Homework_3

Alberto De Benedittis

19/5/2021

LIBRARIES USED

```
library('ggcorrplot')
```

```
## Warning: package 'ggcorrplot' was built under R version 4.0.5
```

```
## Loading required package: ggplot2
```

```
library("ggplot2")
library("ggdendro")
```

```
## Warning: package 'ggdendro' was built under R version 4.0.5
```

```
library("ape")
```

```
## Warning: package 'ape' was built under R version 4.0.5
```

```
library("FactoMineR")
```

```
## Warning: package 'FactoMineR' was built under R version 4.0.4
```

```
library("factoextra")
```

```
## Warning: package 'factoextra' was built under R version 4.0.5
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl
/ve3WBa
```

```
library('mclust')
```

```
## Warning: package 'mclust' was built under R version 4.0.5
```

```
## Package 'mclust' version 5.4.7
## Type 'citation("mclust")' for citing this R package in publications.
```

EXERCISE

This homework deals with PCA and clustering. You should submit an RMarkdown file and a pdf file of the report. The RMarkdown file should reproduce exactly the pdf file that you will submit. The pdf file should be rendered directly from the RMarkdown (e.g. output: pdf_document) and not converted from any other output format. Note that: • your code should run without errors (except for minor adjustments such as file paths); • you should discuss/justify each choice that you make and provide comments on the results that you obtain.

You will be working on a gene expression data set of 128 diseased patients belonging to two subtypes. The data are provided in the attached gene_expr.tsv file, containing expression for 12,625 genes and an additional column with patient subtypes. You will perform an unsupervised analysis, where you will assume no knowledge about these subtypes, so that you can use this information at the validation stage of the unsupervised learning techniques. To this aim:

1 Load the data and perform a PCA. Produce a plot of the variance explained by the first components, and a scree plot.

```
# IMPORT THE DATA SET
dataf <-  read.delim('gene_expr.tsv', sep = '\t', header = T)
# DIMENSION OF THE DATA SET
dim(dataf)
```

```
## [1]   128 12627
```

```
# SUMMARY: BASICS INFORMATION ABOUT THE DATA SET
# summary(dataf) we do not perform this command because this does not give us usef
ul information as typically happens.
```

Before performing any analysis, it is always a good practice to explore the data set (e.g, computing a correlation plot, analyzing the mean and the distribution of the variables) in order to find some *hints* that can make our analysis easier or to inform us. However, due to the dimension of our data set it is impossible to compute some of these operations or it does not have sense to compute them because it would be extremely difficult to evaluate the results. For example, if we want to compute the mean of each numerical variables, to see if there is the need to scale, it would be not possible for a human to considering all the average values at the same time. This being said, let's take a look at the data set.

```
# VIEW
View(dataf)
```

By looking at the dataset we notice that we have all categorical variables with the exception of the second one that, as stated above, indicates 2 possible sub-types of patient involved in the data gathering process. We also notice that the first column indicates the id of the patients. Hence we won't consider them during the PCA.

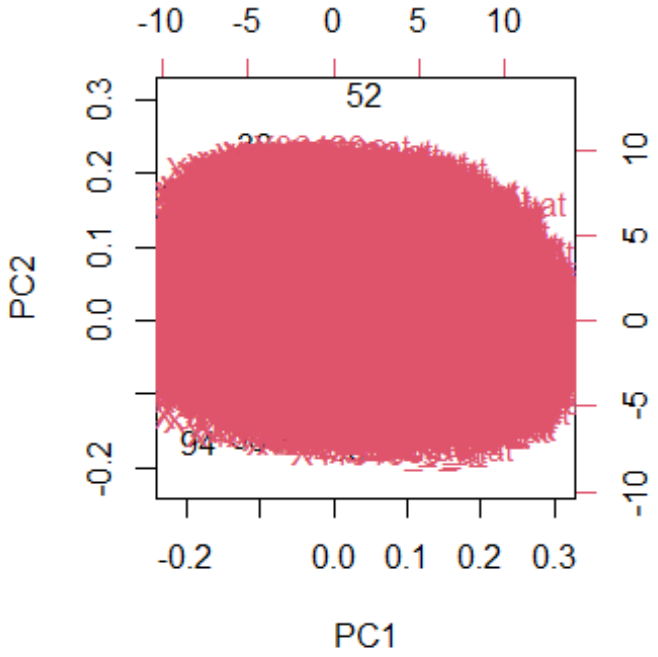
Since we have to compute the PCA and the the hierarchical clustering we know that we must consider just the continuous variable. Hence, we have to perform our analysis considering all the rows from the third to the last columns of our data set. Hence, to write our code as more general as possible we will create two "indexes".

```
beginning <- 3
end <- ncol(dataf)

# PERFORMING THE PCA ON THE CONTINUOUS VARIABLES
pr.out <- prcomp(dataf[,beginning:end], scale = T) # WE DECIDED TO SCALE THE VARIA
BLES
```

Now we want to explore the object `pr.out`: we start with the biplot:

```
biplot(pr.out, scale = T)
```

```
## Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
## arrow.len): zero-length arrow is of indeterminate angle and so skipped
```



The red arrows in the above plot indicate the first two principal component loading vectors. However, we cannot get any useful information from it. To get more useful information we decide to do the summary of `pr.out`.

```
summary(pr.out)
```

```
## Importance of components:
##                             PC1     PC2      PC3     PC4      PC5      PC6
## Standard deviation     47.8103 36.9157 27.73208 24.0204 21.29449 19.64675
## Proportion of Variance  0.1811  0.1079  0.06092  0.0457  0.03592  0.03057
## Cumulative Proportion   0.1811  0.2890  0.34991  0.3956  0.43153  0.46211
##                             PC7      PC8      PC9     PC10     PC11     PC12
## Standard deviation     18.00937 16.52815 16.08110 15.68492 14.73970 13.49120
## Proportion of Variance  0.02569  0.02164  0.02048  0.01949  0.01721  0.01442
## Cumulative Proportion   0.48780  0.50943  0.52992  0.54940  0.56661  0.58103
##                            PC13    PC14     PC15     PC16     PC17     PC18
## Standard deviation     13.46128 13.1528 12.50326 11.62651 11.33948 10.95969
## Proportion of Variance  0.01435  0.0137  0.01238  0.01071  0.01018  0.00951
## Cumulative Proportion   0.59538  0.6091  0.62147  0.63217  0.64236  0.65187
##                            PC19     PC20    PC21    PC22    PC23    PC24
## Standard deviation     10.56977 10.27269 9.98280 9.76071 9.69351 9.35307
## Proportion of Variance  0.00885  0.00836 0.00789 0.00755 0.00744 0.00693
## Cumulative Proportion   0.66072  0.66908 0.67698 0.68452 0.69196 0.69889
##                           PC25    PC26    PC27    PC28    PC29    PC30    PC31
## Standard deviation     9.07879 8.97473 8.85997 8.72421 8.59119 8.53111 8.27069
```

```
## Proportion of Variance 0.00653 0.00638 0.00622 0.00603 0.00585 0.00576 0.00542
## Cumulative Proportion  0.70542 0.71180 0.71802 0.72405 0.72989 0.73566 0.74108
##                            PC32    PC33    PC34    PC35    PC36   PC37    PC38
## Standard deviation     8.23309 8.08897 8.07028 7.83775 7.80235 7.7043 7.56167
## Proportion of Variance 0.00537 0.00518 0.00516 0.00487 0.00482 0.0047 0.00453
## Cumulative Proportion  0.74645 0.75163 0.75679 0.76165 0.76648 0.7712 0.77571
##                            PC39    PC40    PC41    PC42    PC43    PC44    PC45
## Standard deviation     7.54920 7.47852 7.40003 7.33338 7.21207 7.18165 7.07957
## Proportion of Variance 0.00451 0.00443 0.00434 0.00426 0.00412 0.00409 0.00397
## Cumulative Proportion  0.78022 0.78465 0.78899 0.79325 0.79737 0.80145 0.80542
##                            PC46    PC47    PC48    PC49    PC50    PC51    PC52
## Standard deviation     7.00761 6.88692 6.86142 6.81489 6.79102 6.70541 6.68737
## Proportion of Variance 0.00389 0.00376 0.00373 0.00368 0.00365 0.00356 0.00354
## Cumulative Proportion  0.80931 0.81307 0.81680 0.82048 0.82413 0.82769 0.83123
##                            PC53    PC54    PC55   PC56    PC57    PC58    PC59
## Standard deviation     6.61719 6.58283 6.51855 6.4543 6.42488 6.40876 6.33493
## Proportion of Variance 0.00347 0.00343 0.00337 0.0033 0.00327 0.00325 0.00318
## Cumulative Proportion  0.83470 0.83813 0.84150 0.8448 0.84807 0.85132 0.85450
##                           PC60    PC61    PC62    PC63    PC64    PC65   PC66
## Standard deviation     6.2549 6.22671 6.19791 6.17728 6.13009 6.07863 6.0457
## Proportion of Variance 0.0031 0.00307 0.00304 0.00302 0.00298 0.00293 0.0029
## Cumulative Proportion  0.8576 0.86067 0.86371 0.86674 0.86971 0.87264 0.8755
##                            PC67    PC68    PC69    PC70    PC71    PC72    PC73
## Standard deviation     6.00987 5.98143 5.95744 5.87113 5.84515 5.82817 5.76546
## Proportion of Variance 0.00286 0.00283 0.00281 0.00273 0.00271 0.00269 0.00263
## Cumulative Proportion  0.87839 0.88123 0.88404 0.88677 0.88948 0.89217 0.89480
##                            PC74    PC75    PC76    PC77    PC78    PC79    PC80
## Standard deviation     5.74950 5.69443 5.67019 5.66516 5.64871 5.60202 5.56279
## Proportion of Variance 0.00262 0.00257 0.00255 0.00254 0.00253 0.00249 0.00245
## Cumulative Proportion  0.89742 0.89999 0.90253 0.90508 0.90760 0.91009 0.91254
##                            PC81   PC82    PC83    PC84    PC85    PC86    PC87
## Standard deviation     5.53503 5.5092 5.48405 5.44800 5.42787 5.36975 5.33498
## Proportion of Variance 0.00243 0.0024 0.00238 0.00235 0.00233 0.00228 0.00225
## Cumulative Proportion  0.91497 0.9174 0.91975 0.92210 0.92444 0.92672 0.92898
##                            PC88    PC89    PC90    PC91    PC92    PC93   PC94
## Standard deviation     5.29293 5.28715 5.25939 5.22066 5.18953 5.17319 5.1529
## Proportion of Variance 0.00222 0.00221 0.00219 0.00216 0.00213 0.00212 0.0021
## Cumulative Proportion  0.93119 0.93341 0.93560 0.93776 0.93989 0.94201 0.9441
##                            PC95    PC96    PC97    PC98   PC99   PC100   PC101
## Standard deviation     5.10942 5.09202 5.07675 5.03399 5.0260 4.99505 4.96053
## Proportion of Variance 0.00207 0.00205 0.00204 0.00201 0.0020 0.00198 0.00195
## Cumulative Proportion  0.94618 0.94824 0.95028 0.95228 0.9543 0.95626 0.95821
##                           PC102  PC103   PC104   PC105   PC106   PC107   PC108
## Standard deviation     4.92004 4.8988 4.86395 4.85237 4.81879 4.80002 4.72967
## Proportion of Variance 0.00192 0.0019 0.00187 0.00186 0.00184 0.00182 0.00177
## Cumulative Proportion  0.96013 0.9620 0.96390 0.96577 0.96761 0.96943 0.97120
##                           PC109   PC110   PC111   PC112   PC113   PC114   PC115
## Standard deviation     4.69107 4.68160 4.64703 4.61168 4.59981 4.56873 4.54519
## Proportion of Variance 0.00174 0.00174 0.00171 0.00168 0.00168 0.00165 0.00164
## Cumulative Proportion  0.97295 0.97468 0.97639 0.97808 0.97975 0.98141 0.98304
##                           PC116   PC117   PC118   PC119   PC120   PC121   PC122
```

```
## Standard deviation     4.45476 4.42230 4.38206 4.36821 4.30457 4.27980 4.25439
## Proportion of Variance 0.00157 0.00155 0.00152 0.00151 0.00147 0.00145 0.00143
## Cumulative Proportion  0.98462 0.98616 0.98769 0.98920 0.99066 0.99212 0.99355
##                           PC123   PC124   PC125   PC126   PC127     PC128
## Standard deviation     4.18120 4.16837 4.14619 4.00554 3.65401 1.012e-13
## Proportion of Variance 0.00138 0.00138 0.00136 0.00127 0.00106 0.000e+00
## Cumulative Proportion  0.99493 0.99631 0.99767 0.99894 1.00000 1.000e+00
```

