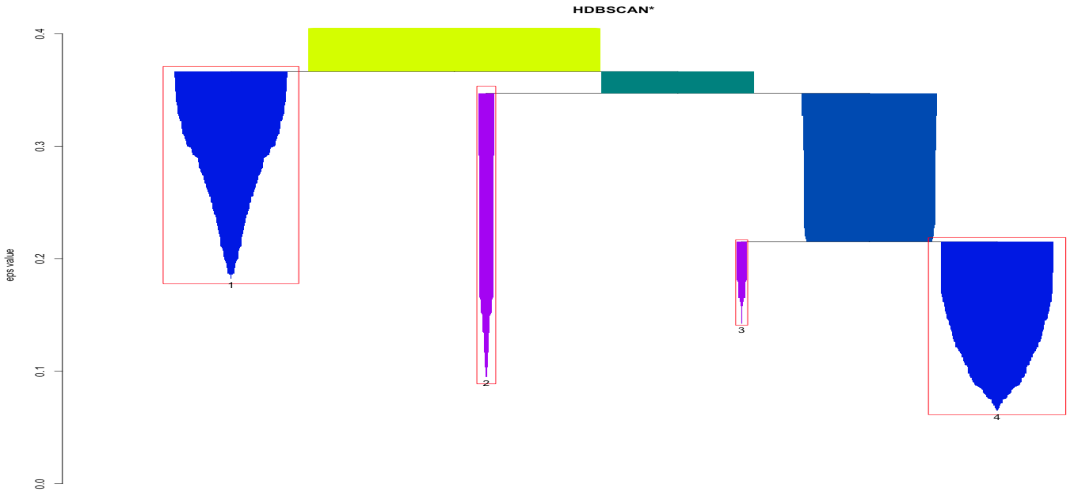

Fig 20: HDBSCAN* simplified cluster tree for the YeastGalactose dataset (*minPts* = 5). There are 4 most prominent clusters at this stage.

```
color_1to8 <- function(x) ifelse(x == 0,1,((x - 1) %% 7) + 2) # function to apply colors to
                                                               clusters

plot(yeast_predictors_scaled, pch = 19, col = color_1to8(yeast_predictors_hdbs$cluster),
     main = "HDBSCAN* Data Set (minpts = 5): 4 clusters", xlab = "x", ylab = "y")
legend("topright",
       legend = c("Outlier","1","2","3","4"),
       col = c("black", "red", "green", "blue", "cyan"),
       pch = c(20,20), pt.cex = 1.5, cex = 0.8,
       text.col = c("black"), horiz = F, inset = c(0.05,0.1)) # plot the extracted partition
                                                               for the datset in Fig 20
```

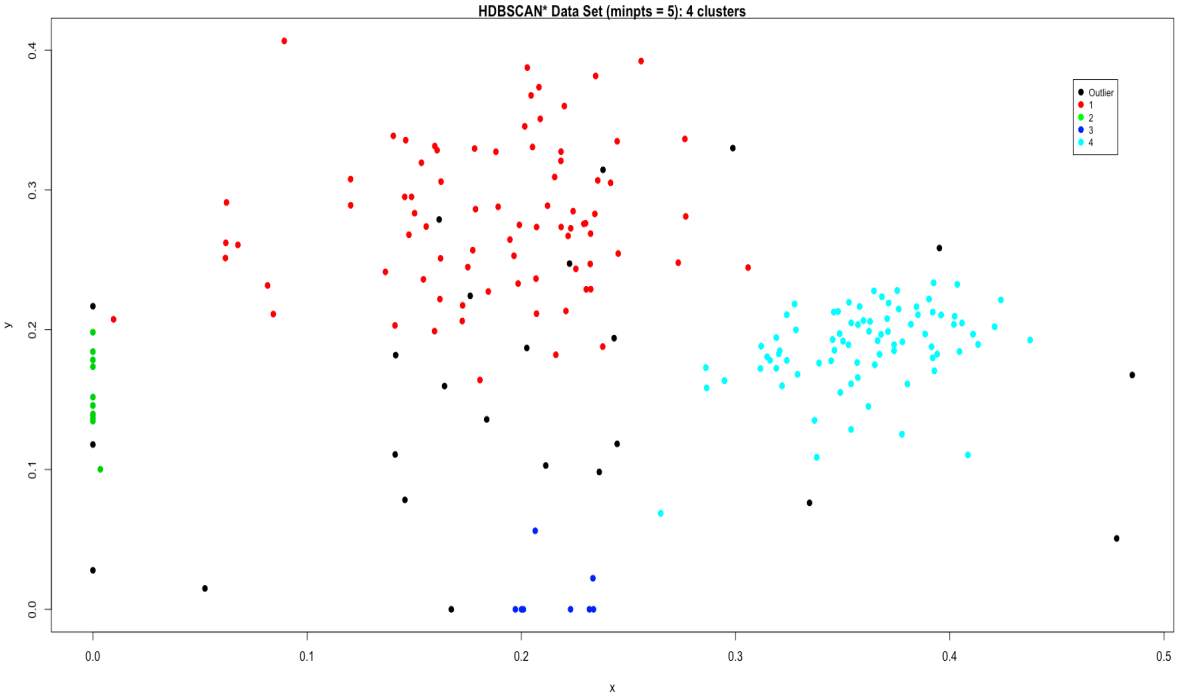

Fig 21: Extracted partition for the dataset in Fig 20 with clusters in colors and outliers (noise) in black.

