Project Network-based Data Analysis

Annalisa Xamin

Contents

Data selection

The analysis will use the dataset GSE20437 obtained from GEO.The dataset is generated from Affymetrix HU133A microarrays and contains 42 tissue samples.

In detail, the data includes:

- 18 reduction mammoplasty (RM) breast epithelium samples,
- 18 histologically normal (HN) epithelial samples from breast cancer patients (9 ER+ and 9 ER-), and
- 6 histologically normal epithelial samples from prophylactic mastectomy patients.

Note that sample numbers correspond to individual patient samples.

```
# download the GSE20437 expression data series
#gse <- getGEO("GSE20437", destdir= './data/', getGPL = F)

# load the local copy of the data
gse <- getGEO(file = "./data/GSE20437_series_matrix.txt.gz", getGPL = FALSE)
```

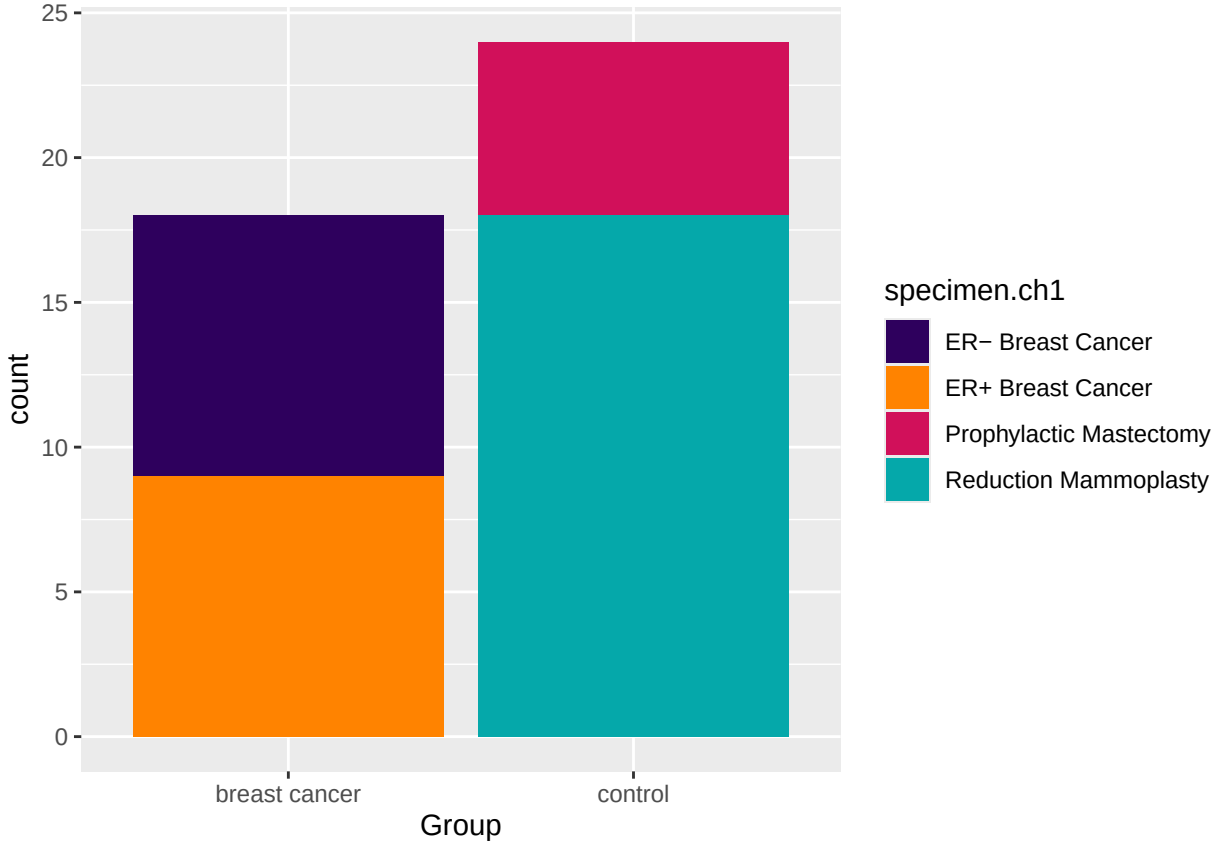
```
# getGEO returns a list of expression objects, but...
length(gse)
```

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

```
# shows us there is only one object in it.
# We assign it to the same variable.
#gse <- gse[[1]] # run only if you download data and
# if you don't use the local copy
```

```
# extract metadata
metadata <- data.frame(gse@phenoData@data)
```

```
p <- ggplot(metadata, aes(x=disease.state.ch1, fill=specimen.ch1))+
  geom_bar()+
  scale_fill_manual(values = my_colors[c(1,4,6,7)])
p + labs(x = "Group")
```



Exploratory analysis

```
# show what we have:
show(gse)
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 22283 features, 42 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM512539 GSM512540 ... GSM512580 (42 total)
##   varLabels: title geo_accession ... tissue:ch1 (38 total)
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
##   pubMedIds: 20197764
## Annotation: GPL96
```

The actual expression data are accessible in the `exprs` section of `gse`, an Expression Set and the generic data class that BioConductor uses for expression data.

```
head(exprs(gse))
```

```
##           GSM512539 GSM512540 GSM512541 GSM512542 GSM512543 GSM512544 GSM512545 GSM512546 GSM512547 G
## 1007_s_at    2461.4    3435.7    1932.5    2377.7    3055.3    2978.1    2348.5    2963.9    2776.9
## 1053_at        26.7     159.0      31.2     140.7      69.9      98.5      37.0      59.9      86.7
## 117_at         82.6     243.4     150.2      95.1     209.3     103.4      91.2     168.4     162.7
## 121_at        942.3     897.5     840.8     870.9     685.4     791.8     886.5     954.2     843.1
## 1255_g_at      71.8      87.9      75.4      58.1      31.8      40.3      70.5      43.3      51.6
## 1294_at       630.2     571.4     346.3     679.9    1289.3     421.1     417.6     811.6     778.1
##           GSM512550 GSM512551 GSM512552 GSM512553 GSM512554 GSM512555 GSM512556 GSM512557 GSM512558 G
## 1007_s_at    3037.1    3545.8    3322.6    1963.7    3609.6    2078.9    4138.6    4260.7    2453.6
## 1053_at        82.9      97.7      69.7      82.0      45.6      84.5      31.7      37.4      82.4
## 117_at        113.5      80.0     186.4     106.6     145.6     144.4     133.6     278.6     173.0
## 121_at        912.2     911.6     862.4     705.0     984.6     853.8     846.8    1273.0     833.6
## 1255_g_at      53.7      30.5      15.2      42.5      76.6      88.2      90.6      65.8      25.8
## 1294_at       987.7     938.5     924.6     480.8    1054.1     632.0     448.0    1345.2    1248.9
##           GSM512561 GSM512562 GSM512563 GSM512564 GSM512565 GSM512566 GSM512567 GSM512568 GSM512569 G
## 1007_s_at    4340.1    3155.3    2390.3    2738.8    3233.1    2836.6    2915.4    3457.5    2798.7
## 1053_at        76.7     100.3     115.4      14.1      47.6      77.1      47.1      47.0      83.2
## 117_at        168.0      95.2      73.6     122.7     107.6     120.9     143.4      92.5      72.1
## 121_at        827.0     629.4     709.2     305.6     877.4     425.7     643.8     771.3     681.1
## 1255_g_at      87.9      44.6      59.3      12.0      82.1      59.2      62.2      28.3      97.6
## 1294_at      2218.1    1321.1     606.7    1709.9     980.8    1268.4     955.8    1157.5     888.6
##           GSM512572 GSM512573 GSM512574 GSM512575 GSM512576 GSM512577 GSM512578 GSM512579 GSM512580
## 1007_s_at    3669.5    3310.1    3942.2    4520.4    3596.1    2989.0    3164.5    2764.3    4258.5
## 1053_at        24.1       8.8      44.6      54.7      56.7      89.9      63.4      57.0      59.5
## 117_at        165.8     141.6      97.1     132.7     124.3     210.5     131.4      89.6     123.3
## 121_at        746.9    1090.3    1008.7     718.6     988.4     295.9     957.3     630.8     869.2
## 1255_g_at      53.0      39.9      11.0      50.2      60.0      34.3      33.5      61.7      50.4
## 1294_at      1138.5     483.0    1326.5    1179.4     668.3     863.2    1055.5    1287.6    1127.8
```

To conveniently access the data rows and columns present in `exprs(gse)`, this matrix is assigned to its own variable `ex`.

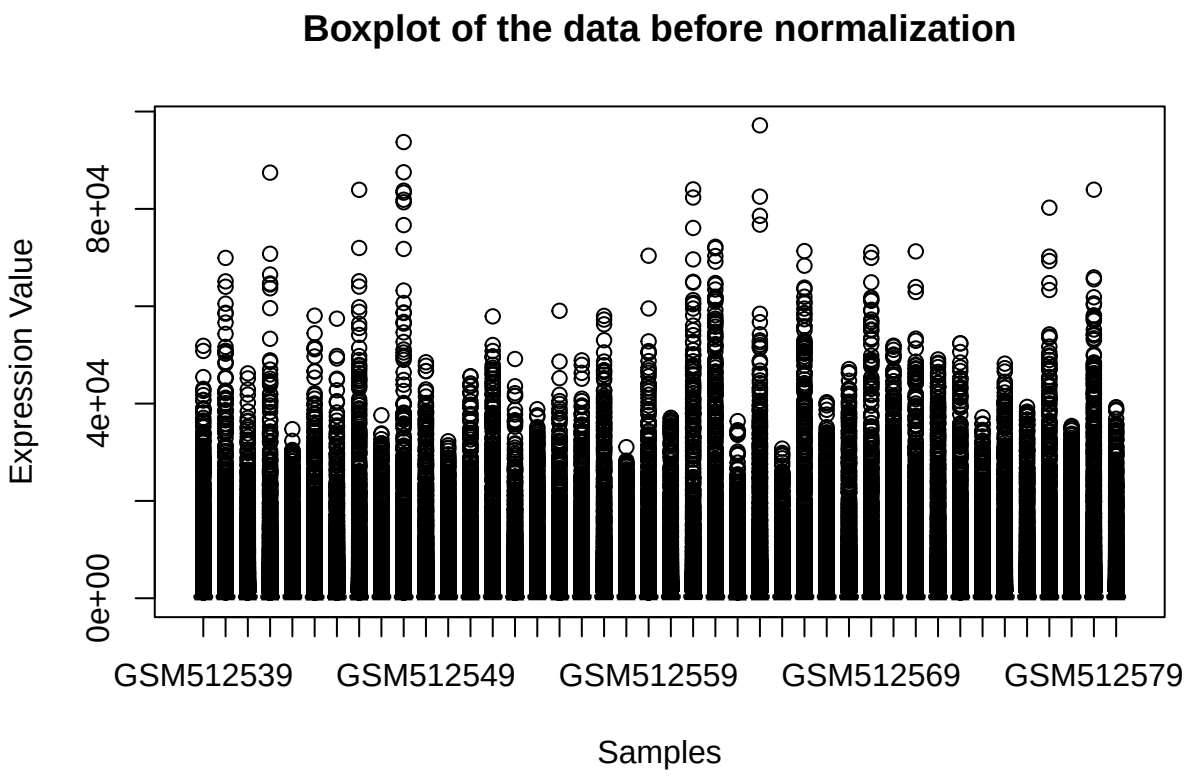
```
# exprs (gse) is a matrix that we can assign to its own variable, to
# conveniently access the data rows and columns
ex <- exprs(gse)
dim(ex) # 42 sample, 22283 genes
```

```
## [1] 22283    42
```

The dataset contains gene expression data of 22283 genes (rows) from 42 patients (columns).