From this summary we get many useful information:

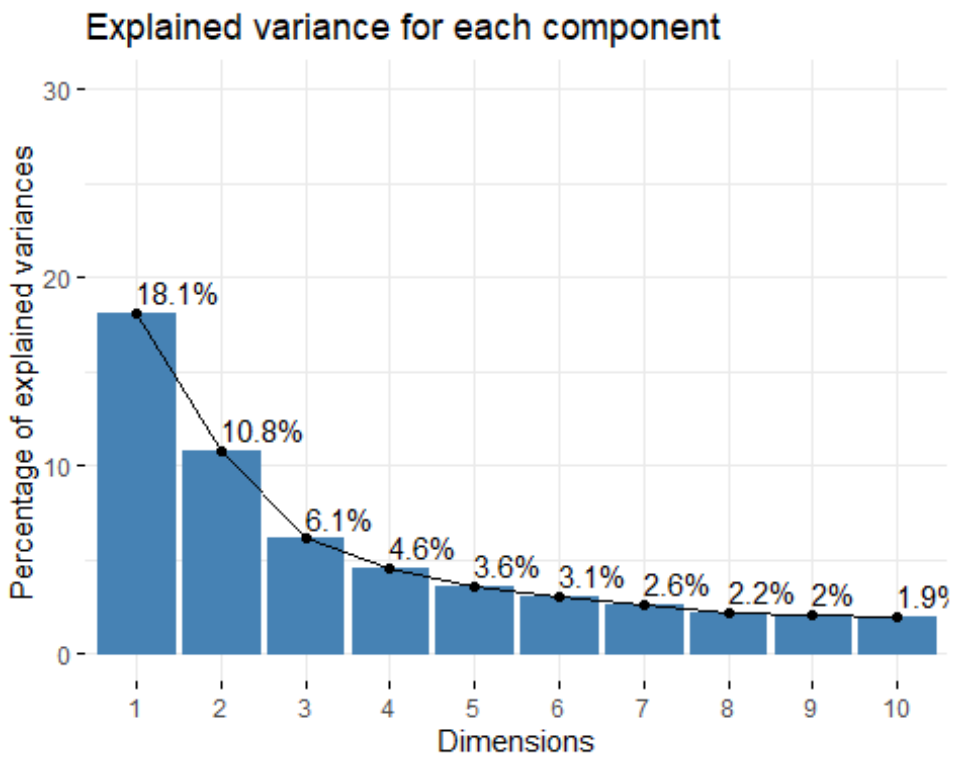
1) The first component explains almost 18% of the variance;

2) The second component explains almost 10% of the variance;

3) With 44 components we get a reasonable good level of explained variance (it is reasonable to believe that the 80% of the explained variance is a good threshold).

Now we want to visualize the contribution to the explained variance for the first 10 components.

```
fviz_eig(pr.out, addlabels=TRUE, ylim=c(0, 30), main = 'Explained variance for eac
h component')
```

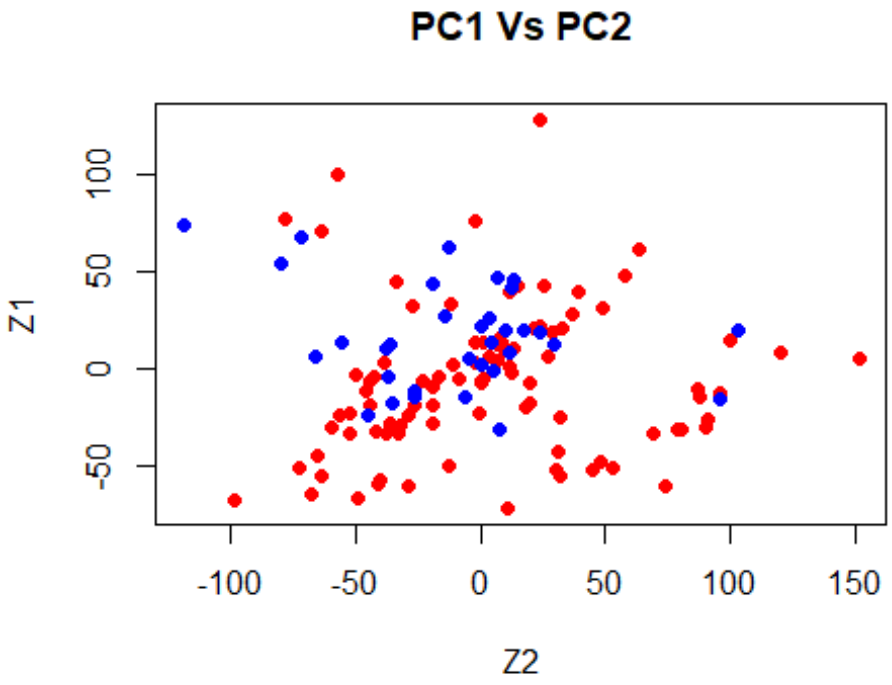


From this plot we get what we have already seen in the summary: the first two components explain a good amount of variance, but all the others explain a really small portion of it. Thus, a relatively high number of variables (44) is needed to explain a sufficiently good amount of variance.

2 Make a few scatterplots of the first principal component score vectors. Plot the observations according to the given subtypes and assess to what extent the subtypes are similar to each other and are captured by the PCA.

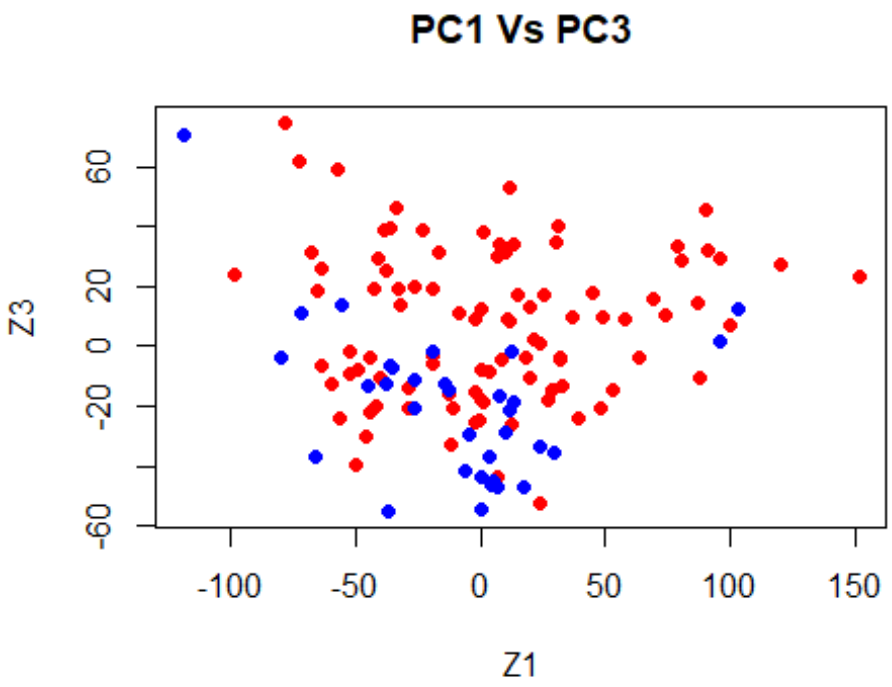
We will plot three scatterplots: 1) in the second the first score vector against the second; 2) in the third the first score vector against the third;

```
plot(pr.out$x[,1:2], col = ifelse(dataf$subtype == 'B', 'red', 'blue'), main = 'PC
1 Vs PC2', ylab = 'Z1', xlab = 'Z2', pch=19)
```



By plotting the first two principal components score vectors, we notice that the distinction between the two subgroups is not clearly defined indeed, we see that there is an overlapping of points in the middle-left area of the above plot.

```
plot(pr.out$x[, c(1, 3)],col = ifelse(dataf$subtype == 'B', 'red', 'blue'), pch=19
, xlab="Z1", ylab="Z3",main = 'PC1 Vs PC3')
```



By plotting the first and the third principal component score vector, we see that there is not a clear distinction between the two classes as well as before. Nevertheless, we can state that the observations belonging to the '*T*' class (the blue ones) are more concentrated in the bottom area of the plot. In both the plots, PC1 Vs PC2 and PC1 Vs PC3 we have seen that the two classes are not well separated and so there is an overlapping between the observations of the two groups. These results may suggest us that also the clustering won't do a perfect job because there is some noise that makes difficult to asses to which class an observation belongs. This is not hard to believe because with this huge data set there will be variables that makes everything cloudy, non-transparent.