A new outlier measure is developed that computes an outlier score of each point in the data based on local and global properties of the hierarchy, defined as the Global-Local Outlier Score from Hierarchies (GLOSH), (Campello *et al.*, 2015). According to HDBSCAN hierarchical analysis, points that are not assigned to any clusters are considers as outliers and the outlier scores computed are not just the inversely-proportional scores to the membership probabilities. Unlike the membership probabilities, the opacity of the point represents the amount of "outlierness".

```
# Presenting the labeled 24 outliers(black points) as data points and colored clusters:
 outliers <- order(yeast_predictors_hdbs$outlier_scores, decreasing = T)[1:24]
 colors <- mapply(function(col, i) adjustcolor
                 (col, alpha.f = yeast_predictors_hdbs$outlier_scores[i]),
                 palette()[yeast_predictors_hdbs$cluster + 1],
                 seq_along(yeast_predictors_hdbs$cluster))
  plot(yeast_predictors_scaled, col = colors, pch = 19, main = "HDBSCAN* Data Set (minpts = 5): 4
      clusters with Labelled Outliers", xlab = "x", ylab = "y")
  text(yeast_predictors_scaled[outliers, ], labels = outliers, pos = 3)
  legend("topright",
         legend = c("Outlier","1","2","3","4"),
         col = c("black", "red", "green", "darkviolet", "cyan"),
         pch = c(20,20), pt.cex = 1.5, cex = 0.8,
         text.col = c("black"), horiz = F, inset = c(0.05,0.1))
```