Pre-processing

```
# Analyze value distributions
boxplot(ex, main = 'Boxplot of the data before normalization',
        xlab = "Samples",
        ylab = "Expression Value",
        varwidth = TRUE
        )
```



The boxplot shows that scaling is necessary. So, in this case, I try to apply a log transformation to the data.

```
ex2<-log(ex)
ex2 <- na.omit(as.matrix(ex2))
#dim(ex2) # 22283    42 same as before
boxplot(ex2, main = 'Boxplot of the data after applying a logarithmic transformation',
        xlab = "Samples",
        ylab = "Expression Value"
        )
```

From the boxplot after the log transformation, I can see that there is some variation in the median of the samples. So, I try to apply a median normalization to the data after the log transformation.

```
##### FIIIIXXXXX
# MEDIAN NORMALIZATION
normalized.log.ex=scale(log(ex))

# boxplot post median normalization on ex
boxplot(normalized.log.ex,
        main = 'Boxplot of the data after median normalization',
        xlab = "Samples",
        ylab = "Expression Value")
```

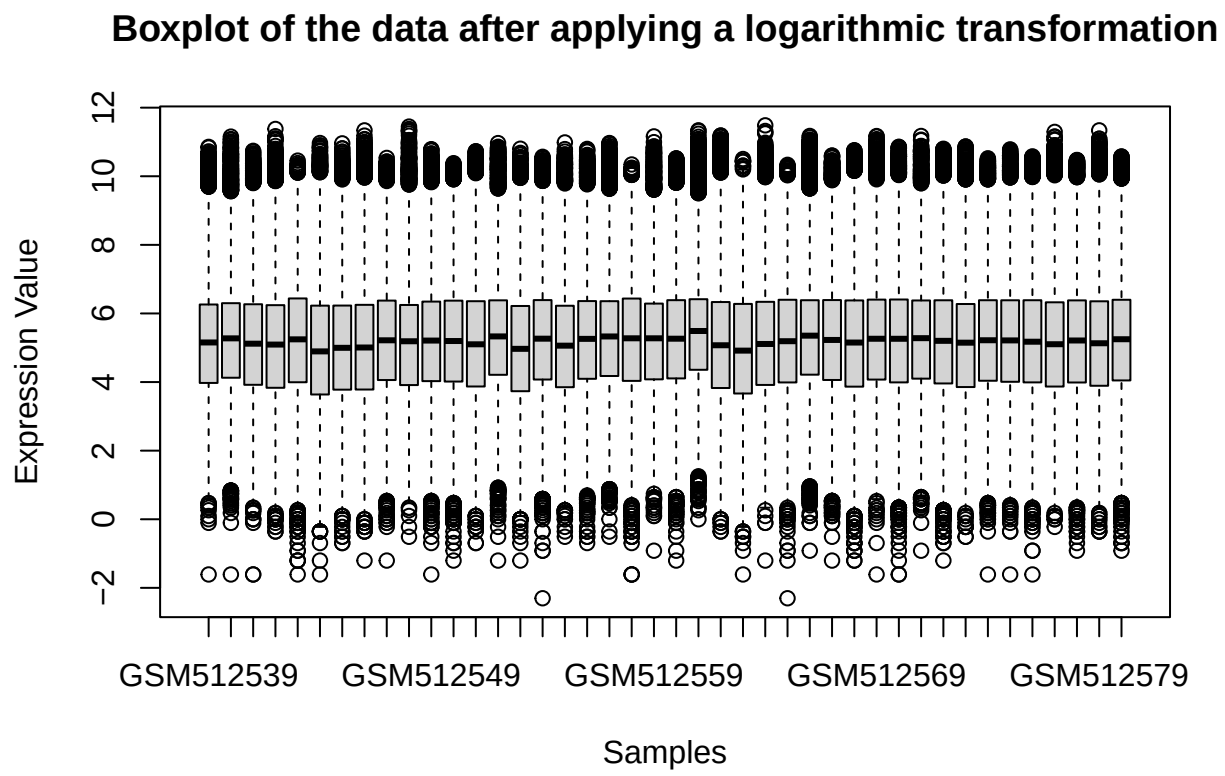
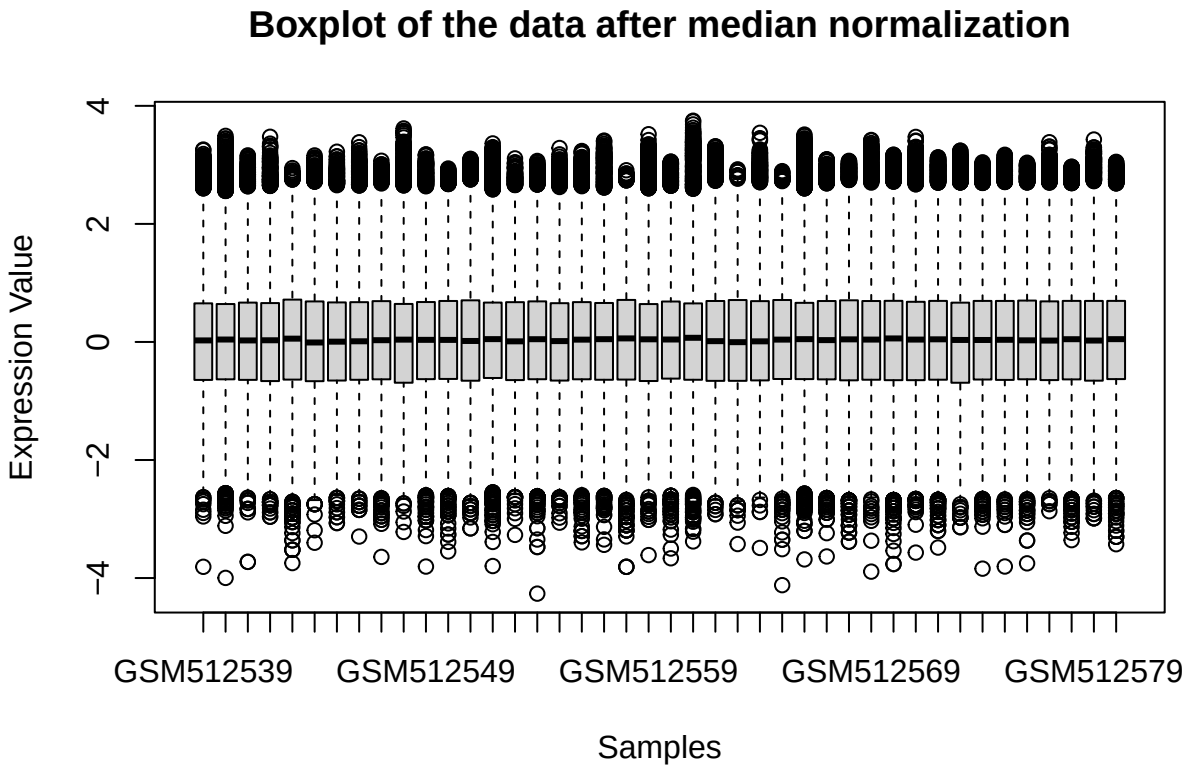


Figure 1: Boxplot of the data after applying a logarithmic transformation




```
# MEDIAN NORMALIZATION
channel.medians=apply(log(ex),2,median)
```

```r
normalized.log.ex=sweep(log(ex),2,channel.medians,"-")

# boxplot post median normalization on ex
boxplot(normalized.log.ex,
        main = 'Boxplot of the data after median normalization',
        xlab = "Samples",
        ylab = "Expression Value")
```