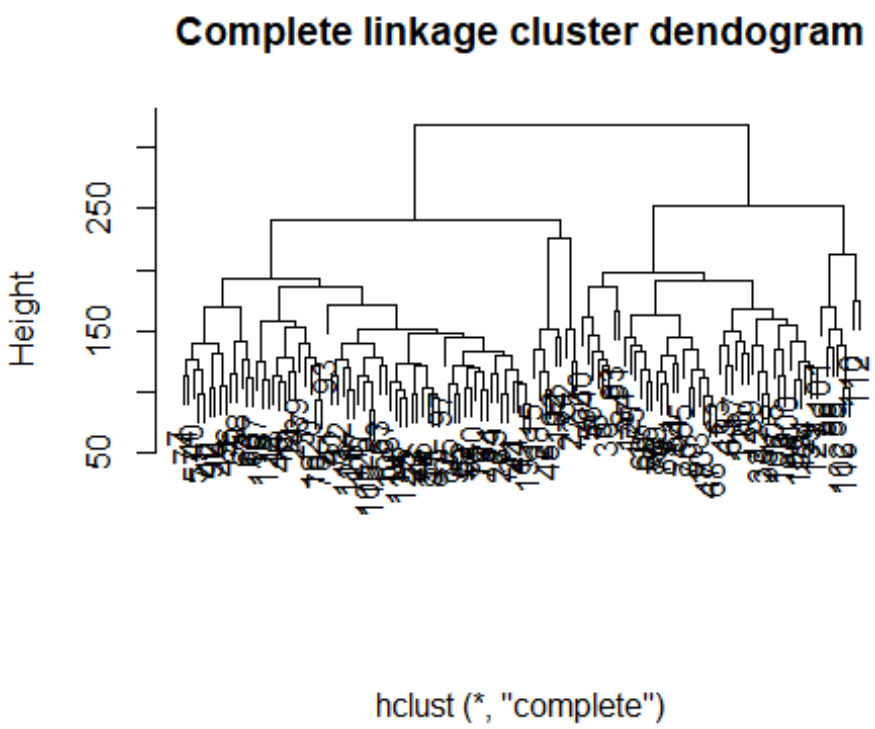
3 Hierarchically cluster the patients using complete linkage and Euclidean distance as dissimilarity measure. Cut the dendrogram so as to have two groups. Evaluate the goodness of the clustering by comparing the groups with the given subtypes. Provide a numerical similarity measure.

Before doing the hierarchical clustering we decide to standardize our data set. In this way, each gene is on the same scale. Moreover, we have already scaled our data set when we have performed the PCA.

```
dataf[,beginning:end] <- scale(dataf[,beginning:end])
```

Now we create our hierarchical clustering with the *complete linkage* and using as dissimilarity measure the *euclidean distance*.

```
hc.complete <- hclust(dist(dataf[,beginning:end]), method = 'complete')
plot(hc.complete, main = 'Complete linkage cluster dendogram', xlab = '')
```

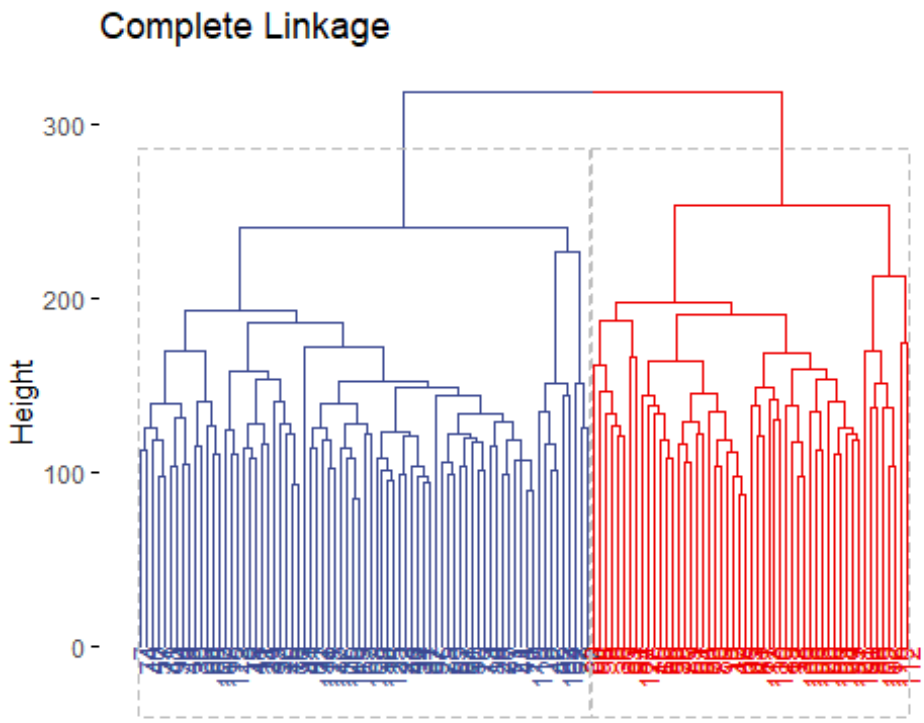


From a first look, we notice that the hierarchical clustering is able to identify two main classes. Now we cut the dendogram in order to show the two sub-groups and we also compute the number of observations for each subgroup.

```
cutted <-  cutree(hc.complete, 2)
table(cutted)

## cutted
##  1  2
## 75 53
```

Now we want to graphically represent the hierarchical clustering by defining the observations according to their groups using colors.

```
fviz_dend(hc.complete, main="Complete Linkage", cex=.7,
          k=2, # cut in two groups
          rect=TRUE,
          palette="aaas")
```



By carefully looking at the above plot we notice that the two classes (B and T) join together only at the root of the dendogram. This means that according to this hierarchical clustering the two classes are quite distinguishable. However, we already notice that the clustering is not perfect because the distribution of the observations among the two classes does not reflect the real one.

EVALUATE THE GOODNESS OF THE CLASSES

To evaluate the goodness of the clustering by comparing the groups with the given sub types and to provide a numerical similarity measure a good choice is typically the **Adjusted Rand Index** (**ARI**). The **ARI** is frequently used in cluster validation since it is a measure of agreement between two partitions: one given by the clustering process and the other defined by external criteria (in our case, we know from the beginning the true division in classes.) The formula for computing the **ARI** is the following:

$$ARI = \frac{RI - \text{expected } RI}{\max(RI) - \text{expected } RI}$$

and from that we get that the **Adjusted Rand Index** may only yield a values between 0 and +1 however, the **ARI** can yield negative values if the index is less than the expected index.

Before computing this measure of goodness of the clustering we decide to show the values that we get from the **ground truth** which is the column vector `dataf$subtype` and the ones that we get from the clustering

```
print('Ground Truth')
```

```
## [1] "Ground Truth"
```

```
table(dataf$subtype)
```

```
##
##  B  T
## 95 33
```

```
print('Complete H-Clustering \n')
```

```
## [1] "Complete H-Clustering \n"
```

```
table(cutted)
```

```
## cutted
##  1  2
## 75 53
```

As we can see, we notice that the complete linkage hierarchical clustering has misclassified 20 observations out of 128. Thus, we do not expect an high value of the **ARI**. Now we are ready to compute the **ARI** and numerically quantify the *' accuracy '* of this classification:

```
truth <- dataf$subtype
mclust::adjustedRandIndex(truth, cutted)
```

```
## [1] 0.02266644
```

As expected, the **ARI** is quite low since 20 of the 128 observations were misclassified. Hence, we can deduce that the results provided by the above cluster are close to the random partition. Hence, we want to improve the perfromance of our clustering algorithm.

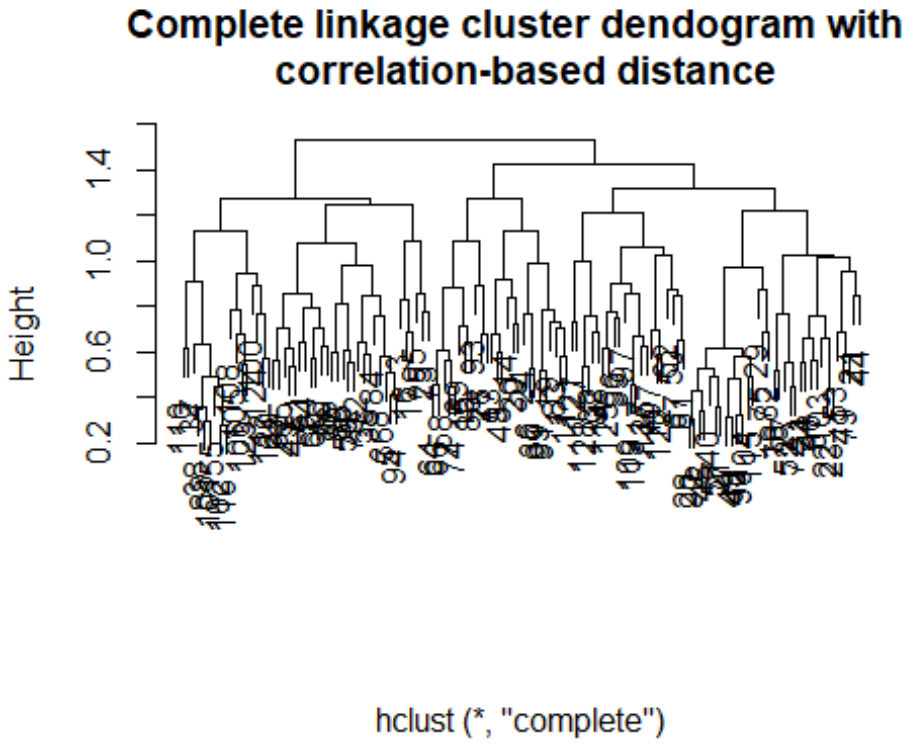
3 See if you can improve the results. For example, repeat the procedure using correlation as dissimilarity measure, and with single or average linkage. Discuss the results and pick the combination of dissimilarity measure and linkage method that works best.