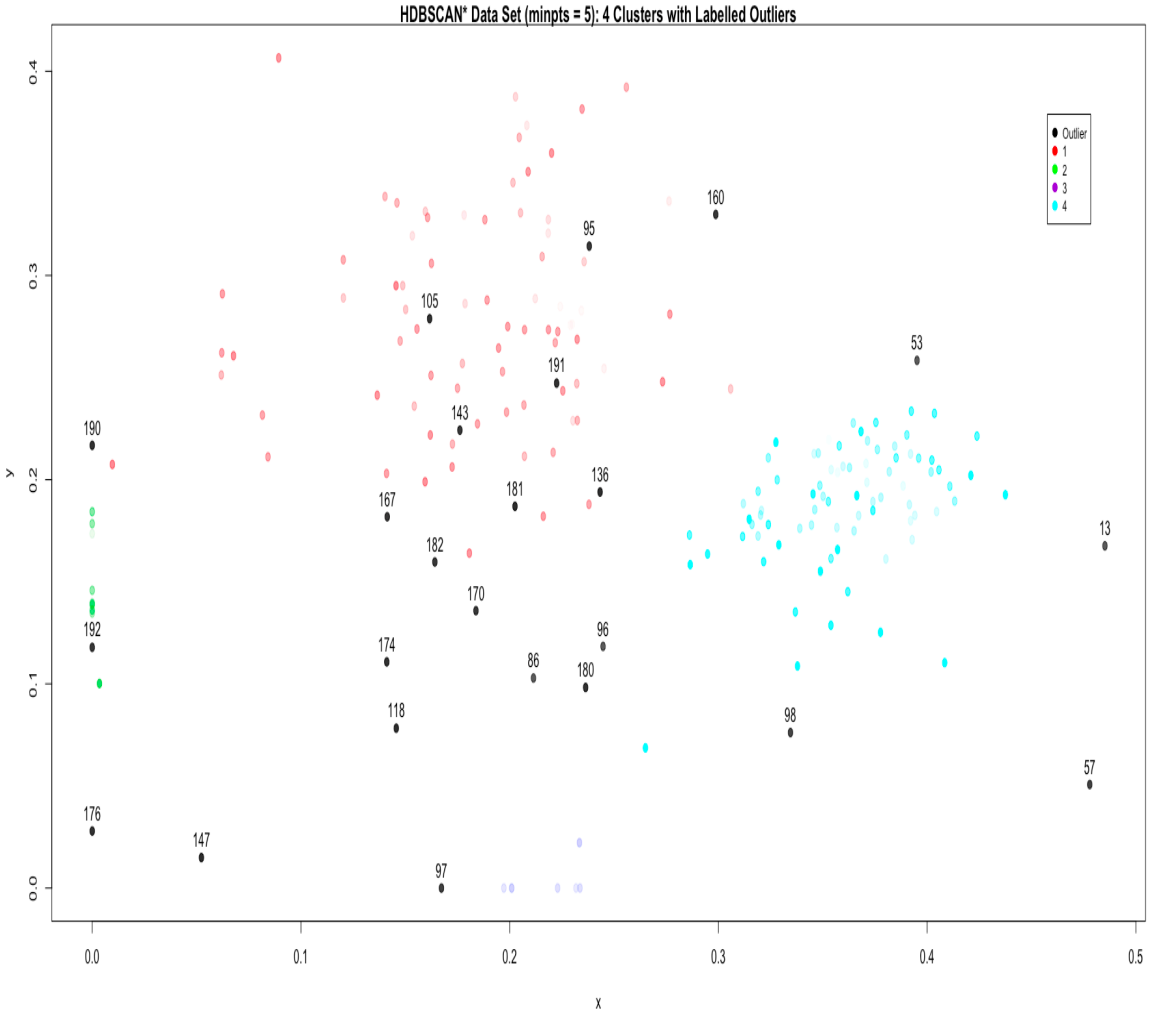

Fig 22: Extracted partition for the dataset in Fig 20 with clusters in colors and labeled outliers (noise) in black.

```r
# Plot HDBSCAN* dendrograms with and without class labels (see Fig 23 and Fig 24):
  plot(yeast_predictors_hdbs$hc, main = "HDBSCAN* Hierarchy", xlab = "", sub = "", hang = -1,
       col = "darkorchid3", labels = FALSE, cex = 0.6) # dendogram without class labels

  dend_hdbs <- as.dendrogram(yeast_predictors_hdbs$hc)
    colours_to_use_hdbs <- as.numeric(yeast_classe$Classe)
    colours_to_use_hdbs <- colours_to_use_hdbs[order.dendrogram(dend_hdbs)]
    labels_colors(dend_hdbs) <- colours_to_use_hdbs
    dend_list_hdbs <- as.character(yeast_classe$Classe)
    labels(dend_hdbs) <- dend_list_hdbs[order.dendrogram(dend_hdbs)]
  plot(dend_hdbs, main = "HDBSCAN* Hierarchical Clustering", ylab = "Height")
  dendrogram_hdbs_rect_yeast_scaled <- rect.dendrogram(dend_hdbs, k = 4, lty = 5, lwd = 0,
                                                       x = 1, col = rgb(0.1, 0.2, 0.4, 0.1))
  legend("topright",
         legend = c("Claster1","Cluster2","Cluster3","Claster4"),
         col = c("black", "red", "green", "blue"),
         title = "Cluster Labels",
         pch = c(20,20), bty = "n", pt.cex = 1.5, cex = 0.8,
         text.col = c("black"), horiz = F, inset = c(0,0.1)) # dendogram with class labels
```