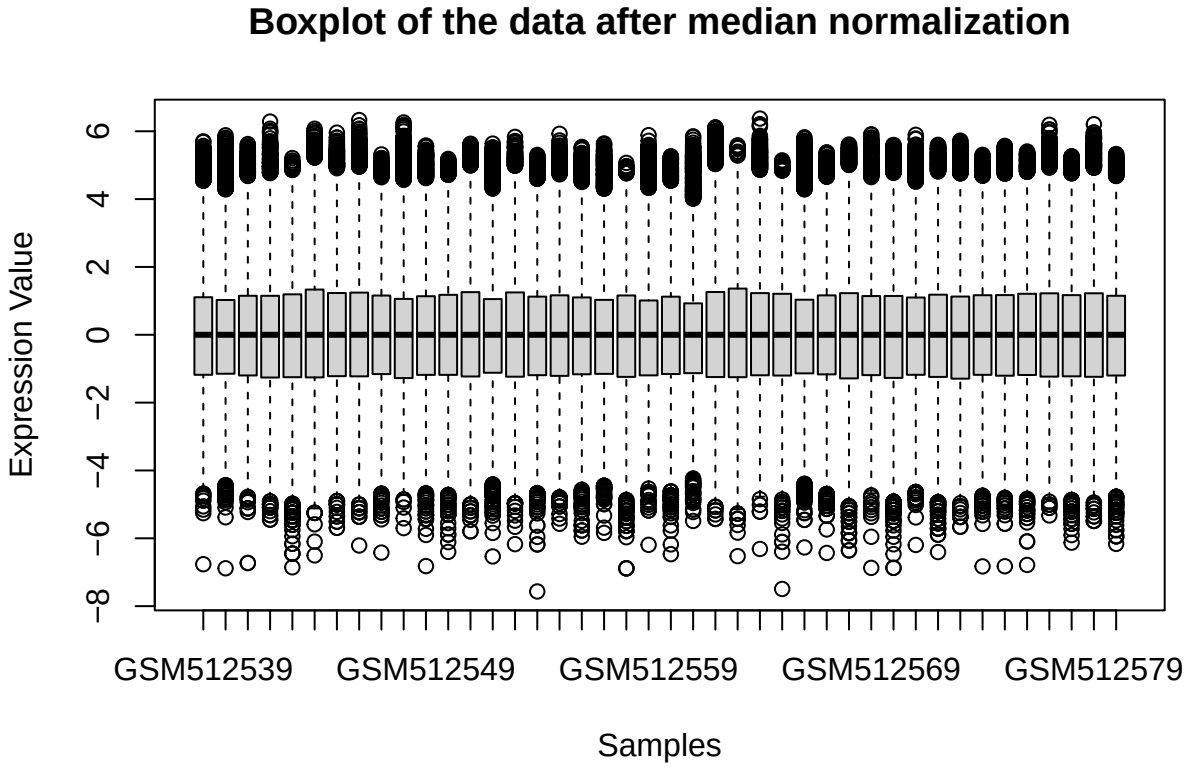


Figure 2: Boxplot of the data after median normalization

PCA

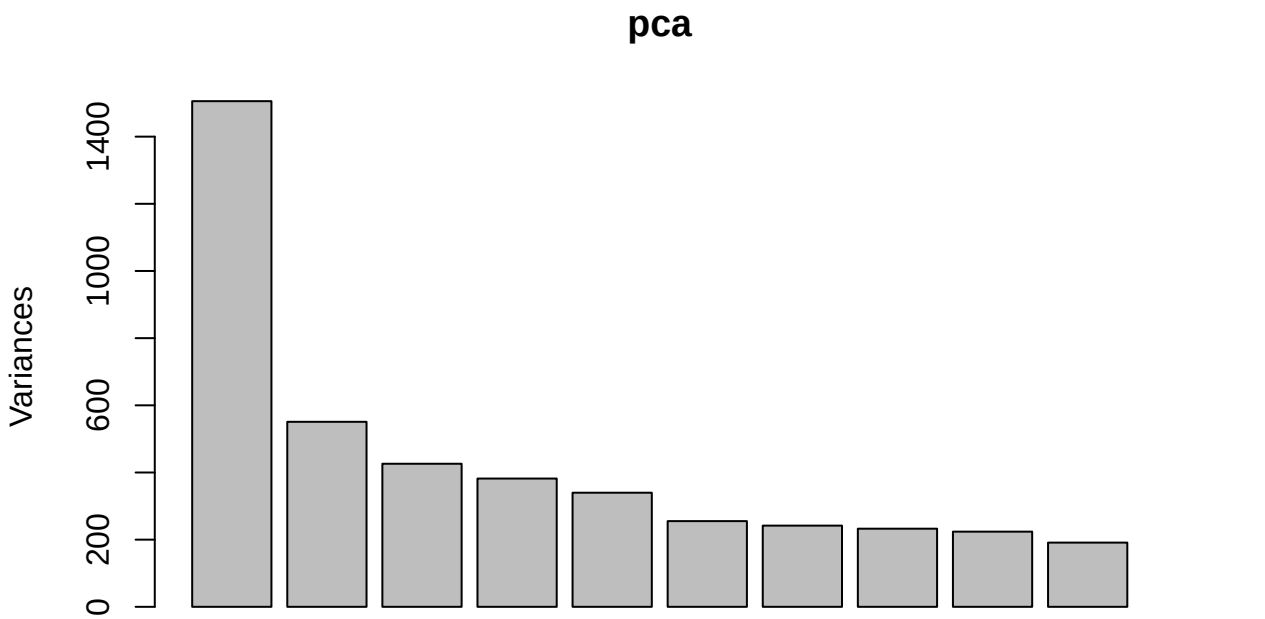
PCA is a dimensionality reduction technique that allows to condense thousands of dimensions into just two or three. For the dataset's samples, the PCA scores display the coordinates in relation to these additional dimensions.

```r
pca <- prcomp(t(normalized.log.ex))

summary(pca)
```

```
## Importance of components:
##                             PC1      PC2      PC3      PC4      PC5      PC6      PC7      PC8       PC
## Standard deviation     38.8020 23.47065 20.63892 19.54327 18.43330 15.97431 15.54787 15.25339 14.9598
## Proportion of Variance  0.1791  0.06552  0.05067  0.04543  0.04042  0.03035  0.02875  0.02767  0.0266
## Cumulative Proportion   0.1791  0.24461  0.29527  0.34070  0.38112  0.41147  0.44023  0.46790  0.4945
##                            PC11     PC12     PC13     PC14     PC15     PC16     PC17     PC18      PC
## Standard deviation     13.67761 13.63388 13.39954 13.05683 12.96288 12.78929 12.64910 12.48152 12.326
## Proportion of Variance  0.02225  0.02211  0.02136  0.02028  0.01999  0.01946  0.01903  0.01853  0.018
## Cumulative Proportion   0.53951  0.56162  0.58298  0.60326  0.62324  0.64270  0.66173  0.68026  0.698
```

```
##                            PC21     PC22     PC23     PC24     PC25     PC26     PC27     PC28      PC
## Standard deviation     12.04722 11.97088 11.90374 11.69483 11.60734 11.57222 11.46507 11.15870 11.049
## Proportion of Variance  0.01726  0.01705  0.01685  0.01627  0.01603  0.01593  0.01564  0.01481  0.014
## Cumulative Proportion   0.73312  0.75017  0.76702  0.78329  0.79932  0.81525  0.83088  0.84569  0.860
##                            PC31     PC32     PC33     PC34    PC35     PC36     PC37     PC38    PC39     P
## Standard deviation     10.68517 10.45191 10.25430 10.13818 9.9583 9.68456 9.50070 9.42259 9.3053 9.17
## Proportion of Variance  0.01358  0.01299  0.01251  0.01223 0.0118 0.01116 0.01074 0.01056 0.0103 0.01
## Cumulative Proportion   0.88818  0.90117  0.91368  0.92591 0.9377 0.94886 0.95960 0.97016 0.9805 0.99
##                              PC42
## Standard deviation     7.039e-14
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

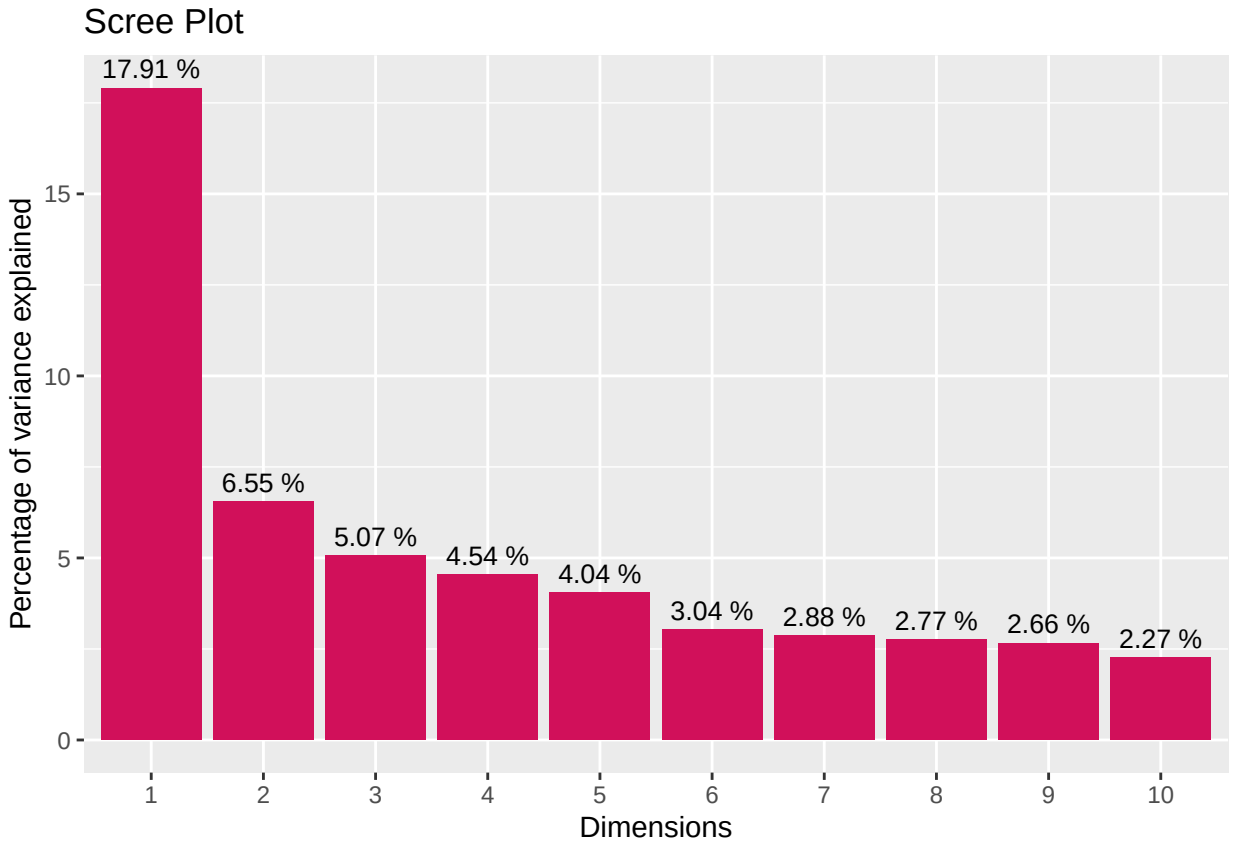
```
screeplot(pca)
```



To get the summary of the PCA and the plot showing the variance explained by the first 10 components, it is possible to use the functions commented in the chunks above.

However, using `ggplot2` and `factoextra` packages is possible to get a more concise and informative plot reporting the same information.

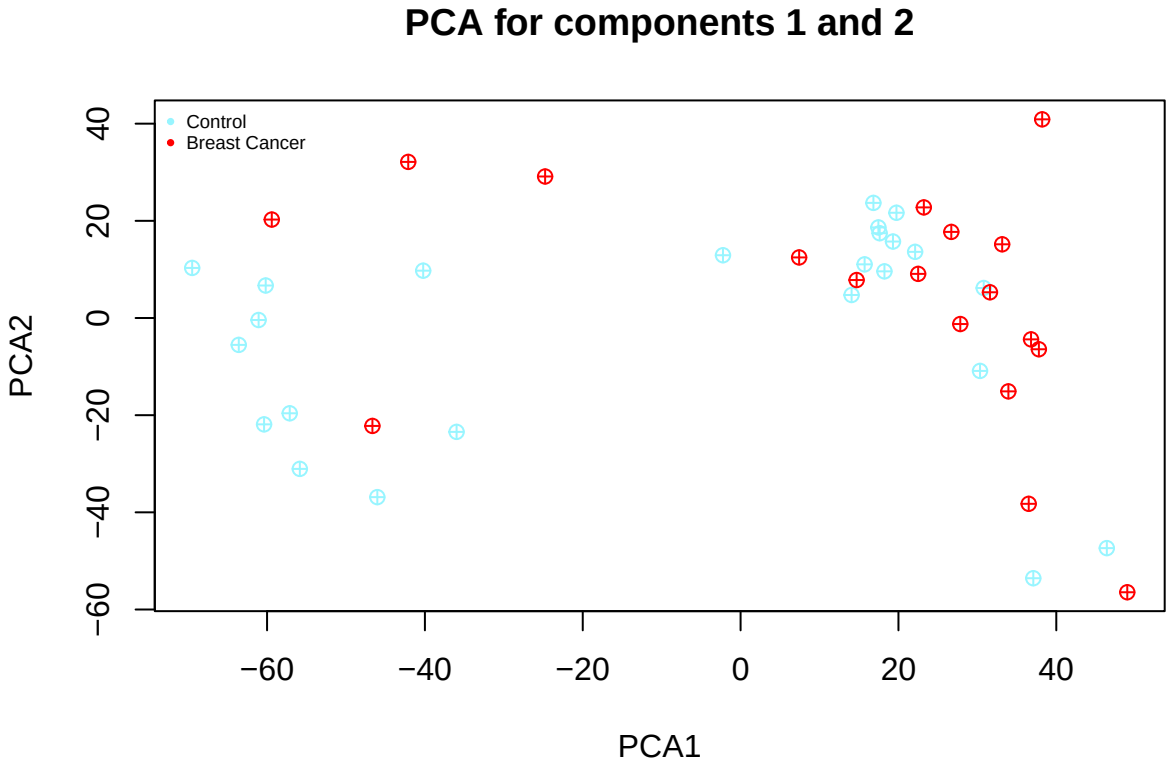
```
pcaVar <- get_eig(pca)
pcaVar <- pcaVar$variance.percent[1:10]
screeDf <- data.frame("Dimensions" = as.factor(seq(1,10)),
                      "Percentages" = pcaVar,
                      "Labels" = paste(round(pcaVar, 2), "%"))