As suggested, to improve the performances of the clustering we may start by redefine what it means for two or more observation to be similar or different. In the first trial, we used the *euclidean distance*, now we want to see if creating the clustering with **correlation** as dissimilarity measure improves the performance. The correlation-based distance considers two observations similar if their features are highly correlated, even tough the observed values may fall apart in terms of *Euclidean distance*. Furthermore, **correlation-based distance** focuses on the shapes of observation profiles rather than their magnitude. Let's compute the **correlation-based distance**.

```
cor.b.dis <-  as.dist(1-cor(t(dataf[beginning:end])))
```

Now that we have the correlation-based distance we can compute and plot a new hierarchical clustering.

```
hc.comp.cor <- hclust(cor.b.dis, method = 'complete')
plot(hc.comp.cor, main = 'Complete linkage cluster dendogram with \n correlation-b
ased distance', xlab = '')
```

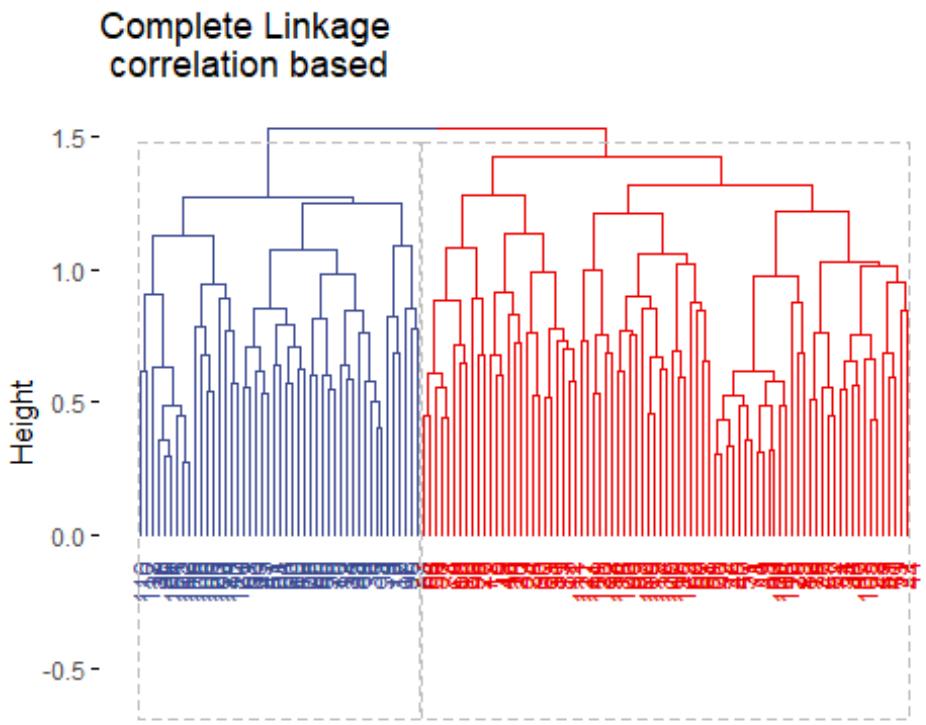


Now as before we want to cut the dendogram in order to have two clusters.

```
cut.cor <-  cutree(hc.comp.cor, 2)
table(cut.cor)

## cut.cor
##  1  2
## 81 47
```

Just from this first representation of the results we may hypothesize an improvement in the classification. Indeed, the number of observations in each group is closer then before to the ground truth. However, this is not enough for our purpose. Before continuing the comparison between the two clustering we decide to plot the new clustering highlighting the difference among the groups with colors as did before.

```
fviz_dend(hc.comp.cor, main="Complete Linkage \n correlation based", cex=.7,
          k=2, # cut in three groups
          rect=TRUE,
          palette="aaas")
```



As before, we can clearly see that two classes are well separated and they join together only at the root. Hence, the clustering shows that the two classes are quite different. However, to compare the goodness of this clustering with the previous one we compute its **ARI**.

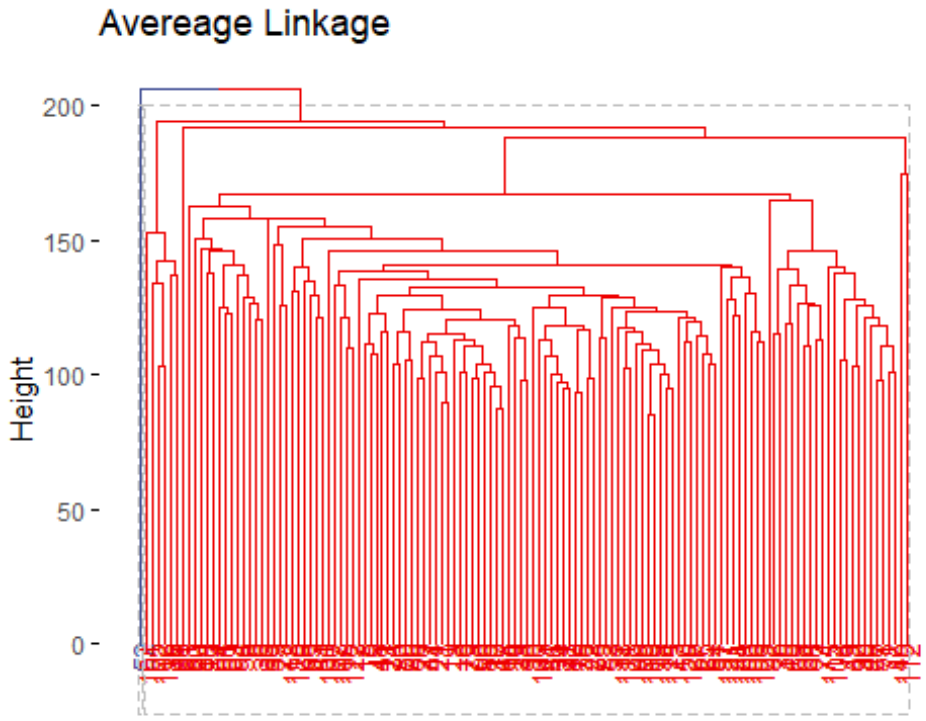
```
mclust::adjustedRandIndex(truth, cut.cor)
```

```
## [1] 0.002309898
```

The **ARI** of this new clustering is positive but quite low. As a matter of fact, the **ARI** of this new clustering is lower than the one of the first clustering. Hence, we may conclude that, in this case, there is no improvement by using the correlation as dissimilarity measure. Since our results until now are quite poor we can try to change the way we measure the dissimilarity between two groups of observations, the *linkage*. Indeed, we know from the theory that there exist many different types of *linkage*, the most used ones are: *complete* (the one that we have used until now), *average*, *single* and *centroid*. Typically, *average* and *complete* are preferred over *single* linkage, as they tend to yield more balanced dendograms. So, we could also skip to use the single linkage if the results are quite satisfying with the *average linkage*. In addition, a good choice would also be the *centroid linkage* because it is frequently used in **genomics** and, this is the case. However, with this type of *linkage* an **inversion** may occur with the consequence that two clusters are fused at height below either of the individual cluster in the dendogram. All this may lead to hard to interpret results and representation. Let's start by computing the clustering with the *average linkage*

```
hc.avg <- hclust(dist(dataf[,beginning:end]), method = 'average')
fviz_dend(hc.avg, main="Avereage Linkage", cex=.7,
          k=2, # cut in three groups
```

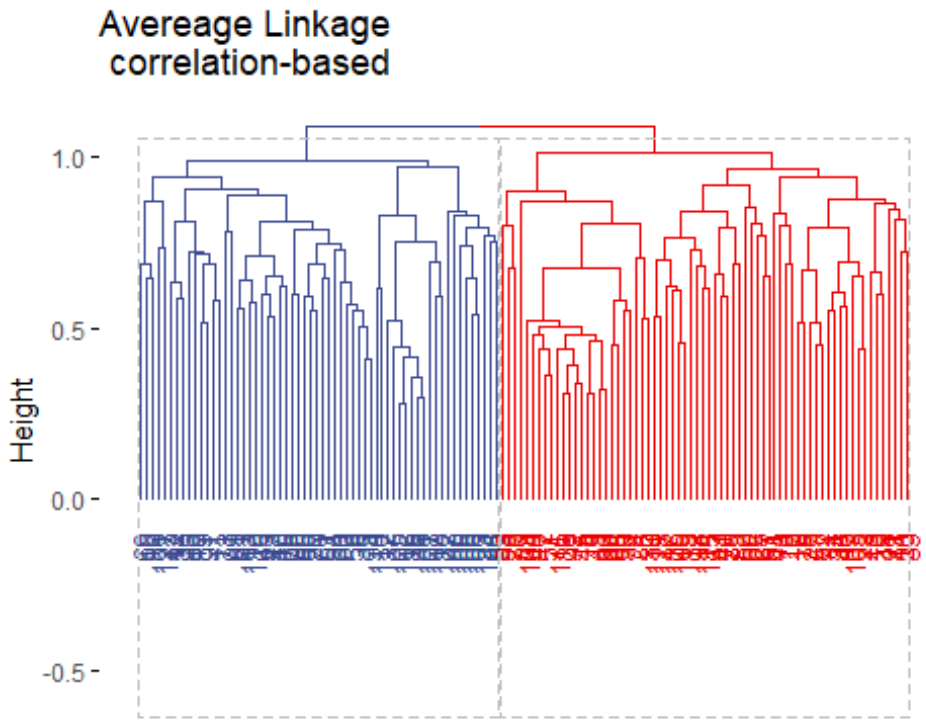
```
          rect=TRUE,
          palette="aaas")
```




```
cut.avg <- cutree(hc.avg, 2)
table(cut.avg)

## cut.avg
##   1   2
## 127   1
```

From these results we notice that there is not any improvement by applying the *average linkage* to the `hclust`. The dendogram shows a dendogram where there are many *' sub-trees '* made by just few observations. Indeed, by zooming and looking carefully the dendogram we spot something that we do not like. After the first split we have two "main" groups, a huge one and another one made by just one observation. This is also visible when we cut the three at the height of two and execute the `table()` function on the cutted dendogram. Hence we stop our analysis on this clustering which did a very poor job. Let's see if creating the dendogram with the *average linkage* and with correlation as dissimilarity measure leads to different results.

```
hc.avg.cor <- hclust(cor.b.dis, method = 'average')
fviz_dend(hc.avg.cor, main="Avereage Linkage \n correlation-based", cex=.7,
          k=2, # cut in three groups
          rect=TRUE,
          palette="aaas")
```




```
cut.avg.cor <- cutree(hc.avg.cor, 2)
table(cut.avg.cor)
```

```
## cut.avg.cor
##  1  2
## 60 68
```

```
mclust::adjustedRandIndex(truth, cut.avg.cor)
```