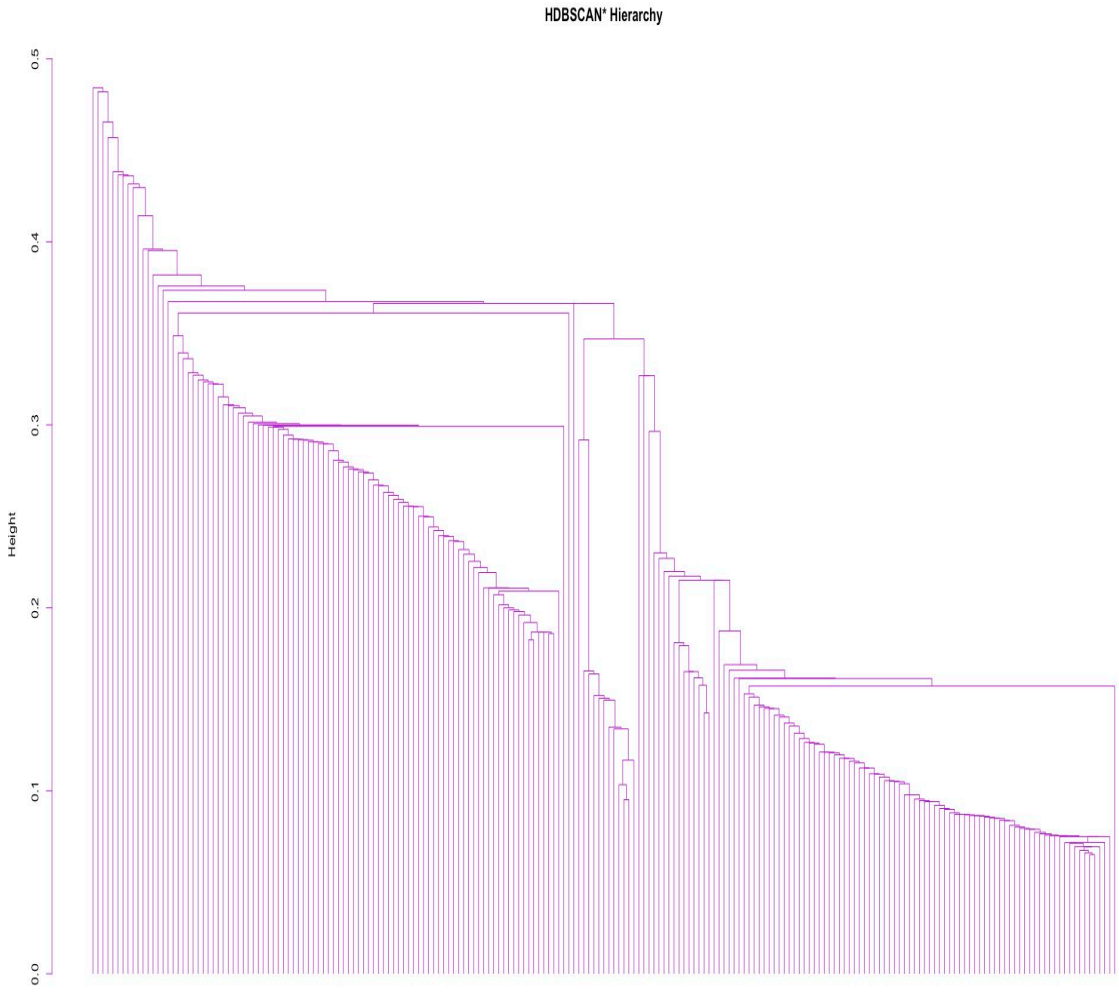

Fig 23: HDBSCAN* traditional dendrogram without class labels for the YeastGalactose dataset *(minPts = 5)*.

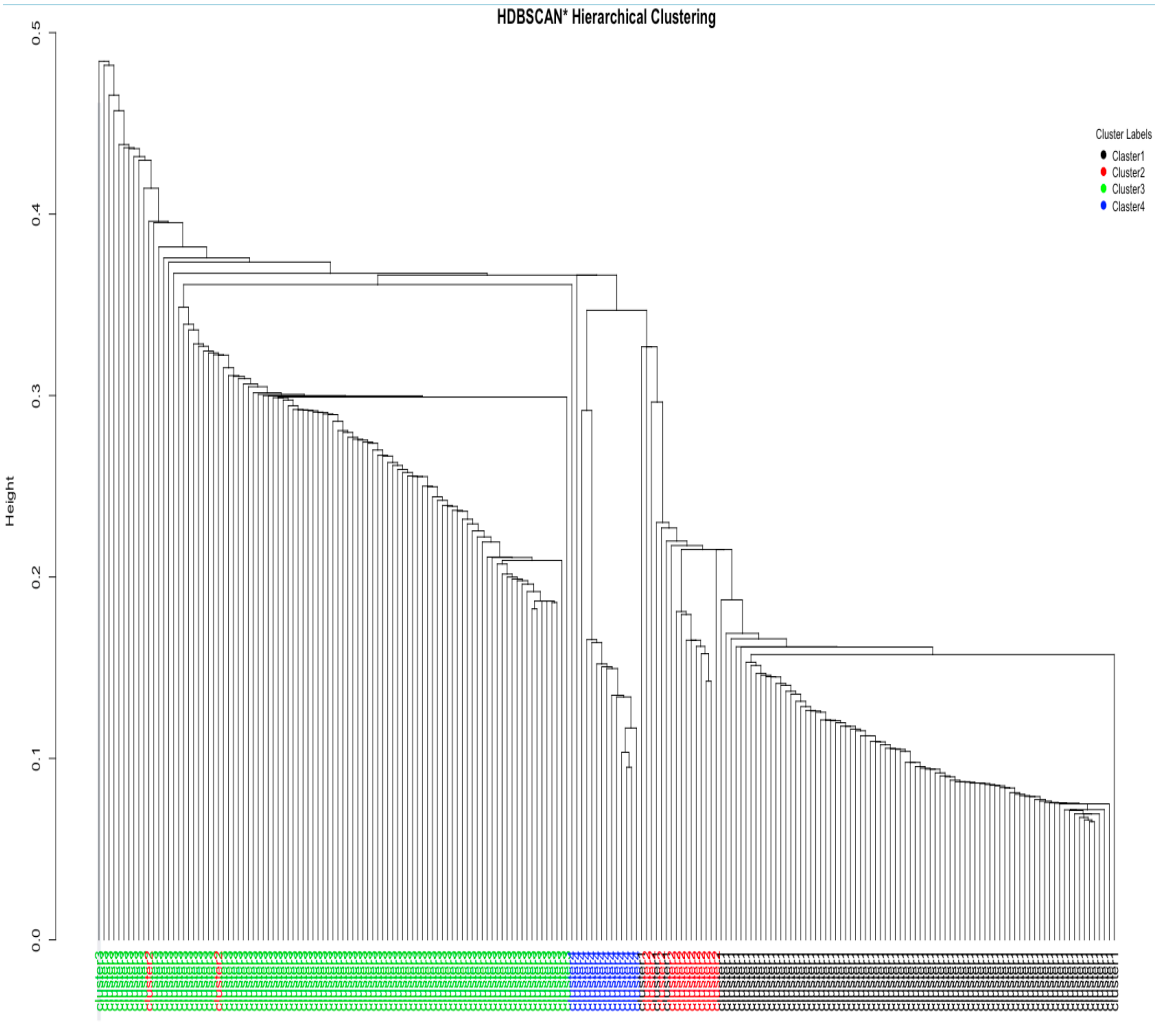

Fig 24: HDBSCAN* traditional dendrogram with class labels for the YeastGalactose dataset *(minPts = 5)*.