p <- ggplot(data = screeDf, aes(x=Dimensions, y=Percentages))+
  geom_bar(stat = "identity", fill = "#d1105a")+
  geom_text(aes(label=Labels), vjust=-0.5, color="black", size=3.6)+
  ggtitle("Scree Plot")+
  ylab("Percentage of variance explained")+
  scale_x_discrete(labels = as.factor(seq(1,10)))
p
```



The scree plot shows that the first dimensions on the left are the more important because the percentage of variance explained by them is higher. The remaining principal components account for a very small proportion of the variability and are probably unimportant.

Let's try to plot the PCA, looking if we can see a separation between Control and Breast Cancer groups.

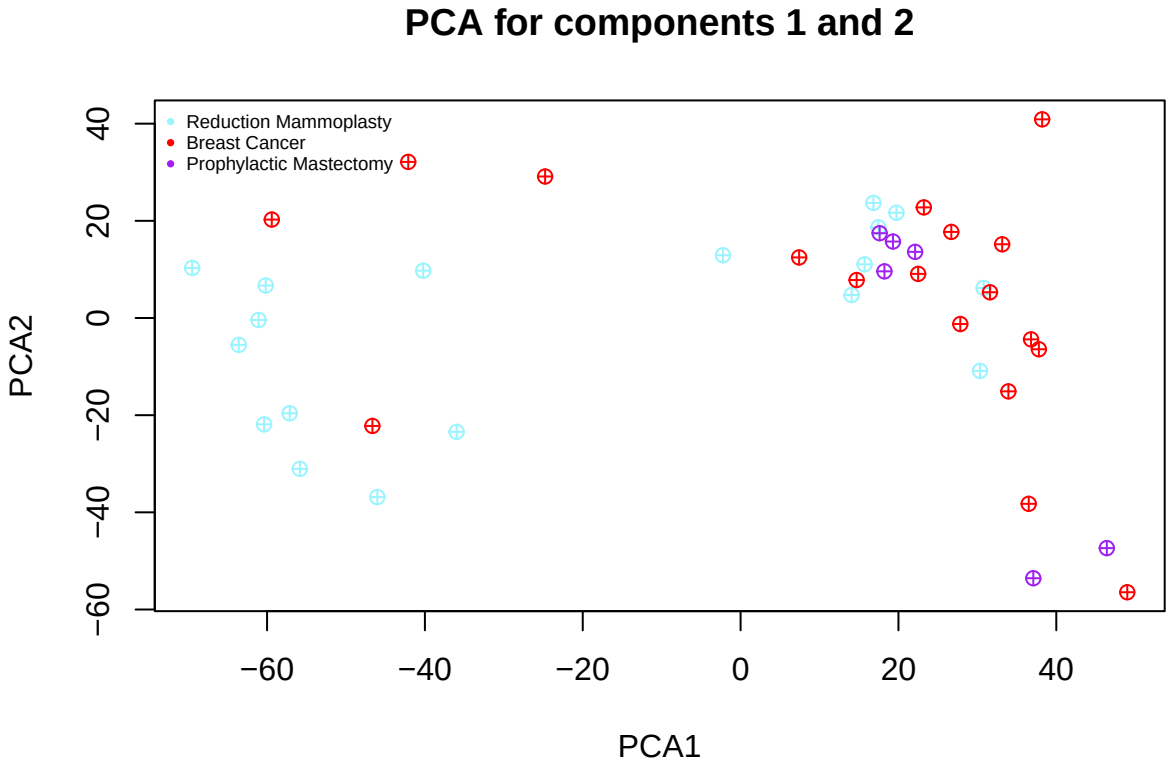
```
# draw PCA plot control VS breast cancer
group <- c(rep("cadetblue1",18), rep("red",18), rep("cadetblue1",6) )
plot(pca$x[,1], pca$x[,2], xlab="PCA1", ylab="PCA2", main="PCA for components 1 and 2", type="p", pch=1
text(pca$x[,1], pca$x[,2], rownames(pca$data), cex=0.75)
legend("topleft", col=c("cadetblue1","red"), legend = c("Control", "Breast Cancer"),
    pch = 20, bty='n', cex=.55)
```



Let's try to add the control subtypes. The vector group used in the PCA plot is based on the data. The samples corresponding to the colors are the following:

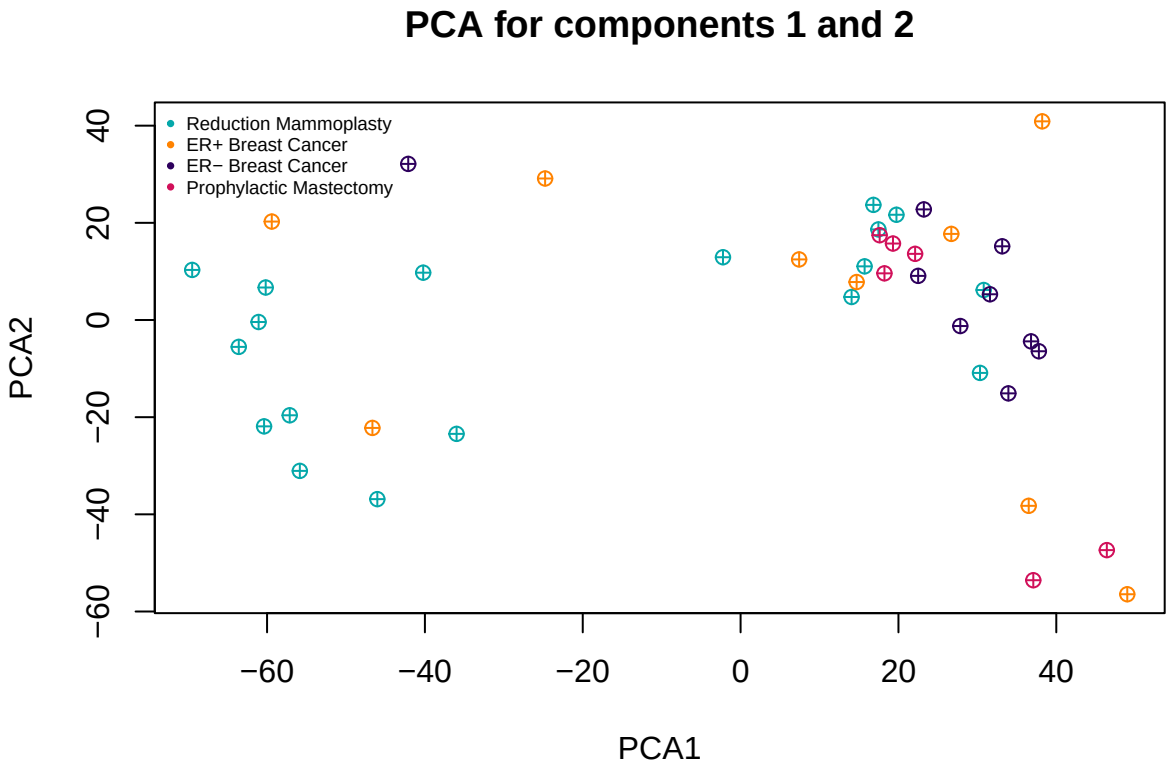
- **Light blue**: reduction mammoplasty (RM) breast epithelium samples
- **Red**: histologically normal (HN) epithelial samples from breast cancer patient
- **Purple**: histologically normal breast epithelium (NlEpi) from prophylactic mastectomy patient samples

```
# draw PCA plot
group <- c(rep("cadetblue1",18), rep("red",18), rep("purple",6) ) # vector of colors based on the order
plot(pca$x[,1], pca$x[,2], xlab="PCA1", ylab="PCA2", main="PCA for components 1 and 2", type="p", pch=1
text(pca$x[,1], pca$x[,2], rownames(pca$data), cex=0.75)
legend("topleft", col=c("cadetblue1","red","purple"), legend = c("Reduction Mammoplasty", "Breast Cance
    pch = 20, bty='n', cex=.55)
```



Then, I try to see if there is a separation also inside different types of Breast Cancer.

```
# draw PCA plot with all subtypes
group <- c(rep(my_colors[7],18), rep(my_colors[4],9), rep(my_colors[1],9), rep(my_colors[6],6) ) # vect
plot(pca$x[,1], pca$x[,2], xlab="PCA1", ylab="PCA2", main="PCA for components 1 and 2", type="p", pch=1
text(pca$x[,1], pca$x[,2], rownames(pca$data), cex=0.75)
legend("topleft", col=c(my_colors[7],my_colors[4],my_colors[1],my_colors[6]), legend = c("Reduction Mamm
    pch = 20, bty='n', cex=.55)
```



Interactive PCA plot

Let's try to explore an interactive PCA plot.

```
components<-pca[["x"]]
components<-data.frame(components)
type<-c(rep("RM", 18), rep("HN",18), rep("NlEpi",6))
components<-cbind(components, type )

fig <- plot_ly(components, x=~PC1, y=~PC2,
               color=type,colors=c('cadetblue1', 'red','purple'),
               type='scatter',mode='markers')
fig
```

```
fig2 <- plot_ly(components, x=~PC1, y=~PC2, z=~PC3,
                color=type, colors=c('cadetblue1', 'red','purple'),
                mode='markers', marker = list(size = 4))
fig2
```

```
fig3 <- plot_ly(components, x=~PC1, y=~PC3,
                color=type, colors=c('cadetblue1', 'red','purple'),
                type='scatter',mode='markers')
fig3
```

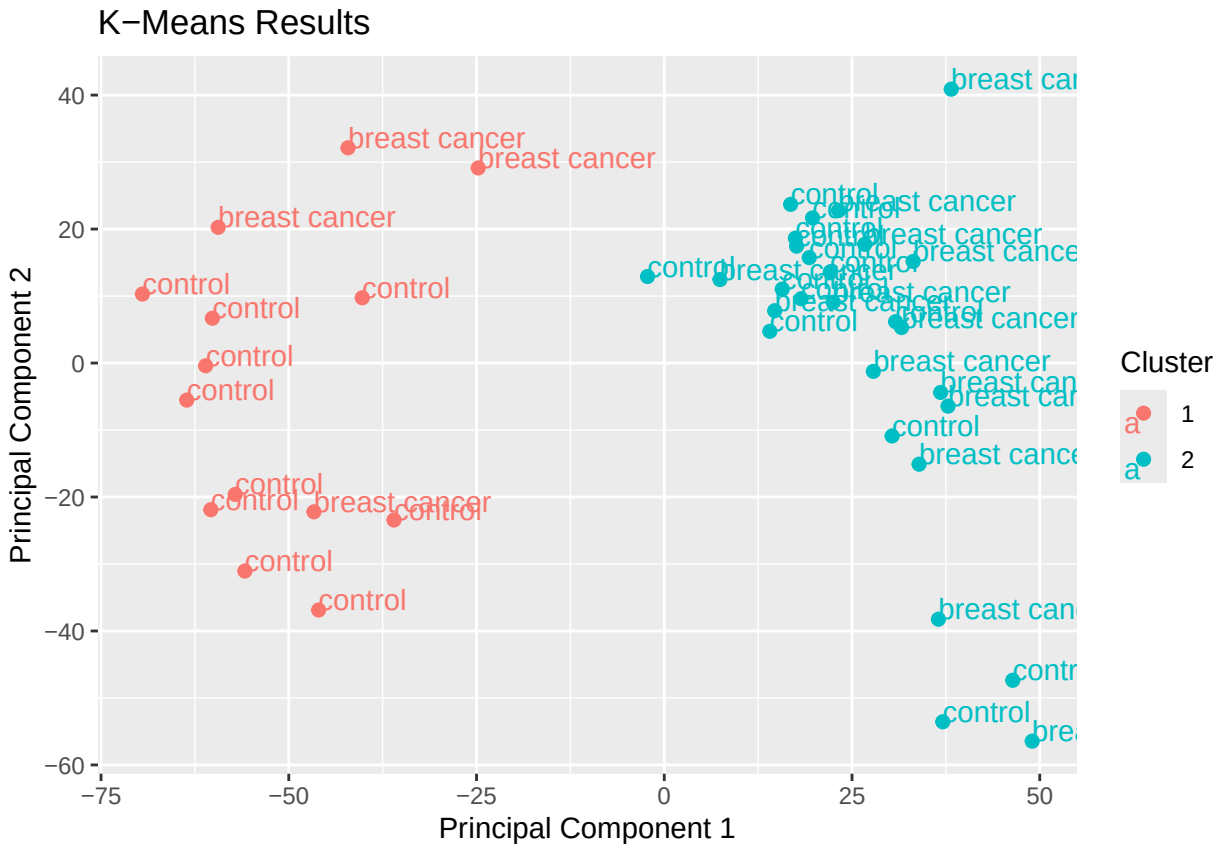
Clustering

K-means

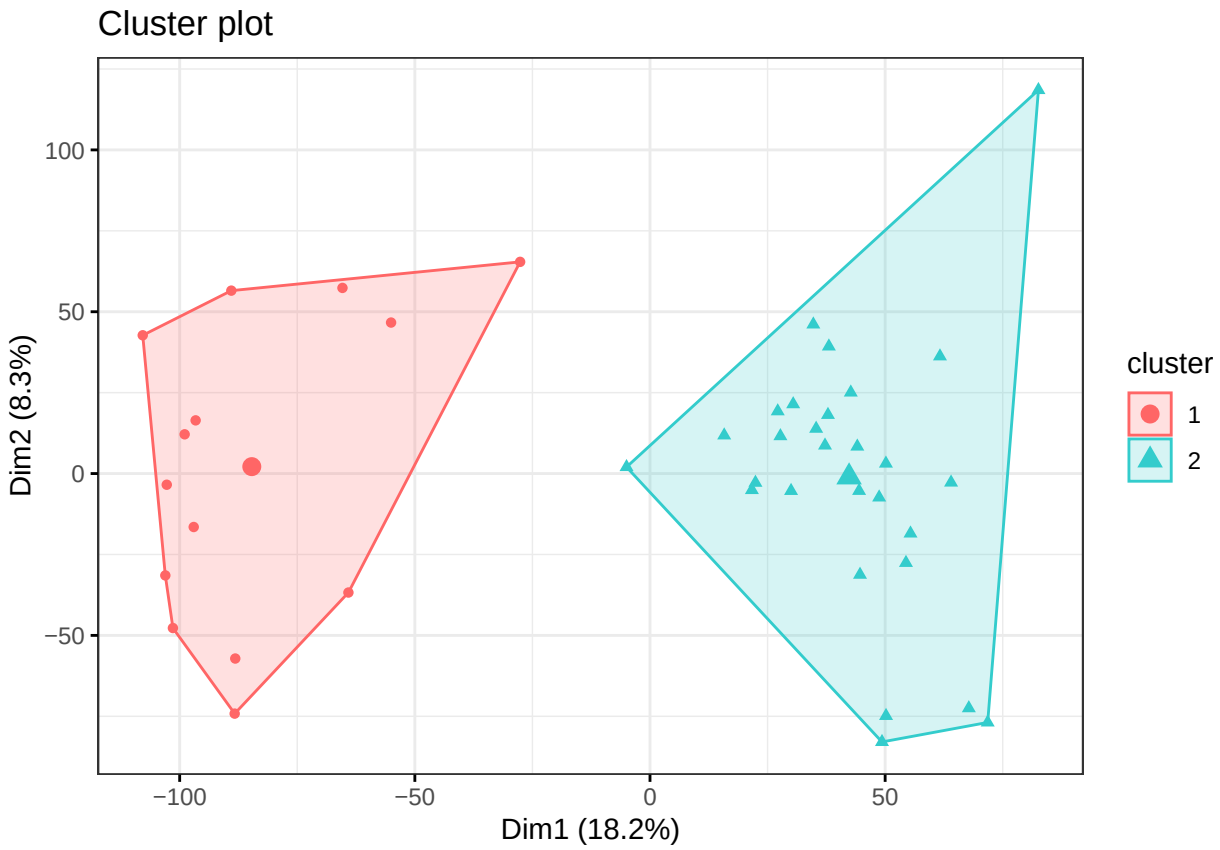
```r
set.seed(1)
k <- 2 # number of clusters