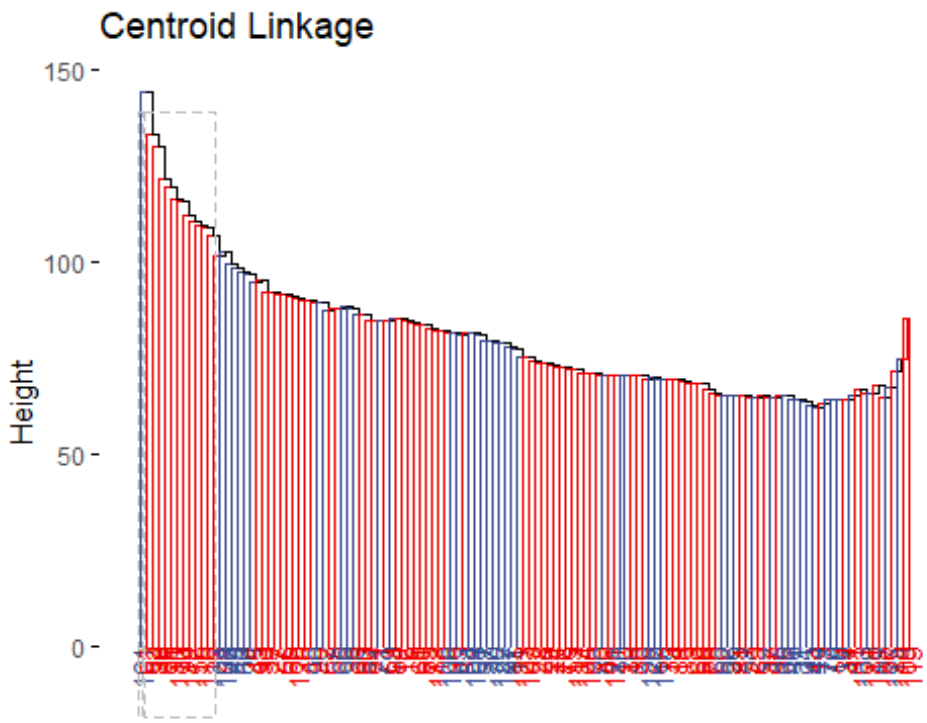
```
## [1] 0.03457948
```

As we can see from both the dendogram and the table, there is not such improvement. In fact, the results are quite far from the ground truth. Nevertheless, the *average linkage* used together with the *correlation* as dissimilarity measure leads to a small improvement of the performance. Indeed, the **ARI** associated to this clustering is the higher until now but it is still too low to stop our analysis.

Now we want to test if applying the *centroid linkage* can help us to produce a more accurate clustering.

```
hc.cen <- hclust(dist(dataf[,beginning:end]), method = 'centroid')
fviz_dend(hc.cen, main="Centroid Linkage", cex=.7,
          k=2, # cut in three groups
          rect=TRUE,
          palette="aaas")
```

```
## Warning in get_col(col, k): Length of color vector was shorter than the number
## of clusters - color vector was recycled
```

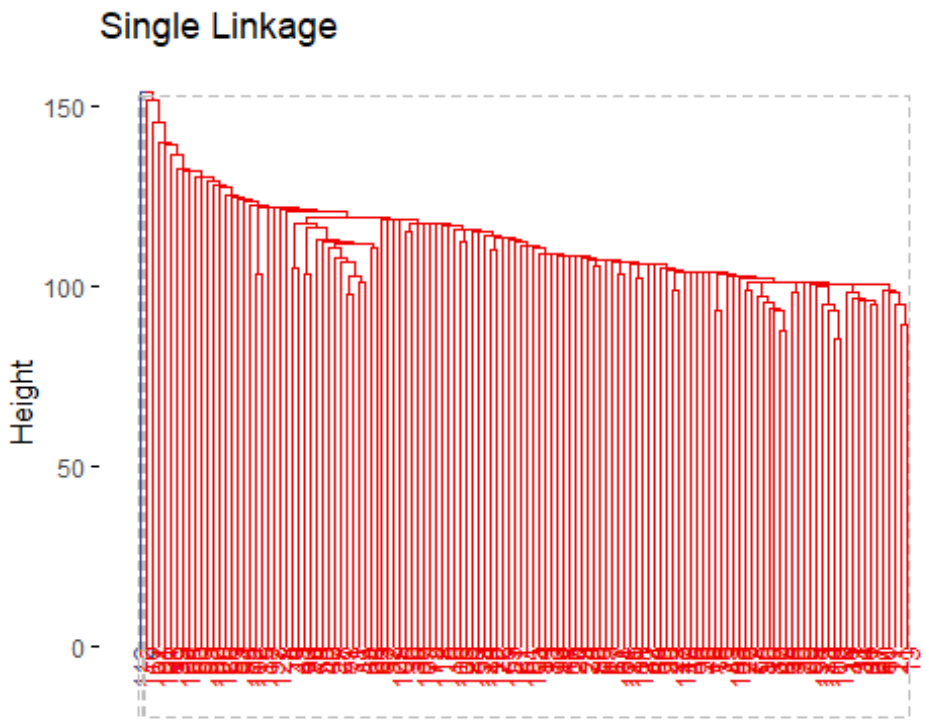



```
cut.cen <- cutree(hc.cen, 2)
table(cut.cen)

## cut.cen
##   1   2
## 127   1

mclust::adjustedRandIndex(truth, cut.cen)

## [1] 0.02908671
```

Despite the **ARI** associated to this clustering is not the lowest we are prone to discard this method because the results shown by the dendogram are less readable compared to previous models.

Before moving on we want to make a last attempt by considering the *single linkage*.

```
hc.sing <- hclust(dist(dataf[,beginning:end]), method = 'single')
fviz_dend(hc.sing, main="Single Linkage", cex=.7,
          k=2, # cut in three groups
          rect=TRUE,
          palette="aaas")
```




```
cut.sing <-  cutree(hc.sing, 2)
table(cut.sing)

## cut.sing
##   1   2
## 127   1

mclust::adjustedRandIndex(truth, cut.sing)

## [1] 0.02908671
```

The results provided by the clustering with the *single linkage* are almost identical to the ones obtained with the *centroid linkage*. We can conclude after this brief analysis that we are not satisfied with our results.

4 Another way to possibly improve the results is through gene filtering: – For example, from the full dataset select the top N (e.g., N = 200) genes that differ the most across all samples, based on PCA; repeat the hierarchical clustering; cut the dendrogram to have two groups; evaluate the results. – A different approach that is popular in gene expression analysis is to keep only the most variable genes for downstream analysis. Since most of the 10K genes have low expression or do not vary much across the experiments, this step usually minimizes the contribution of noise. An unsupervised technique would then aim to identify what explains this variance. Start again from the full dataset and keep only genes whose standard deviation is among the top 5%; perform PCA and produce a scatterplot of the first two principal components scores, coloring observations by subtype; repeat the clustering/cutting/evaluation procedure.

Let's try to follow the first hint and select the top N genes that differ the most across all samples, based on PCA. Thus we want to build a new data set with these top N genes.

A simple method to extract the results, from variables, from a PCA output is to use the function `get_pca_var()`. This function provides a list of matrices containing all the results for the active variables.

```
var <- get_pca_var(pr.out)
```

`var$contrib`contains the contributions (in percentage) of the variables to the principal components. The contribution of a variable (var) to a given principal component is (in percentage) : (var.cos2 * 100) / (total cos2 of the component).

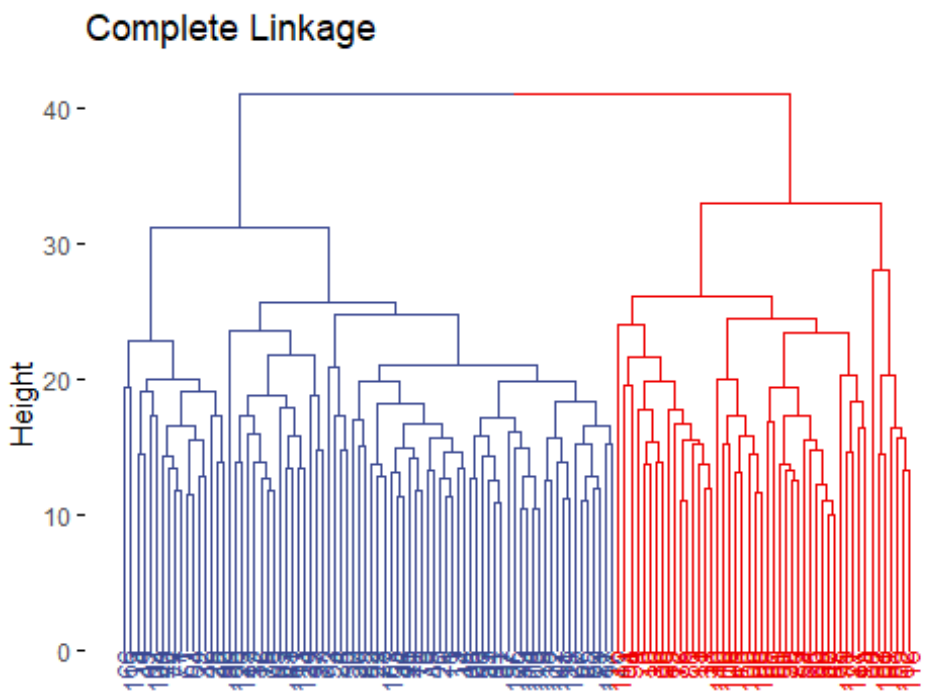
```
most_differing_genes <- head(var$contrib, 200)
names_most_variating <- rownames(most_differing_genes)
```

We decide to select, as suggested, the first 200 genes. Then, we are able to create our `new_df`.

```
new_df <- dataf[,c(names_most_variating)]
```