We can notice that objects belonging to each of the four classes known, as class labels according to the ground truth and that have not been used during HDBSCAN algorithm performance are perfectly captured as four prominent groups in the dendrogram. In addition, we can observe that classes ("cluster3" and "cluster4") are apparently closer to each other and might be considered as subgroups of a bigger group/cluster. The same observation can be applied to classes ("cluster1" and "cluster2"). At the initial stages of exploratory data analysis, this type of information can be undeniably treasured when the information about class labels is nonexistent at all (Campello *et al.*, 2013a).

HDBSCAN* is a state-of-the-art density-based clustering algorithm (Campello, Moulavi, & Sander, 2013), which has also been extended as a complete framework for unsupervised and semi-supervised data clustering, outlier detection and advanced visualisations (Campello, Moulavi, Zimek, & Sander, 2013). Hierarchical DBSCAN* or HDBSCAN* generates a complete clustering hierarchy composed of all possible density-based clusters for an infinite range of density thresholds by using Hartigan's concept of rigid clusters (Hartigan 1975).

Data Standardization

Relation between Pearson Similarity and Euclidean Distance

According to the literature (Kassambara, 2017), standardization makes the distance measure methods: Correlation and Euclidean, others as well, more similar that they would be with non-transformed data. When the data are standardized, there is a functional relationship between Pearson correlation coefficient r(x, y) and Euclidean distance defined as following:

$$d_{euc}(x,y) = \sqrt{2m[1 - r(x,y)]}$$

where x and y are two standardized vectors with zero mean and unit length. Thus, the results obtained from Pearson correlation measures and standardized Euclidean distances are comparable.

As a matter of interest, Pearson dissimilarity distant matrix is computed on standardized YeastGalactose dataset with z-score transformation and resulted dendrogram is compared with the obtained dendrogram on non-transformed original dataset from Activity 2 (Question 13) and HDBSCAN dendrogram (Activity 3, question 16).

```
# Dendrogram with Single-Linkage method on normalized dataset with z-score transformation before
  Pearson dissimilarity matrix computation is created (see Fig 25):
  yeast_predictors_scaled <- scale(yeast_predictors) # normalize the data set with mean 0 and
                                                       standard deviation 1
  yeast_predictors_cm <- cor(t(yeast_predictors_scaled), method = "pearson") # compute Pearson's
                                                                               correlation matrix (similarity matrix)
  yeast_predictors_dd <- as.dist((1 - yeast_predictors_cm)/2) # compute Pearson's based
                                                               dissimilarity distance matrix as object type "dist"
  yeast_sl_norm <- hclust(yeast_predictors_dd, method = "single") # hierachical cluster analysis
                                                                    with Single-Linkage method
  plot(yeast_sl_norm, xlab = "", sub = "", cex = 0.6, hang = -1, col = "red2", labels = FALSE,
       main = "Single Linkage Method with Correlation Based Distance on Normalized Dataset") #
                                                                                              plot the dendrogram
```

Comparison of the observed figures (Fig 25) shows that there is a relation between Pearson similarity and Euclidean distance when the observations are normalized as unit vectors, that is, when the rows of the data matrix are rescaled so that each row is a vector with magnitude one (i.e., length = 1). Clustering the normalized data with Euclidean distance provide results that are similar to those that are obtained by clustering the original data with Pearson similarity. In addition, the provided results about clustering the normalized data with Euclidean distance are even more similar to the results obtained by clustering the transformed original data with z-score normalization before Pearson dissimilarity matrix calculations are performed.