kmeans_result <- kmeans(t(normalized.log.ex),k)
table(kmeans_result$cluster) # tells how many samples were assigned to each cluster
```

```
##
##  1  2
## 14 28
```

```r
plot(kmeans_result, data=t(normalized.log.ex)) + geom_text(aes(label=metadata$disease.state.ch1),hjust=0
```




```r
fviz_cluster(kmeans_result, data = t(normalized.log.ex),
             palette = c("#FF6666", "#33cccc"),
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw()
             )
```

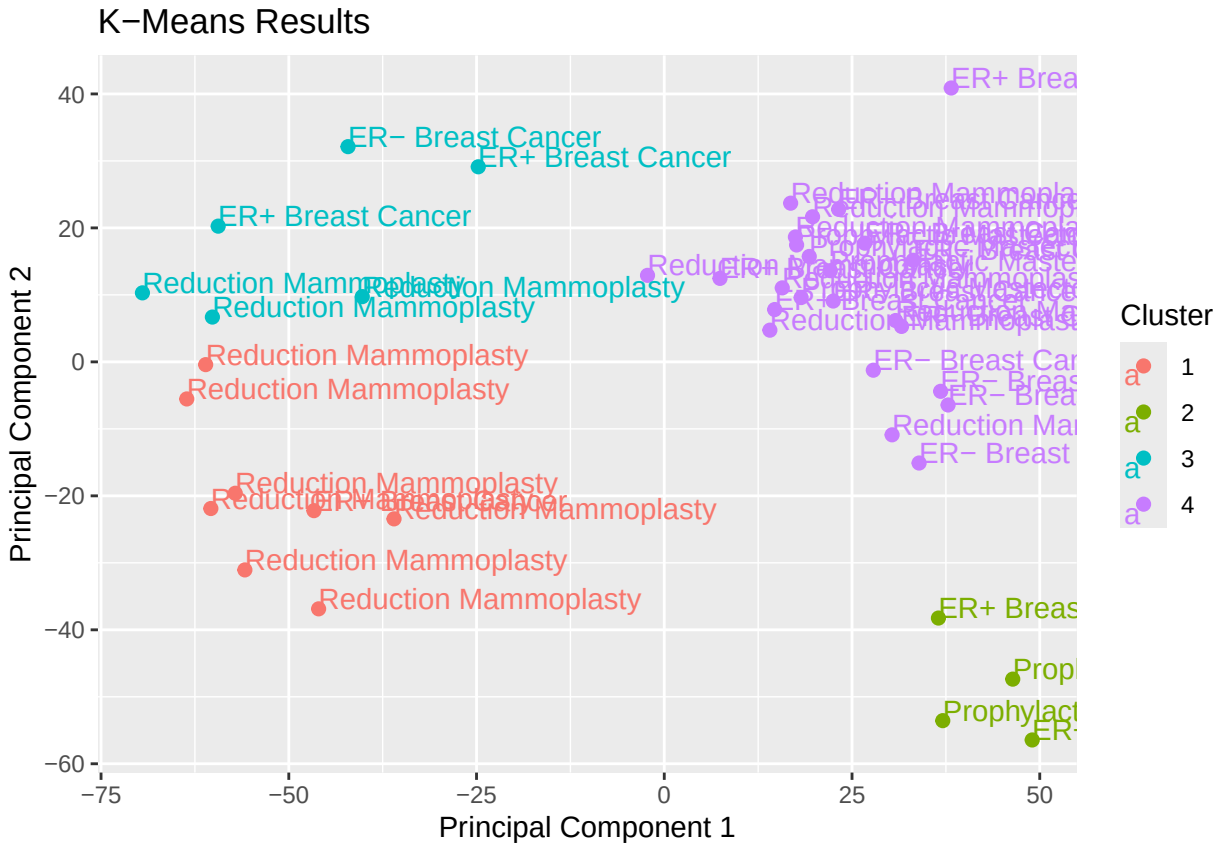


Let's try increasing the number of clusters.

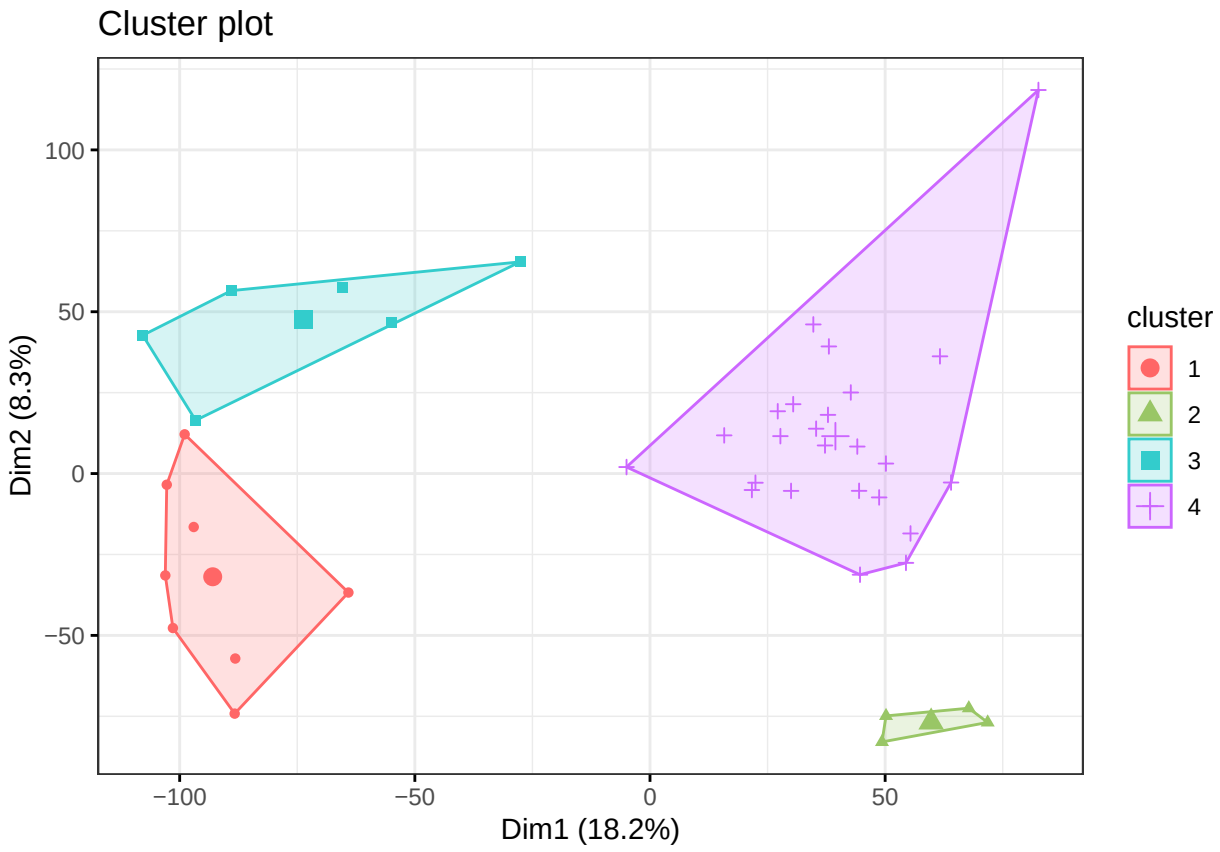
```
set.seed(1)
k <- 4 # number of clusters

kmeans_result <- kmeans(t(normalized.log.ex),k)
table(kmeans_result$cluster) # tells how many samples were assigned to each cluster
```