Now we are ready to see how the hierarchical clustering will perform on this new reduced data set. We may expect an improvement in the performance of the clustering because we are reducing the variables and we expect that there would be less 'noise'. Firstly, we will perform the hierarchical clustering with *complete linkage* based on *euclidean distance*.

```
hc.com.red.pc <- hclust(dist(new_df), method = 'complete')
# we create the cutted object
cut.cmp.red.pc <- cutree(hc.com.red.pc, 2)
table(cut.cmp.red.pc)

## cut.cmp.red.pc
##  1  2
## 80 48

fviz_dend(hc.com.red.pc, main="Complete Linkage", cex=.7,
          k=2, # cut in two groups
          palette="aaas")
```

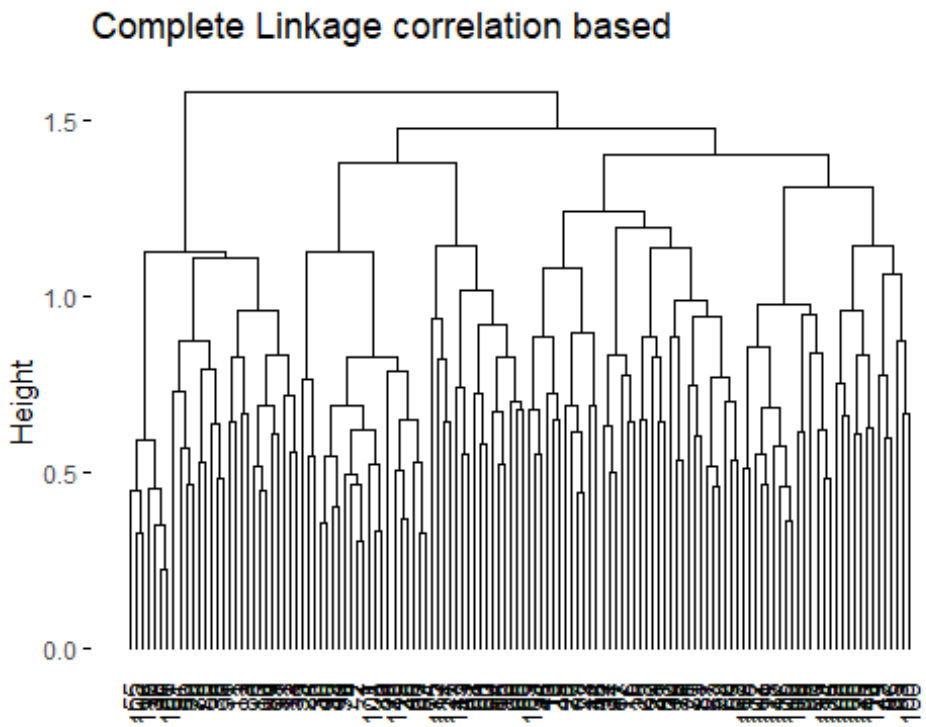


Looking at both the results of the table and the dendogram we may think that there is an improvement in the performances with this new reduced data set. Indeed, the results provided by `cut.cmp.red.pc` are closer to the results shown by the ground truth. Nevertheless, to confirm our intuition we need to compute the **ARI**.

```
mclust::adjustedRandIndex(truth, cut.cmp.red.pc)

## [1] -0.008523915
```

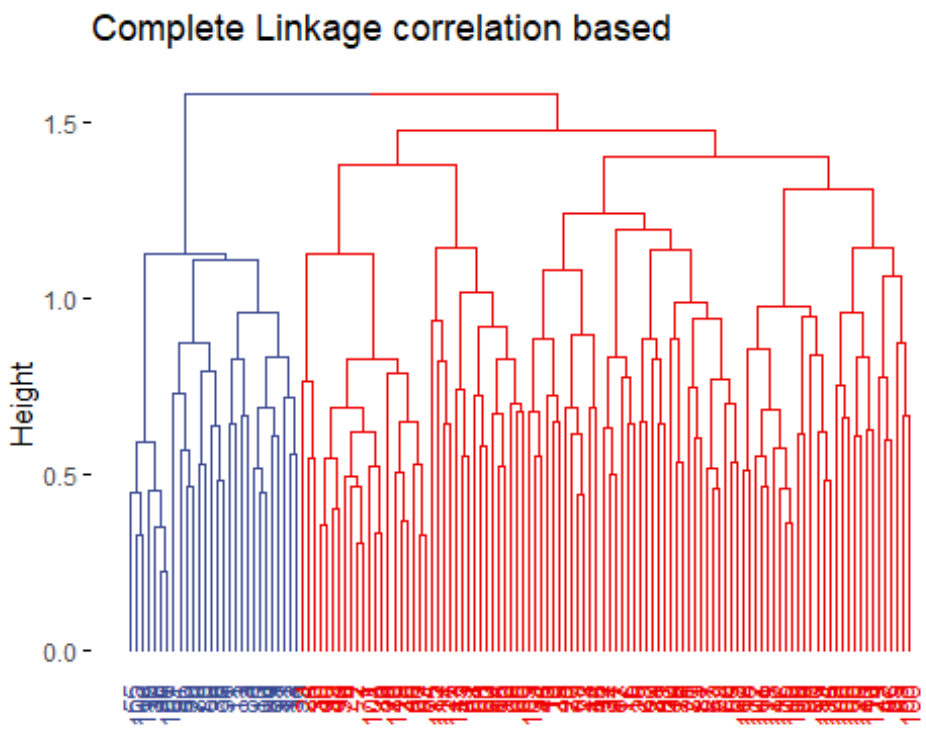
As we can see, the **ARI** is negative. This means that the results of this classification are very poor. Indeed, a negative **ARI** occurs rarely and it tells that the clustering performs worse than randomly assignment of the class. The first thing that we may do to improve the performance is to build the dendogram using the correlation-based distance instead of the euclidean one

```
hc.com.red.pc.cor <- hclust(as.dist(1-cor(t(new_df))), method = 'complete')
#plot
fviz_dend(hc.com.red.pc.cor, main="Complete Linkage correlation based", cex=.7)
```



By looking at this new dendogram we may hypothesize a worse result because we notice that the first split has created a huge group on the right and a too small group of observations on the left. Let's see if our intuition is right.

```
# we create the cutted object
cut.cmp.red.pc.cor <- cutree(hc.com.red.pc.cor, 2)
table(cut.cmp.red.pc.cor)

## cut.cmp.red.pc.cor
##   1   2
##  28 100

fviz_dend(hc.com.red.pc.cor, main="Complete Linkage correlation based", cex=.7, k=
2, # cut in two groups
          palette="aaas")
```




```
mclust::adjustedRandIndex(truth, cut.cmp.red.pc.cor)

## [1] -0.06355952
```

As we can see, there are is higher but negative. So, we tend to prefer the clustering done with the *euclidean distance*. To conclude, we do not go further in our analysis with this new data set because we do not expect any improvement by changing the linkage method since there was any with the full data set. As a general final comment, we would have expected an increase of the performance with this reduce data set but there could be at least two factors that could have negatively affected the results:

1) the variance explained by the first 200 genes is not enough for our purpose (the performance of the cluster would definitely increase with an higher number of genes);

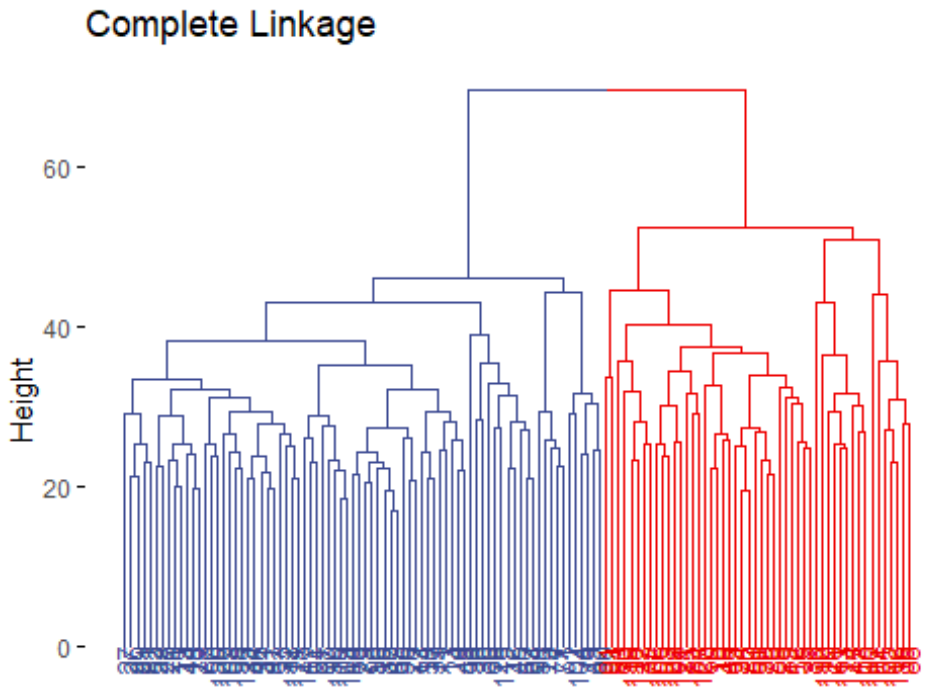
2) the data is still built on the base of the PCA that as we have seen as its limits.

To prove what we have just said we re-do the above computation considering an higher number of genes, let's say 600.

```
most_differing_genes <- head(var$contrib, 600)
names_most_variating <- rownames(most_differing_genes)
new_df <- dataf[,c(names_most_variating)]

hc.com.red.pc <- hclust(dist(new_df), method = 'complete')
# we create the cutted object
cut.cmp.red.pc <- cutree(hc.com.red.pc, 2)
table(cut.cmp.red.pc)
```

```
## cut.cmp.red.pc
##  1  2
## 78 50
```

```
fviz_dend(hc.com.red.pc, main="Complete Linkage", cex=.7,
          k=2, # cut in two groups
          palette="aaas")
```




```
mclust::adjustedRandIndex(truth, cut.cmp.red.pc)
```

```
## [1] 0.03862173
```

As we can see this new clustering performed much better than the previous one. Moreover, it is the most performing one until now.

Now, we move on and we consider the second hint given: we now want to create a second reduced data set where we keep only the most variable genes for downstream analysis.

```
dataf <-  read.delim('gene_expr.tsv', sep = '\t', header = T)
top_600 <- as.matrix(head(sort(apply(dataf[,-c(1,2)],2, var),decreasing = T), n =
600))
colnames(top_600) <- 'variance'
#rownames(top_200)
new_df2 <-  dataf[, c(rownames(top_600))]
new_df2 <- scale(new_df2)
```

Now that we have our `new_df2` we can: * perform PCA * produce the scatterplot of the first two components scores, coloring observations by subtype; * repeat the clustering, cutting and evaluation procedure.