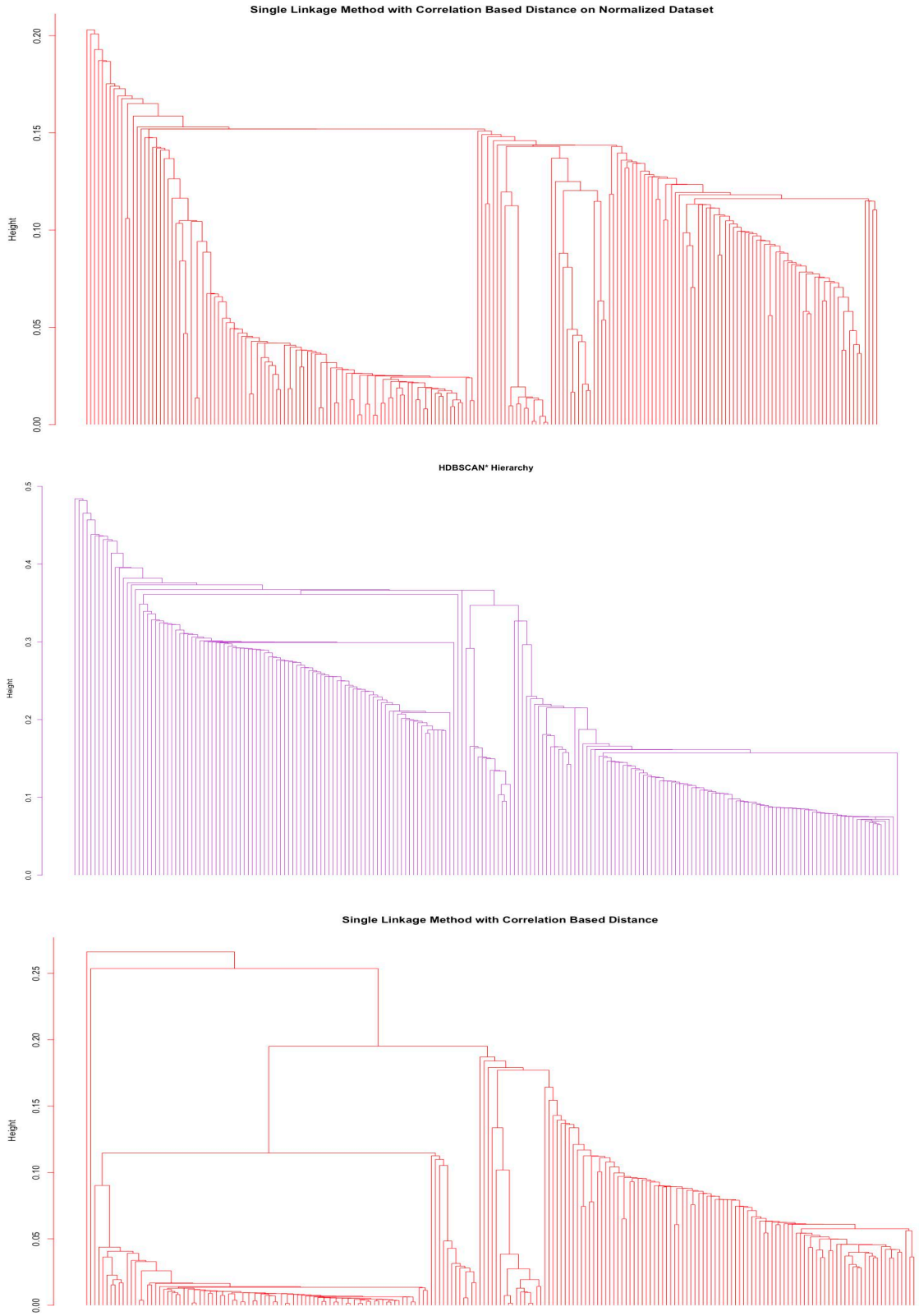

Fig 25: Comparison of cluster dendrograms with Single-Linkage algorithm applied on Pearson correlation based distance matrix on: a) normalized dataset, c) non-normalized dataset and b) HDBSCAN* algorithm (*minPts = 5)* on normalized with Euclidean distance YeastGalactose dataset.

17. Plot a contingency table. By setting MinPts = 5, the automatic cluster extraction method provided by HDBSCAN* extracts four clusters from the resulting hierarchy. Plot a contingency table of these clusters (labelled '0', '1', '2', '3' and '4', where '0' means objects left unclustered as noise/outliers) against the ground truth class labels that you stored separately in Item 12 (a factor with levels 'cluster1', 'cluster2','cluster3', 'cluster4').

```
  yeast_table <- table(yeast_predictors_hdbs$cluster, labels = yeast_classe$Classe) # plot a
                                                                                    contingency table
  yeast_table
## labels
##     cluster1 cluster2 cluster3 cluster4
##   0        3        5       15        1
##   1        0        1       78        1
##   2        0        0        0       12
##   3        0        8        0        0
##   4       80        1        0        0
```

Fig 26: Contingency table results

18. Interpret the contingency table. In particular:
(a) What is the best correspondence between the four found clusters and the clusters according to the ground truth, that is, the best association between cluster labels '1', '2', '3' and '4' as named by HDBSCAN* and the four known functional categories 'cluster1', 'cluster2', 'cluster3' and 'cluster4' as named in the ground truth?

According to the contingency table results:

- 80 objects identified as members of "cluster1" according to the ground truth class labels are captured as prominent group of cluster labeled as "4" except 3 objects that are detected as outliers;
- Majority of objects (8 objects) that belong to "cluster2" are captured as prominent group of cluster labeled as "3"; 1 objects became members of cluster "1" and another object became member of cluster "4", while 5 objects are detected as outliers;
- 78 objects that belong to "cluster3" became members of cluster labeled as "1", remaining 15 objects are detected as outliers;
- 12 objects of "cluster4" became members of cluster labeled as "2", 1 object became part of cluster labeled as "1" and 1 object is identified as an outlier;
- Overall 24 objects are identified as outliers;

(b) What is the functional category for which most genes have been labeled as noise/outliers?

Functional category for which most genes have been labeled as outliers is "cluster3" with 15 detected outliers that is 16.13% of its data but 62.5% overall. Functional category "cluster2" has 5 outliers but that is 33.33% of its data. Following is "cluster4" with 8.37% of outliers and a "cluster1" with 3.75% of outliers of its own data.