```
##
##  1  2  3  4
##  8  4  6 24
```

```
plot(kmeans_result, data=t(normalized.log.ex)) + geom_text(aes(label=metadata$specimen.ch1),hjust=0,vjus
```

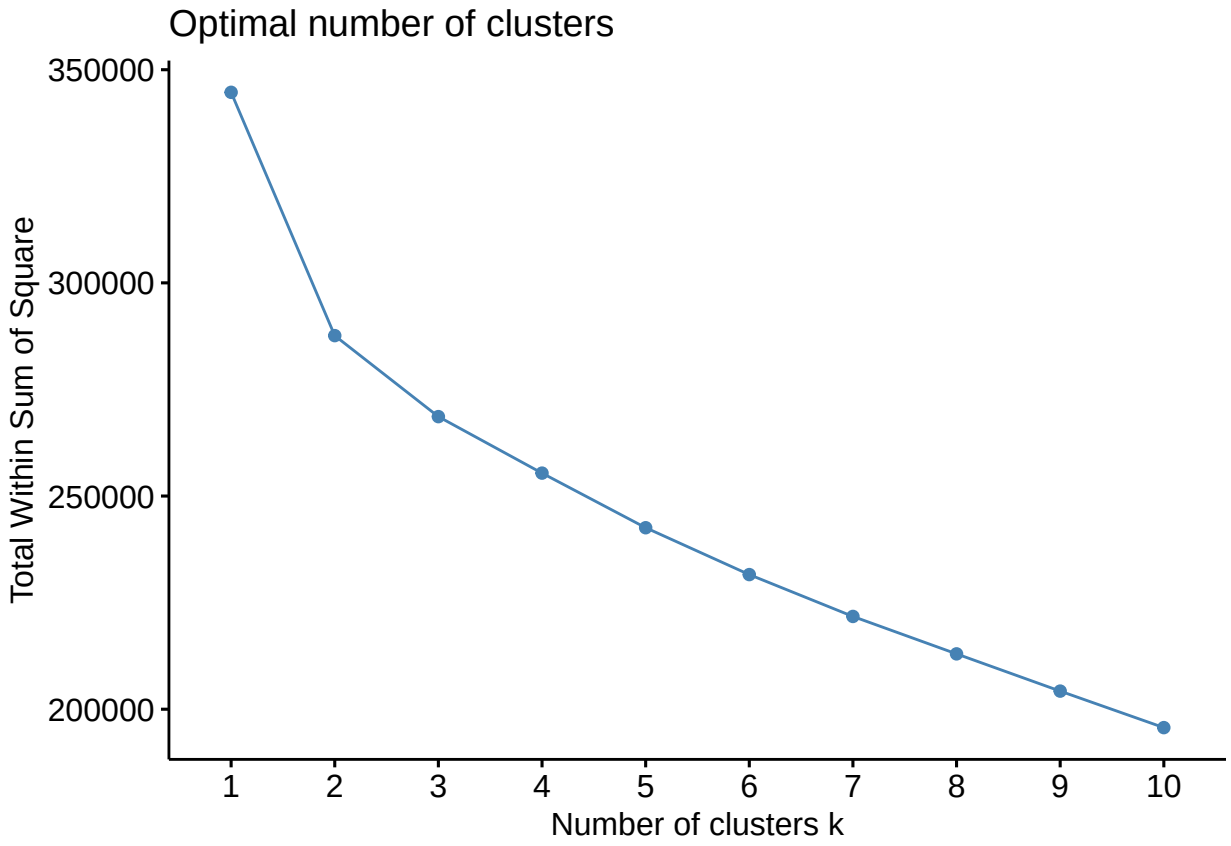



```
fviz_cluster(kmeans_result, data = t(normalized.log.ex),
             palette = c("#FF6666", "#99C666", "#33cccc", "#cc66ff"),
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw()
             )
```



Hierarchical

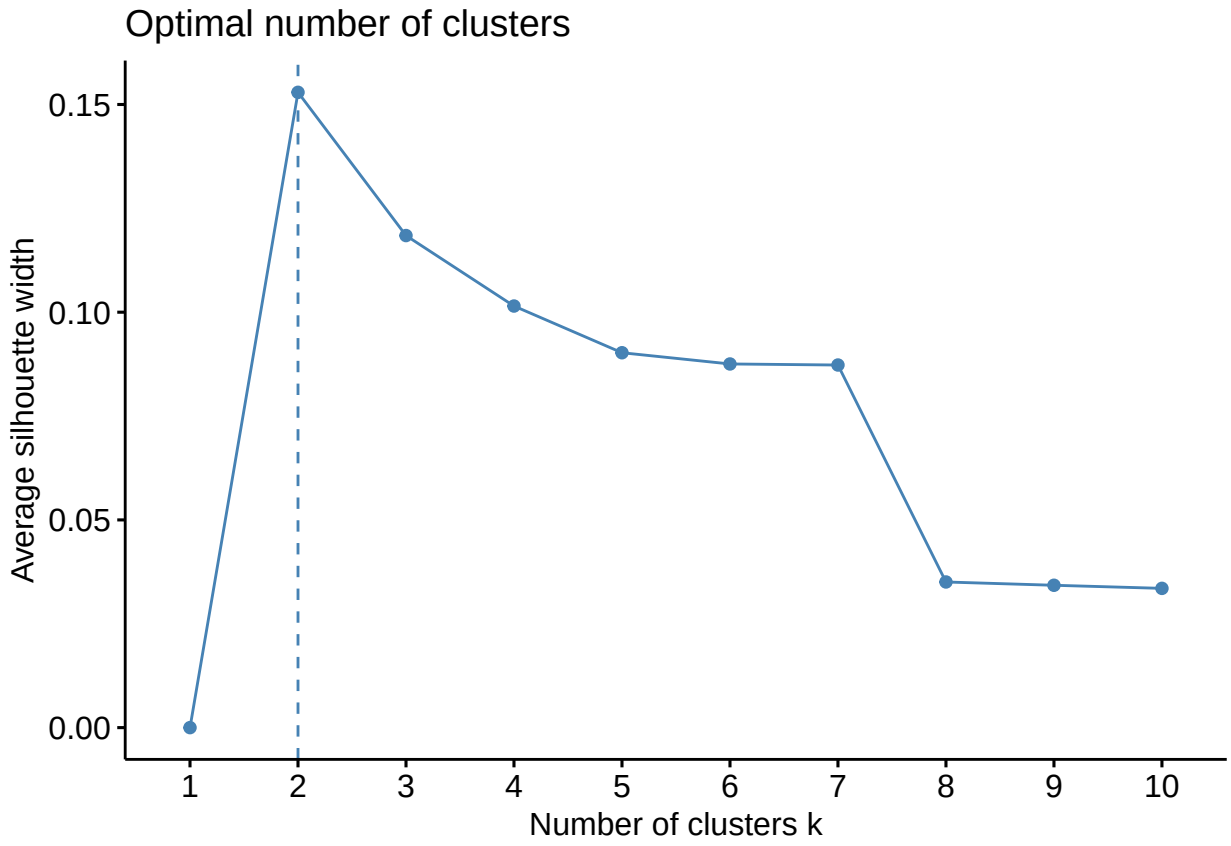
```
# Elbow method
fviz_nbclust(t(normalized.log.ex), FUN = hcut, method = "wss")
```



FIX DA QUA

```
## seems setting number of clusters equal to 2
```

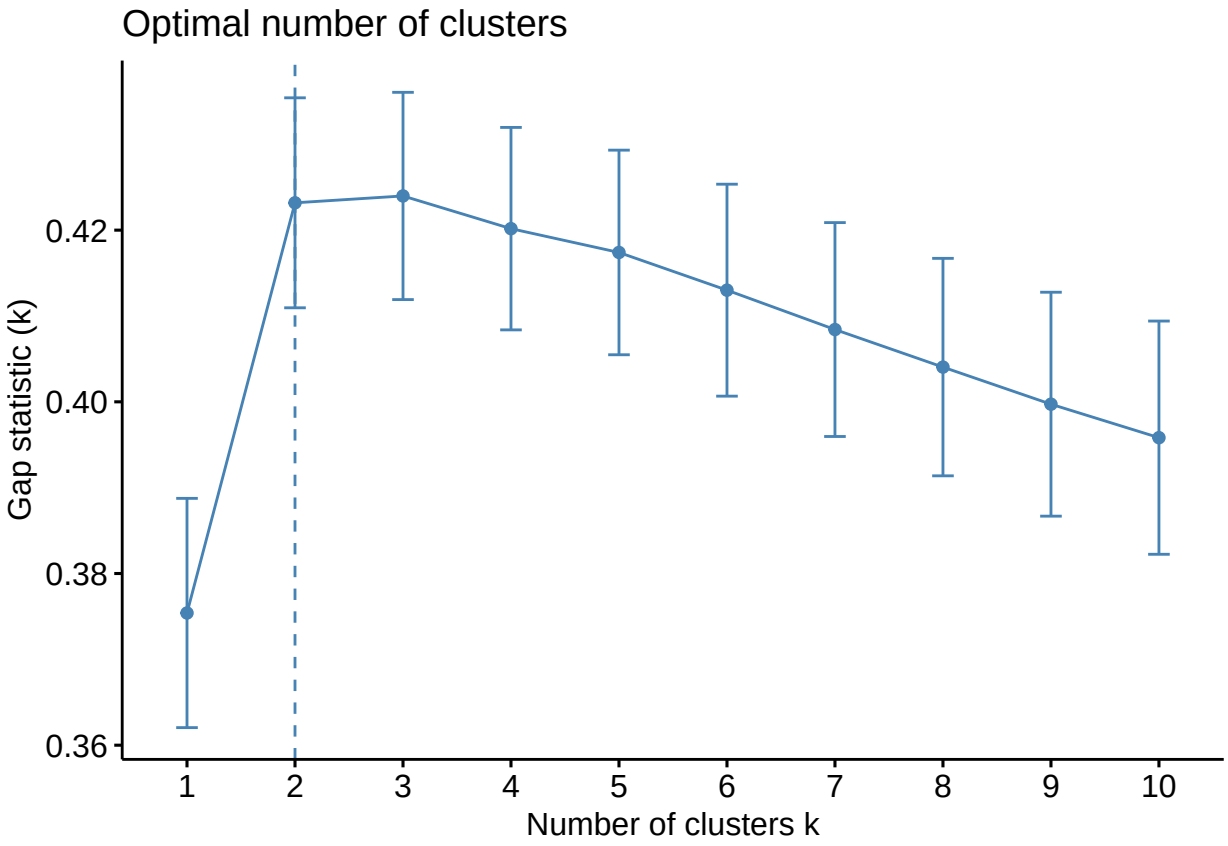
```
# Silhouette method
fviz_nbclust(t(normalized.log.ex), FUN = hcut, method = "silhouette")
```




```
## seems setting number of clusters equal to 2
```

We use the Gap statistic to calculate the goodness of clustering.

```
# Gap Statistic Method
gap_stat <- clusGap(t(normalized.log.ex), FUN = hcut, nstart = 25, K.max = 10, B = 50)
# K.max -> the maximum number of clusters to consider
# B -> number of Monte Carlo samples

fviz_gap_stat(gap_stat)
```




```
hc_result <- dist(t(normalized.log.ex)) %>% hclust(method = "ave")
hc_result2<- dist(t(normalized.log.ex), method="euclidean") %>% hclust( method = "complete")
hc_result3 <- dist(t(normalized.log.ex)) %>% hclust(method = 'single')

k_hc <- 2

groups <- cutree(hc_result, k=k_hc)
table(groups,type)
```

```
##       type
## groups HN NlEpi RM
##      1  4     0 10
##      2 14     6  8
```

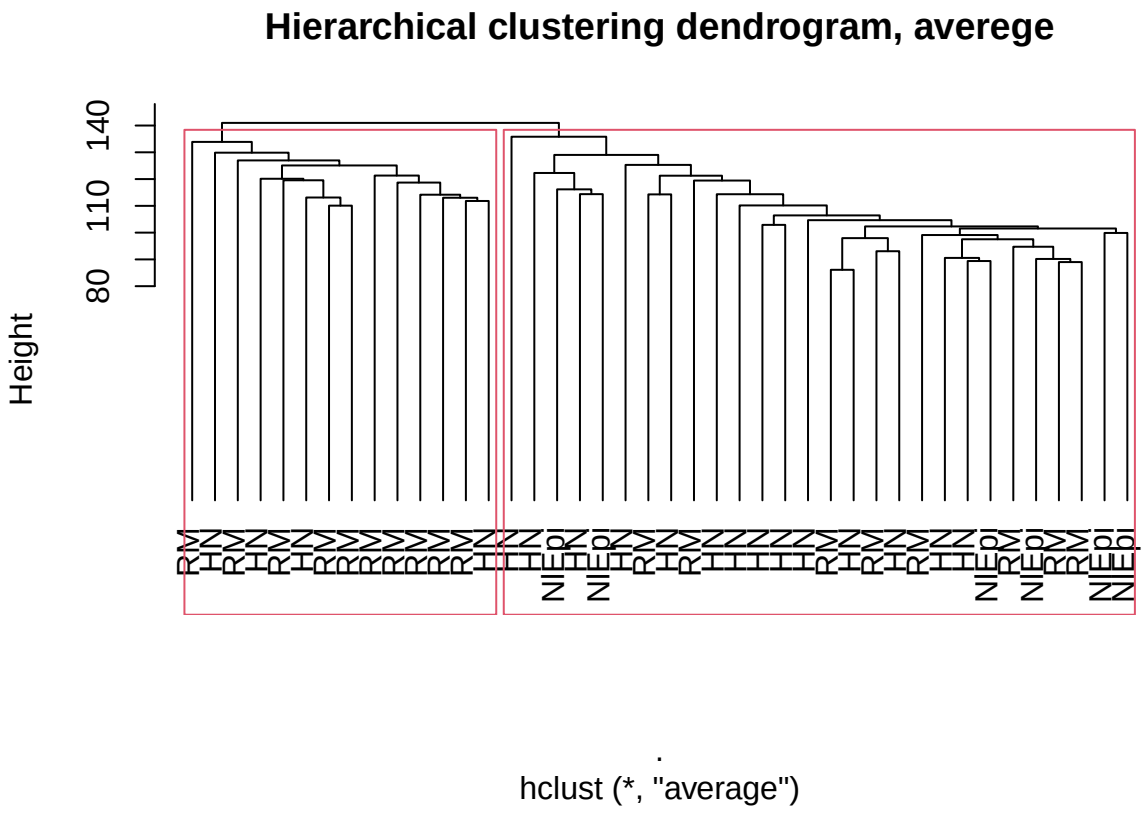
```
groups2<-cutree(hc_result2, k=k_hc)
table(groups2,type)
```