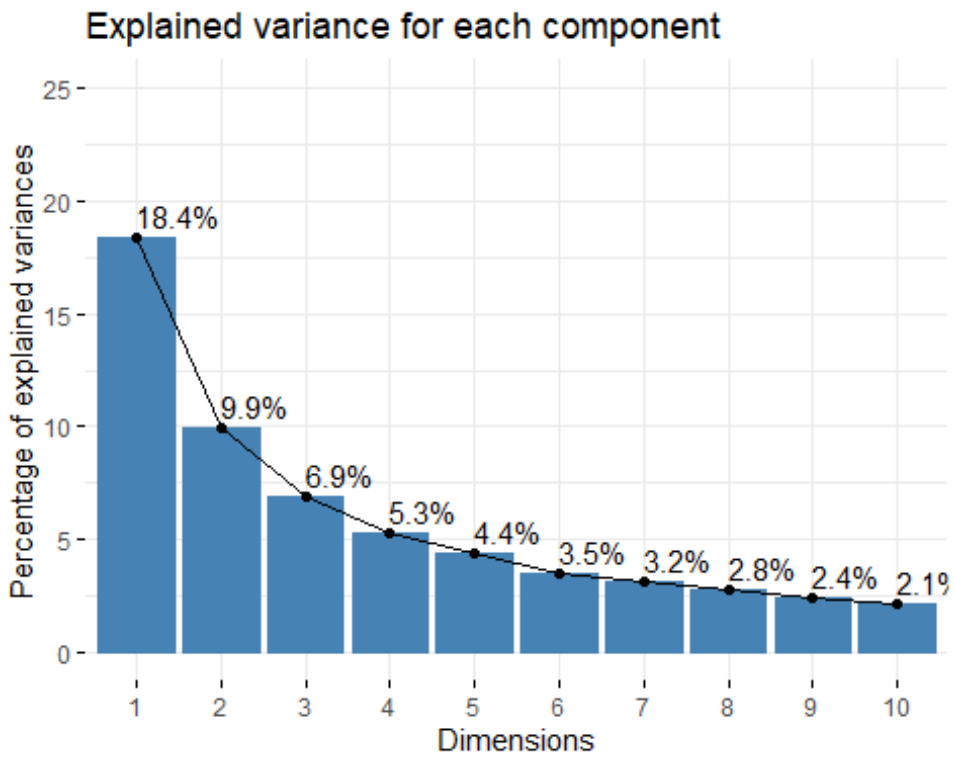
```r
pr.out2 <- prcomp(new_df2, scale = T)
summary(pr.out2)
```

```
## Importance of components:
##                            PC1     PC2     PC3     PC4    PC5     PC6     PC7
## Standard deviation     10.507 7.72193 6.44726 5.63261 5.1207 4.60658 4.35361
## Proportion of Variance  0.184 0.09938 0.06928 0.05288 0.0437 0.03537 0.03159
## Cumulative Proportion   0.184 0.28336 0.35264 0.40552 0.4492 0.48459 0.51618
##                            PC8    PC9    PC10    PC11    PC12    PC13   PC14
## Standard deviation     4.06977 3.8107 3.58789 3.41304 3.17965 3.07099 2.9090
## Proportion of Variance 0.02761 0.0242 0.02145 0.01941 0.01685 0.01572 0.0141
## Cumulative Proportion  0.54378 0.5680 0.58944 0.60886 0.62571 0.64143 0.6555
##                           PC15    PC16   PC17    PC18    PC19    PC20    PC21
## Standard deviation     2.85289 2.75419 2.5922 2.53549 2.48012 2.40269 2.36550
## Proportion of Variance 0.01356 0.01264 0.0112 0.01071 0.01025 0.00962 0.00933
## Cumulative Proportion  0.66909 0.68174 0.6929 0.70365 0.71390 0.72352 0.73285
##                           PC22    PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     2.27274 2.25719 2.18354 2.14341 2.05625 2.01593 1.95356
## Proportion of Variance 0.00861 0.00849 0.00795 0.00766 0.00705 0.00677 0.00636
## Cumulative Proportion  0.74146 0.74995 0.75790 0.76555 0.77260 0.77937 0.78573
##                           PC29    PC30    PC31    PC32    PC33    PC34    PC35
## Standard deviation     1.93344 1.88744 1.86218 1.85666 1.82124 1.78145 1.74677
## Proportion of Variance 0.00623 0.00594 0.00578 0.00575 0.00553 0.00529 0.00509
## Cumulative Proportion  0.79196 0.79790 0.80368 0.80943 0.81496 0.82024 0.82533
##                           PC36    PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     1.68814 1.67419 1.67078 1.66867 1.61432 1.59878 1.58251
## Proportion of Variance 0.00475 0.00467 0.00465 0.00464 0.00434 0.00426 0.00417
## Cumulative Proportion  0.83008 0.83475 0.83940 0.84404 0.84839 0.85265 0.85682
##                           PC43    PC44    PC45    PC46    PC47    PC48    PC49
## Standard deviation     1.57371 1.54673 1.52206 1.49194 1.47950 1.42455 1.40925
## Proportion of Variance 0.00413 0.00399 0.00386 0.00371 0.00365 0.00338 0.00331
## Cumulative Proportion  0.86095 0.86494 0.86880 0.87251 0.87616 0.87954 0.88285
##                           PC50    PC51    PC52    PC53    PC54    PC55    PC56
## Standard deviation     1.40017 1.39172 1.36893 1.35051 1.34425 1.32508 1.31177
## Proportion of Variance 0.00327 0.00323 0.00312 0.00304 0.00301 0.00293 0.00287
## Cumulative Proportion  0.88612 0.88934 0.89247 0.89551 0.89852 0.90144 0.90431
##                           PC57   PC58    PC59    PC60    PC61    PC62    PC63
## Standard deviation     1.29022 1.2723 1.25787 1.21491 1.20719 1.20154 1.19296
## Proportion of Variance 0.00277 0.0027 0.00264 0.00246 0.00243 0.00241 0.00237
## Cumulative Proportion  0.90709 0.9098 0.91242 0.91488 0.91731 0.91972 0.92209
##                           PC64    PC65    PC66    PC67    PC68    PC69    PC70
## Standard deviation     1.17262 1.15853 1.14759 1.13124 1.11944 1.10359 1.10068
## Proportion of Variance 0.00229 0.00224 0.00219 0.00213 0.00209 0.00203 0.00202
## Cumulative Proportion  0.92438 0.92662 0.92881 0.93095 0.93303 0.93506 0.93708
##                           PC71    PC72    PC73    PC74   PC75    PC76    PC77
## Standard deviation     1.09017 1.08098 1.06353 1.05389 1.0388 1.03505 1.01884
## Proportion of Variance 0.00198 0.00195 0.00189 0.00185 0.0018 0.00179 0.00173
## Cumulative Proportion  0.93906 0.94101 0.94290 0.94475 0.9466 0.94833 0.95006
##                           PC78    PC79    PC80    PC81    PC82   PC83    PC84
## Standard deviation     1.01157 0.99694 0.97534 0.96307 0.95768 0.9482 0.94187
## Proportion of Variance 0.00171 0.00166 0.00159 0.00155 0.00153 0.0015 0.00148
```

```
## Cumulative Proportion  0.95177 0.95342 0.95501 0.95655 0.95808 0.9596 0.96106
##                           PC85    PC86    PC87    PC88    PC89    PC90    PC91
## Standard deviation     0.92619 0.92117 0.91277 0.90873 0.89695 0.88538 0.87282
## Proportion of Variance 0.00143 0.00141 0.00139 0.00138 0.00134 0.00131 0.00127
## Cumulative Proportion  0.96249 0.96390 0.96529 0.96667 0.96801 0.96932 0.97059
##                           PC92    PC93    PC94    PC95    PC96    PC97   PC98
## Standard deviation     0.86095 0.85252 0.84292 0.83787 0.83272 0.82585 0.8128
## Proportion of Variance 0.00124 0.00121 0.00118 0.00117 0.00116 0.00114 0.0011
## Cumulative Proportion  0.97182 0.97303 0.97422 0.97539 0.97654 0.97768 0.9788
##                           PC99   PC100   PC101   PC102   PC103   PC104   PC105
## Standard deviation     0.80900 0.79587 0.78408 0.76820 0.76463 0.74614 0.73890
## Proportion of Variance 0.00109 0.00106 0.00102 0.00098 0.00097 0.00093 0.00091
## Cumulative Proportion  0.97987 0.98093 0.98195 0.98294 0.98391 0.98484 0.98575
##                          PC106   PC107   PC108   PC109  PC110   PC111   PC112
## Standard deviation     0.72357 0.71827 0.70821 0.70018 0.6926 0.68569 0.67875
## Proportion of Variance 0.00087 0.00086 0.00084 0.00082 0.0008 0.00078 0.00077
## Cumulative Proportion  0.98662 0.98748 0.98832 0.98913 0.9899 0.99072 0.99148
##                          PC113   PC114   PC115   PC116   PC117  PC118   PC119
## Standard deviation     0.66097 0.65447 0.64197 0.63166 0.62276 0.6007 0.59686
## Proportion of Variance 0.00073 0.00071 0.00069 0.00066 0.00065 0.0006 0.00059
## Cumulative Proportion  0.99221 0.99293 0.99361 0.99428 0.99492 0.9955 0.99612
##                          PC120   PC121   PC122  PC123   PC124   PC125   PC126
## Standard deviation     0.59213 0.56674 0.56141 0.5494 0.53611 0.52059 0.49763
## Proportion of Variance 0.00058 0.00054 0.00053 0.0005 0.00048 0.00045 0.00041
## Cumulative Proportion  0.99670 0.99724 0.99776 0.9983 0.99875 0.99920 0.99961
##                          PC127     PC128
## Standard deviation     0.48325 4.666e-15
## Proportion of Variance 0.00039 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00
```