19. Plot the genes grouped by their class labels (that is, functional categories 'cluster1', 'cluster2', 'cluster3'and 'cluster4'), in such a way that all the genes belonging to the same class are plotted in a separate sub-figure (four sub-figures in total, each one in a different colour). Plot each gene as a time-series with 20 data points (where each point is connected by lines to its adjacent points in the series).

```
  yeast_plot <- cbind(yeast_predictors, cluster = yeast_classe$Classe) # bind the Classe variable
                                                          to the original yeast data frame
  yeast_plot_final <- cbind(yeast_plot, row_number = seq(1, nrow(yeast_plot)))
                                                        # bind row_number column to the data frame
  yeast_plot_melt <- melt(yeast_plot_final, id.vars = c("row_number", "cluster" ))
              # convert the variables into a molten data frame with 4 columns and 2 id.variables

  colnames(yeast_plot_melt) <- c("Row_number", "Cluster", "Parameter", "Expression") # define
                                                           the column names of the 4 variables
# Assign the variables as vectors:
  Cluster <- yeast_plot_melt$Cluster
  Parameter <- yeast_plot_melt$Parameter
  Expression <- yeast_plot_melt$Expression
  Row_number <- yeast_plot_melt$Row_number

# Plot the ggplot2 of gene expressions levels of subset of 205 selected genes of S. cerevisiae
from 20 different experimental conditions as time_series: time series of expression of 4 clusters
of genes (see Fig 27):
  ggplot(yeast_plot_melt, aes(x = Parameter, y = Expression, colour = factor(Cluster))) +
    geom_line(aes(group = Row_number)) + facet_grid(Cluster~., space = "free") +
     ggtitle("Genes Expression Patterns Grouped in Four Functional Categories")
```

20. Plot a figure analogous to the one in Item 19, but now with genes grouped in separate sub-figures according to their cluster as assigned by HDBSCAN* (‘1’, ‘2’, ‘3’ and ‘4’), rather than by class labels. Do not plot genes that were left unclustered as noise by HDBSCAN* (labelled ‘0’). Use the best class-tocluster association, as in your answer to Item 18, in order to assign each sub-figure of a cluster the same colour used in the sub-figure of the corresponding class in Item 19. For instance, supposing that the best association of class ‘clusterX’ in the ground truth is with HDBSCAN* cluster ‘Y’, according to the contingency table in Item 18, then if the genes belonging to class ‘clusterX’ have been plotted in red in Item 19, then the genes belonging to HDBSCAN* cluster ‘Y’ should also be plotted in red.

```
  yeast_plot <- cbind(yeast_predictors, cluster = yeast_predictors_hdbs$cluster) # adding the
                                                                  clusters from HDBSCAN* solution
  yeast_plot_final <- cbind(yeast_plot, row_number = seq(1, nrow(yeast_plot)))
                                                        # bind row_number column to the data frame
  yeast_plot_melt <- melt(yeast_plot_final, id.vars = c("row_number", "cluster" ))
              # convert the variables into a molten data frame with 4 columns and 2 id.variables

  colnames(yeast_plot_melt) <- c("Row_number", "HDBS_Cluster", "Parameter", "Expression")
                                  # define the column names of the 4 variables

  HDBS_Cluster <- yeast_plot_melt$HDBS_Cluster # assign the variable as a vectors
  sub_yeast_plot <- yeast_plot_melt %>% filter(HDBS_Cluster > 0 ) %>% droplevels() # remove
                                                                        the cluster"0"

   sub_yeast_plot$HDBS_Cluster = factor(sub_yeast_plot$HDBS_Cluster, ordered = F,
                                        levels = c(4,3,1,2)) # re-ordering the factor levels of
                                     HDBSCAN* solution according to the ground truth class labels
# Assign the variables as vectors:
  HDBS_Cluster <- sub_yeast_plot$ HDBS_Cluster
  Parameter <- sub_yeast_plot$Parameter
  Expression <- sub_yeast_plot$Expression
  Row_number <- sub_yeast_plot$Row_number

# Plot the ggplot2 of gene expressions levels of subset of 205 selected genes of S. cerevisiae
from 20 different experimental conditions as time_series: time series of expression of 4 clusters
of genes (see Fig 27):
ggplot(sub_yeast_plot, aes(x = Parameter, y = Expression, colour = factor(HDBS_Cluster))) +
    geom_line(aes(group = Row_number)) + facet_grid(HDBS_Cluster~., space = "free") +
       ggtitle("Genes Expression Patterns in YeastGalactose Dataset by HDBSCAN* Algorithm”,
               subtitle = "HDBSCAN* generated clusters arranged according to the order of ground
                           truth class labels\n where 4 = cluster1, 3 = cluster2, 1 = cluster3,
                           2 = cluster4")
```

Fig 27: Clustering of gene expression activity of 205 selected genes of *Saccharomyces cerevisiae* into four clusters according to the ground truth class labels.