```
##        type
## groups2 HN NlEpi RM
##       1  4     0 10
##       2 14     6  8
```

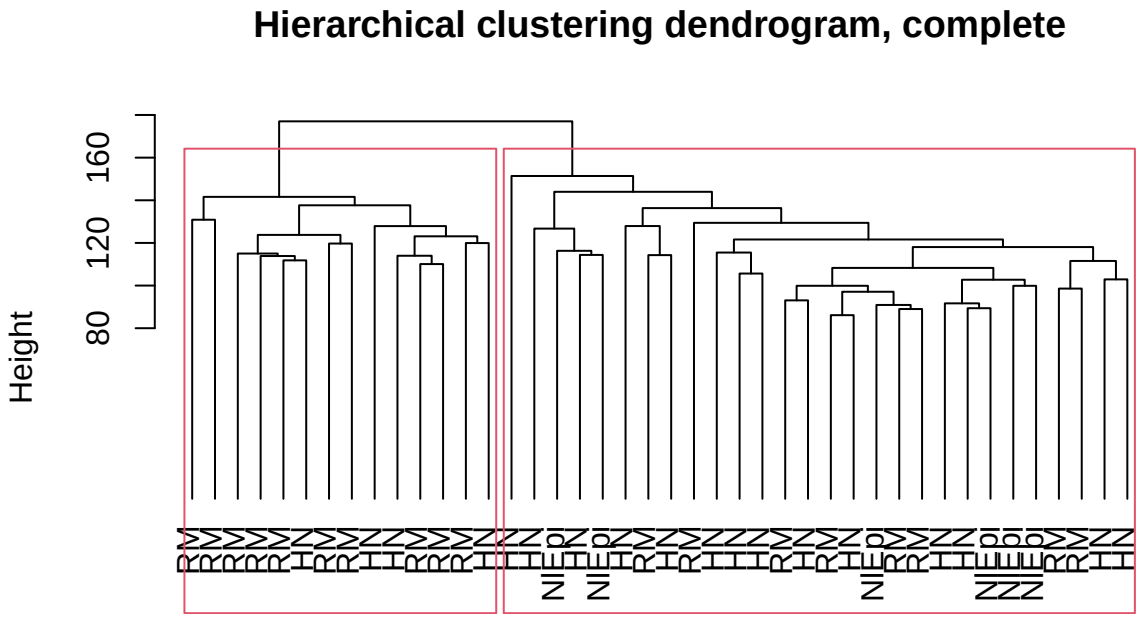
```
groups3 <- cutree(hc_result3,k=k_hc)
table(groups3,type)
```

```
##        type
## groups3 HN NlEpi RM
##       1 18     6 17
##       2  0     0  1
```

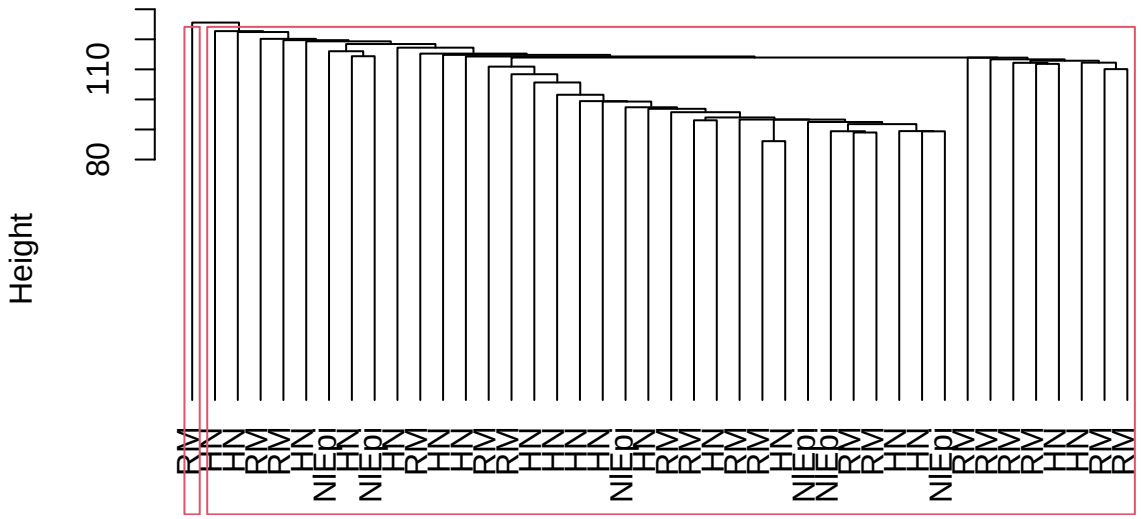
```
plot(hc_result, hang <- -1, labels=type, main = 'Hierarchical clustering dendrogram, averege')
rect.hclust(hc_result, k = 2, which = NULL, x = NULL, h = NULL, border = 2, cluster = NULL) # red boxes
```




```
plot(hc_result2, hang <- -1, labels=type, main = 'Hierarchical clustering dendrogram, complete')
rect.hclust(hc_result2, k = 2, which = NULL, x = NULL, h = NULL, border = 2, cluster = NULL) # red boxes
```

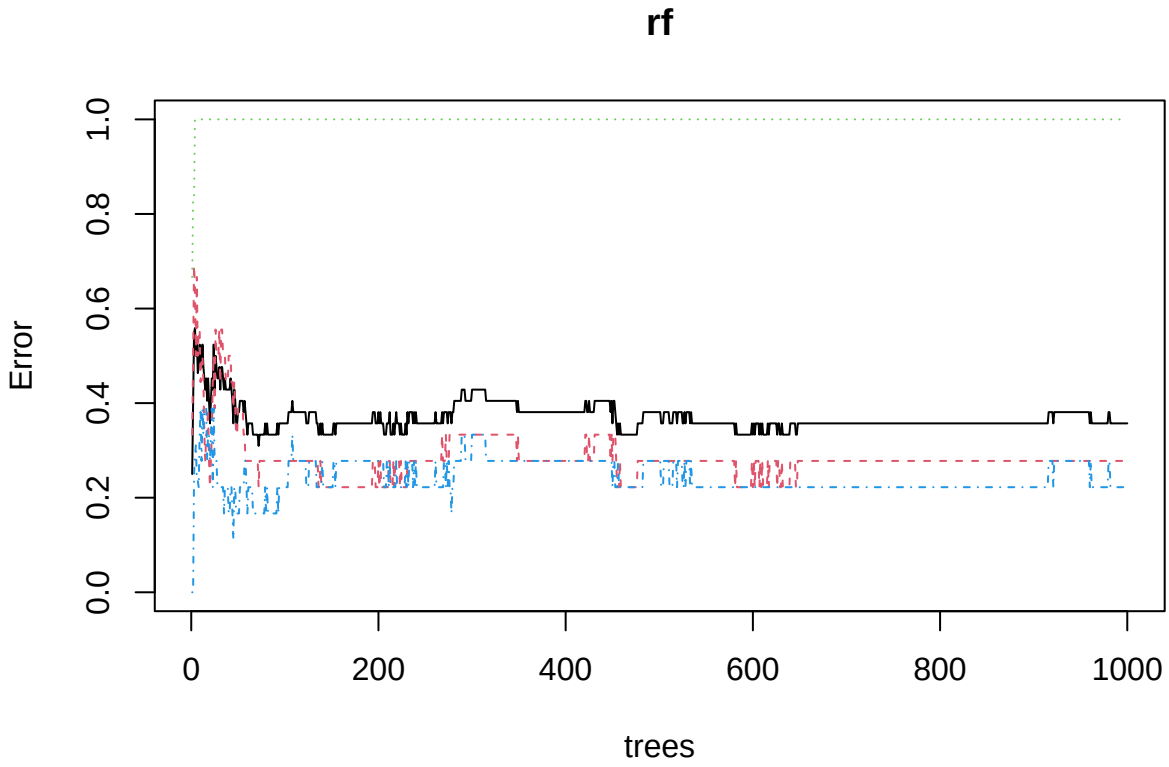



```
plot(hc_result3, hang <- -1, labels=type, main = 'Hierarchical clustering dendrogram, single')
rect.hclust(hc_result3, k = 2, which = NULL, x = NULL, h = NULL, border = 2, cluster = NULL) # red boxe
```



Random forest

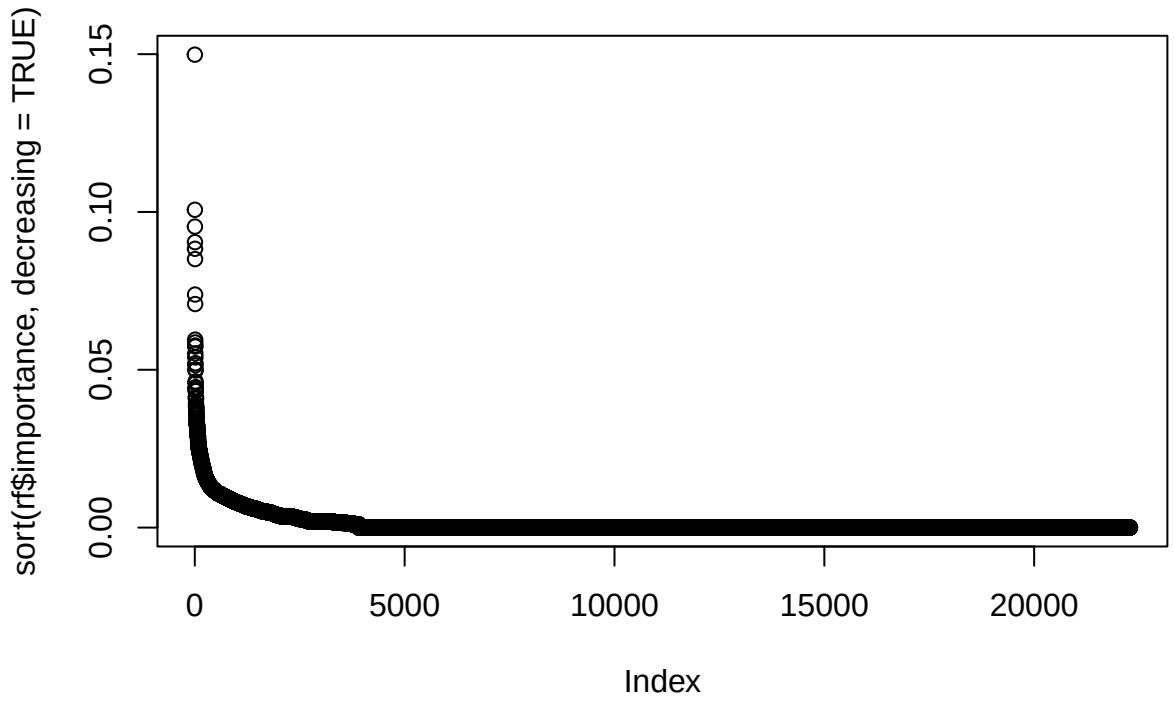
```
set.seed(1234)
rf <- randomForest(x=t(normalized.log.ex), y=as.factor(type), ntree=1000)