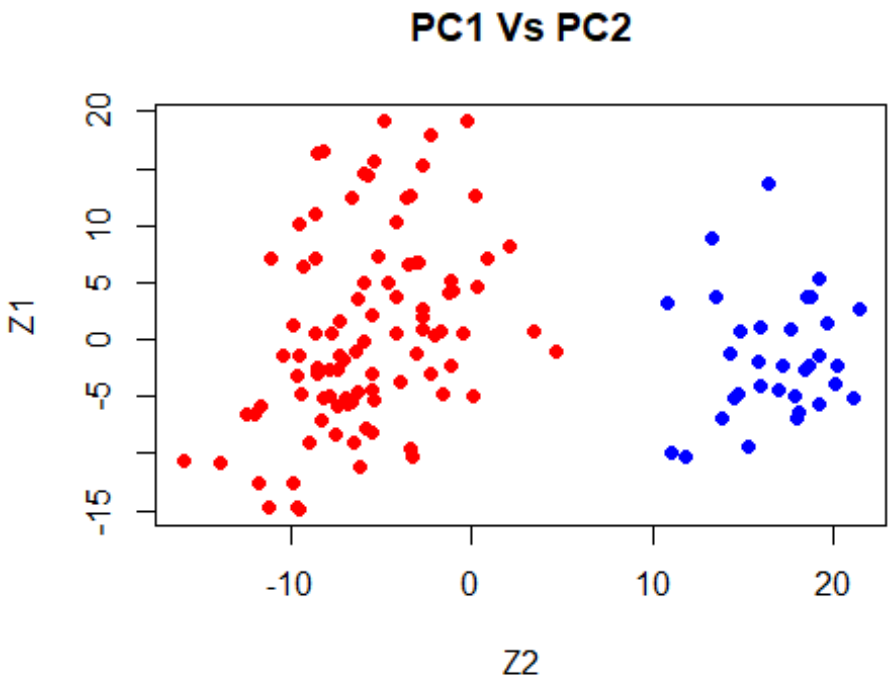
From the summary of the PCA we get that: * the first component explains the 18% of the variance; * the first six components explain almost half of the total variance; * the 75% of the variance is explained by 23 components. Now we want to plot the results as we did before.

```
fviz_eig(pr.out2, addlabels=TRUE, ylim=c(0, 25), main = 'Explained variance for ea
ch component')
```



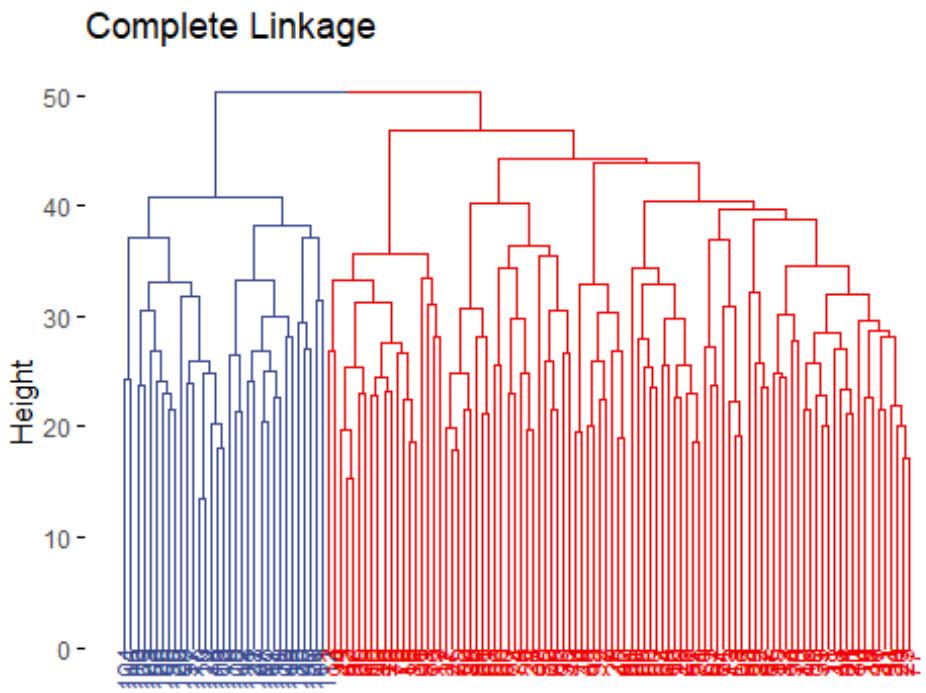
From this representation we can better appreciate the importance of the first component compared to the others. Now we want to plot the scatterplot of Z1 vs Z2, the two first principal components scores vectors and focusing on the two classes of observations.

```
plot(pr.out2$x[,1:2], col = ifelse(dataf$subtype == 'B', 'red', 'blue'), main = 'PC1 Vs PC2', ylab = 'Z1', xlab = 'Z2', pch=19)
```



The results that we obtained is in line with our expectations. Indeed, by reducing the data set, we have discarded all the genes' information that produces noise inside the data set. Hence, we can clearly see the two groups of observations. Moreover, we also notice that there is no overlapping between the two groups and that they are well separated. Now we are ready to compute the dendogram associated with this new reduce data set. We are quite confident that the clustering will do a relatively good job.

```
hc.com.red.2 <- hclust(dist(new_df2), method = 'complete')
# we create the cutted object
cut.cmp.red.2 <- cutree(hc.com.red.2, 2)
table(cut.cmp.red.2)

## cut.cmp.red.2
##  1  2
## 95 33

fviz_dend(hc.com.red.2, main="Complete Linkage", cex=.7,
          k=2, # cut in two groups
          palette="aaas")
```



The results are quite promising since the output of the `table(cut.cmp.red.2)` are equal to the ones of the ground truth. Let's give a quantitative measure of the goodness of this clustering with the well-known **ARI**.

```
mclust::adjustedRandIndex(truth, cut.cmp.red.2)

## [1] 1
```

The **ARI** shows a **perfect agreement**. This clustering correctly classified the observations in the reduced data set.
Since the results is completely satisfying we stop here our analysis also because, we have experienced no improvement by changing the linkage or the way we compute the distance between the observations.

5 What do you observe and what conclusions do you make? Before the final comment on the results of our analysis we want to sum up the results of the different **ARIs** into a table.

```
type_of_clustering <-  c('Complete & Euclidean', 'Complete & Correlation', 'Centro
id', 'Single', 'Complete & Euclidean Red1', 'Complete & Euclidean Red2')
ARIs <-  c()

ARIs <- append(ARIs,mclust::adjustedRandIndex(truth, cutted))
ARIs <- append(ARIs,mclust::adjustedRandIndex(truth, cut.cor))
ARIs <- append(ARIs,mclust::adjustedRandIndex(truth, cut.cen))
ARIs <- append(ARIs,mclust::adjustedRandIndex(truth, cut.sing))
ARIs <- append(ARIs,mclust::adjustedRandIndex(truth, cut.cmp.red.pc))
ARIs <- append(ARIs,mclust::adjustedRandIndex(truth, cut.cmp.red.2))
results <- as.data.frame(ARIs)
```

```
rownames(results) <- type_of_clustering
colnames(results) <- 'ARI'
results
```

```
##                                   ARI
## Complete & Euclidean      0.022666439
## Complete & Correlation    0.002309898
## Centroid                  0.029086707
## Single                    0.029086707
## Complete & Euclidean Red1 0.038621733
## Complete & Euclidean Red2 1.000000000
```

In this table are represented the ARIs associated with the different clustering. As we can see, the performances of the clustering with the full data set are quite low despite the *linkage* and the *dissimilarity measure* used. On the other hand, we notice an improvement in the performance when we reduce the variables considered in the analysis and consequently the noise inside the data set. As a matter of fact, the clustering correctly classify the observation when we consider the data set with only the most variable genes. After this brief recap, it seems appropriate to retrace all the steps of our analysis.

The aim of this exercise was to investigate and get the best out of the **clustering** an unsupervised learning technique used to find hidden subgroups inside a data set. Actually, we already knew the hidden groups and so our main objective was to see how the **clustering** would have behaved. The first step of our analysis as usual involved the data exploration however, proceeding with the usual computations was not convenient due to the dimension of our data frame. Thus, we have overcame this issue by computing a PCA analysis that serves also as a tool for data visualization. Indeed, we have examined the two-dimensional scatterplots of the data based on the first principal components. Actually, looking at the low dimensional representation of the data helped us to understand that although we were considering just the 'first' components, our data frame was 'dirty'/'cloudy' and it was very difficult to distinguish between the two hidden groups. As a matter of fact, in the first two scatterplots there is an overlapping of points of the two classes that makes it difficult to make a clear right division. This was also evident when we have tried to compute the **ARI** of the clustering made with this data set. Indeed, we have seen very low performances also after having changed the *linkage method* and the *dissimilarity measure* among the observations.

Hence, the next move was centered on the data set with the aim to reduce it. Indeed, clustering methods are not very robust to perturbation of the data so, if we remove a subset of observations the clusters obtained are quite different. As a matter of fact, we have manipulated the original data set in two different ways: in a first attempt we have selected the top 200 genes that differ the most across all samples, based on PCA. However, this does not lead to an improvement. Probably, the number of genes considered was too low and thus, unable to explain a reasonable amount of variance that would have helped to find the differences among the observation of the two classes. In addition, during this step of the analysis we have faced an unexpected result: a **a negative value of the ARI**. This result is quite strange especially if we take a look at the formula: it is impossible to have a negative denominator and so the negative value comes from the numerator. Indeed, a negative ARI says that the agreement is less than what is expected from a random result. This means the results are 'orthogonal' or 'complementary' to some extend. With

this we mean that the clustering results are more different than random i.e. there is a *pattern* to the differences.

As consequence, we have tried to increase the number of genes considered in this new data set. This lead to an improvement of the performances of the hierarchical clustering but the results was still not completely satisfying. The second attempt involved an approach that is frequently used in gene expression analysis. With this method we only keep the most variables genes. We made this choice because many genes do not vary much among the different observations and others have low expression. This step usually minimizes the contribution of noise and it worked well also in this case. In fact, the clusters obtained with this new reduced data set were completely satisfying because we get an ARI = 1 which means perfect agreement between the groups obtained from the clustering and the groups defined by the ground truth. These clusters were built with the *complete linkage and the euclidean distance as dissimilarity measure*. To conclude, what we have seen is that the hierarchical clustering as unsupervised learning technique is not very robust and may vary a lot with changes of the *linkage*, the *dissimilarity measure among variables* and in the data set. We are satisfied with the results just because we knew from the beginning the real division of the groups but without it it would have been really difficult to state which was the best clustering. Lastly, we want to steer the attention to the reduction of the data set that has been the key factor to de-noising the data set and allows a better analysis of the data.