Fig 28: HDBSCAN* clustering of gene expression activity of 205 selected genes of *Saccharomyces cerevisiae* grouped into four clusters without ground truth class labels but arranged according to the order of ground truth class labels as per Figure 27.

21. Compare the pairs of sub-figures with the same colour in the plots of Items 19 and 20. In particular:
(a) Visually, do the genes in each cluster found by HDBSCAN* (that is, each sub-figure in Item 20) correspond reasonably well to the associated functional category in the ground truth (that is, the corresponding sub-figure in Item 19)?

Definitely, all the genes in each cluster found by HDBSCAN* (that is, each sub-figure in Fig 28) correspond reasonably well to the associated functional category in the ground truth (that is, the corresponding sub-figure in Fig 27). It is shown that cluster hierarchy can be processed in different way but still yielded much better and more robust results than other state-of-the-art methods.

(b) Look at the contingency table for the functional categories that have had genes left unclustered as noise. Now look at the corresponding pairs of sub-figures in Items 19 and 20, noticing that these genes are plotted in Item 19 but not in Item 20. Does the removal of these genes make the most prominent pattern in each cluster visually more clear (which would indicate that those genes labelled as noise by HDBSCAN* are indeed outliers)?

According to the contingency table results, there are detected 24 outliers overall that are 11.7 % of the dataset, when min-max normalization along with Euclidean distance row-wise normalization is applied before HDBSCAN* hierarchy is constructed.

It is evident that removal of outliers makes the most prominent pattern in each cluster visually clearer, especially in smaller clusters, such as “cluster2” and “cluster4” labels according to the ground truth. Outlier removal has a significant impact on smaller clusters because removing a small amount of observations from the cluster can cause undeniable change of its shape. The most affected by removing of outliers is class label (“cluster2”) or HDBSCAN* cluster 3 because 33.33% of its data are removed that make its most characteristic pattern of gene expression visually transparent, and indicative of sufficient variations between individual gene activities that were masked by the outliers. This effect is also visible in “cluster4” or HDBSCAN* cluster 2 gene expression patterns as the gaps between atr6 and atr7 became more prominent after removal of one outlier.

Removal of outliers at “cluster1” is not significant because outliers occupied 3.75% of its data. We may assume that the genetic activity of “Cluster3” is at steady levels under different experimental conditions because it did not demonstrate any significant change in the pattern of gene expression even though outliers represent 16.13% of its data.

REFERENCES

Abrahamowicz, M. (1985, July). The use of non-numerical a priori information for measuring dissimilarities. In *Fourth European Meeting of the Psychometric Society and the Classification Societies, July* (Vol. 25).

Batagelj, V., & Bren, M. (1995). Comparing resemblance measures. *Journal of classification*, *12*(1), 73-90.

Brentari, E., Dancelli, L., & Manisera, M. (2016). Clustering ranking data in market segmentation: a case study on the Italian McDonald's customers’ preferences. *Journal of Applied Statistics*, *43*(11), 1959-1976.

Campello, R. J., Moulavi, D., & Sander, J. (2013, April). Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 160-172). Springer, Berlin, Heidelberg.

Campello, R. J., Moulavi, D., Zimek, A., & Sander, J. (2013). A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies. *Data Mining and Knowledge Discovery*, *27*(3), 344-371.

Campello, R. J., Moulavi, D., Zimek, A., & Sander, J. (2015). Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *10*(1), 1-51.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1, No. 10). New York: Springer series in statistics.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., ... Lander, E. S. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, *286*(5439), 531–537.

James, G., Witten, D., Hastie T., & Tibshirani, R. (2013). *An introduction to statistical learning with applications in R*. New York, NY: Springer.

Hartigan, J. A. (1975). *Clustering algorithms*. John Wiley & Sons, Inc.

Murtagh, F. (1985). Multidimensional clustering algorithms. *Compstat Lectures, Vienna: Physika Verlag, 1985*.

Murtagh, F., & Legendre, P. (2014). Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion?. *Journal of classification*, *31*(3), 274-295.

Odong, T. L., Van Heerwaarden, J., Jansen, J., van Hintum, T. J., & Van Eeuwijk, F. A. (2011). Determination of genetic structure of germplasm collections: are traditional hierarchical clustering methods appropriate for molecular marker data? *Theoretical and applied genetics*, *123*(2), 195-205.

Kassambara, A. (2017). *Practical guide to cluster analysis in R: Unsupervised machine learning* (Vol. 1). STHDA.

Rousseeuw, P. J., & Kaufman, L. (1990). Finding groups in data. *Hoboken: Wiley Online Library*.

Sarstedt, M., & Mooi, E. (2014). A concise guide to market research. *The Process, Data, and*, *12*.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, *58*(301), 236-244.

Witten, I. H., Frank, E., & Hall, M. A. (2005). Practical machine learning tools and techniques. *Morgan Kaufmann*, 578.

Yeung, K. Y., Medvedovic, M., & Bumgarner, R. E. (2003). Clustering gene-expression data with repeated measurements. *Genome Biology*, *4*(5), R34.