plot(rf)
```




```
# a trivial test
predict(rf, t(normalized.log.ex[, 1:5]))
```

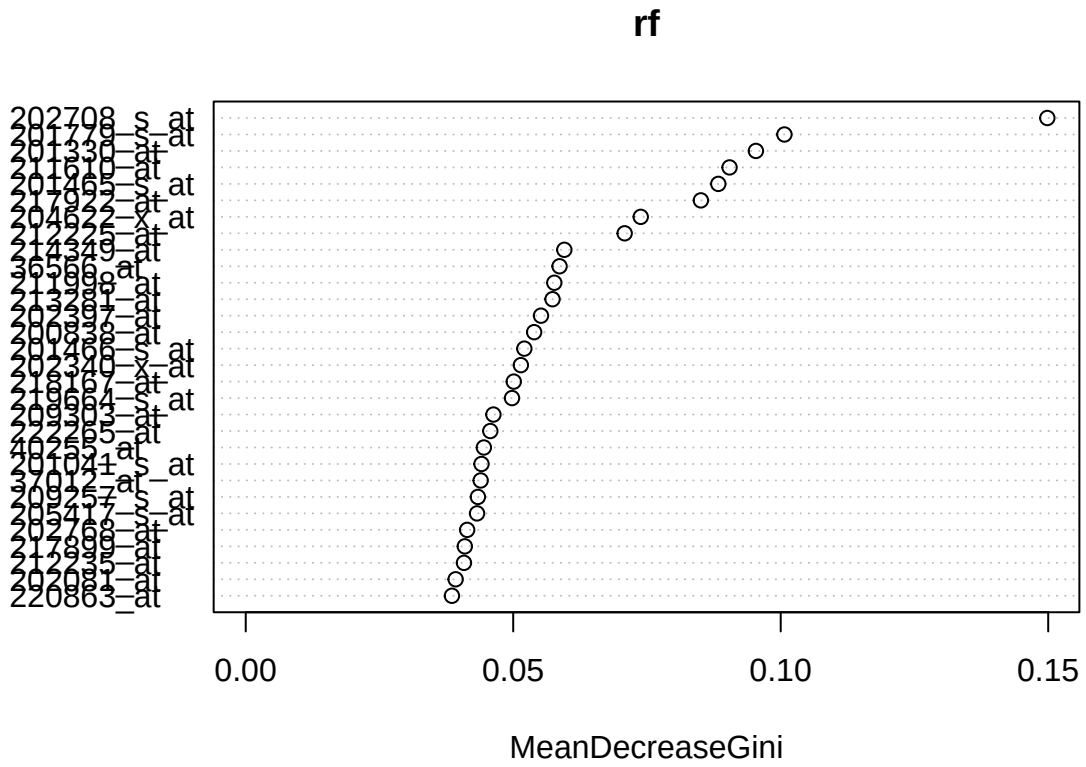
```
## GSM512539 GSM512540 GSM512541 GSM512542 GSM512543
##        RM        RM        RM        RM        RM
## Levels: HN NlEpi RM
```

```
# graph of sorted importance values

plot(sort(rf$importance, decreasing=TRUE))
```




```
#this plot goes into the report

# can also use: varImpPlot(rf)
```

```
varImpPlot(rf)
```



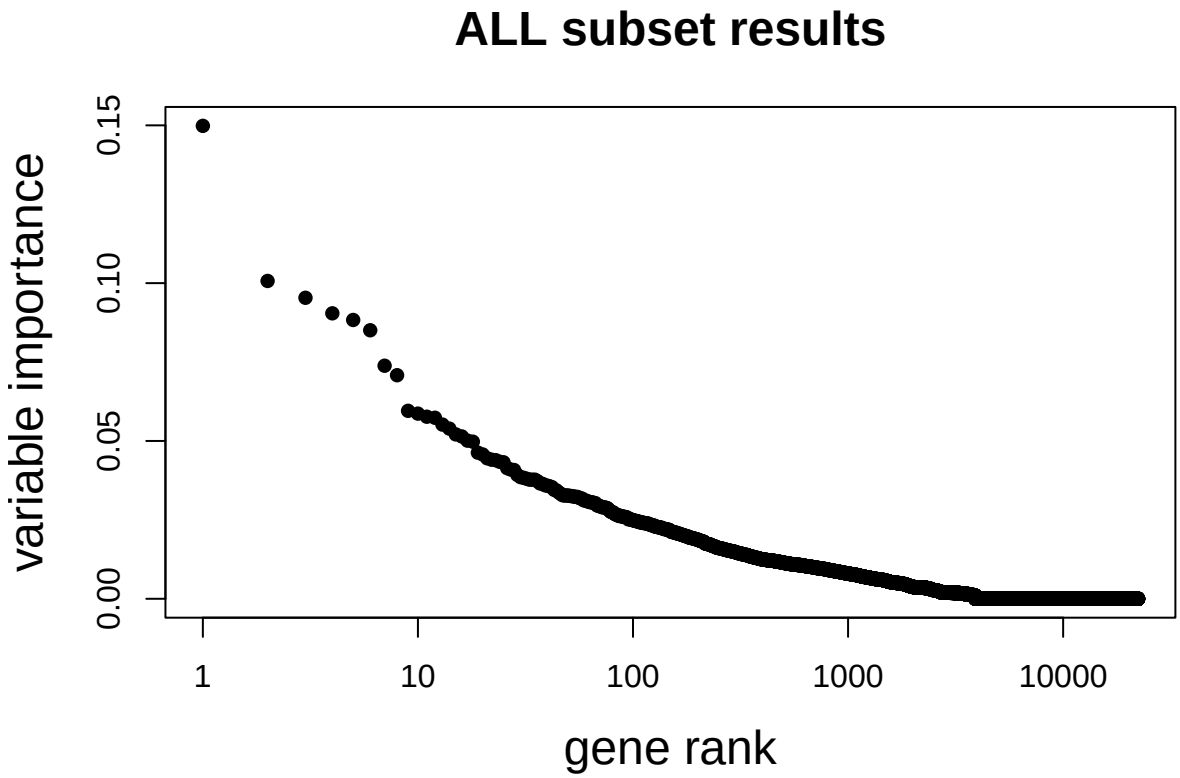

```
#extract the most 'important' genes
probe.names <- rownames(rf$importance)
```

```r
top200 <- probe.names[order(rf$importance, decreasing=TRUE)[1:200]]

write.csv(top200, file = "output/probes-top200.txt", quote=FALSE, row.names =FALSE, col.names=FALSE)
```

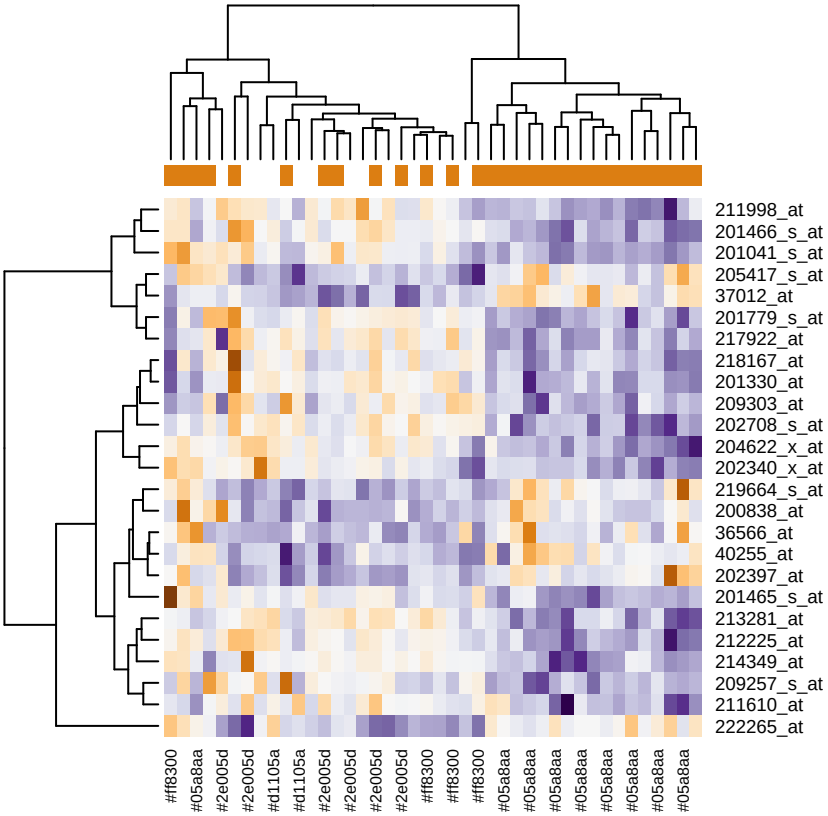
Heatmap

This is optional. Not suggested to include heatmap in the report, because at the end of the project there will be too many graphs and this is not a valuable one.

```r
# Look at variable importance
imp.temp <- abs(rf$importance[,])
t <- order(imp.temp,decreasing=TRUE)
plot(c(1:nrow(normalized.log.ex)),imp.temp[t],log='x',cex.main=1.5,
xlab='gene rank',ylab='variable importance',cex.lab=1.5,
pch=16,main='ALL subset results')
```




```r
# Get subset of expression values for 25 most 'important' genes
gn.imp <- names(imp.temp)[t]
gn.25 <- gn.imp[1:25] # vector of top 25 genes, in order
t <- is.element(rownames(normalized.log.ex),gn.25)
sig.ex <- normalized.log.ex[t,] # matrix of expression values, not necessarily in order

## Make a heatmap, with group differences obvious on plot
hmcol <- colorRampPalette(brewer.pal(11,"PuOr"))(256)
colnames(sig.ex) <- group # This will label the heatmap columns
csc <- rep(hmcol[50],30)
csc[group=='T'] <- hmcol[200]
# column side color will be purple for T and orange for B
heatmap(sig.ex, scale="row", col=hmcol, ColSideColors=csc)
```



Feature selection

```
set.seed(1)
#group <-c(rep("Reduction Mammoplasty", 18), rep("ER+ Breast Cancer",9), rep("ER- Breast Cancer", 9), r

group <- c(rep("Control",18), rep("Breast_cancer",18), rep("Control",6))
design <- model.matrix(~0+group)
colnames(design) <- c("Controls","Breast_Cancer")
rownames(design)<-colnames(exprs(gse))
#design
```

```
fit <- lmFit(exprs(gse), design)
cont.matrix <- makeContrasts(contrasts = "Breast_Cancer-Controls", levels=design)
#cont.matrix
```

```
fit2 <- contrasts.fit(fit, cont.matrix)
fit2
```

```
## An object of class "MArrayLM"
## $coefficients
##            Contrasts
##             Breast_Cancer-Controls
##   1007_s_at            -143.329167
##   1053_at                 3.704167
##   117_at                  3.179167
##   121_at                 56.700000
##   1255_g_at               4.733333
## 22278 more rows ...
```

```
##
## $rank
## [1] 2
##
## $assign
## [1] 1 1
##
## $qr
## $qr
##            Controls Breast_Cancer
## GSM512539 -4.242641     0.0000000
## GSM512540  0.000000    -4.8989795
## GSM512541  0.000000     0.2041241
## GSM512542  0.000000     0.2041241
## GSM512543  0.000000     0.2041241
## 37 more rows ...
##
## $qraux
## [1] 1.000000 1.204124
##
## $pivot
## [1] 1 2
##
## $tol
## [1] 1e-07
##
## $rank
## [1] 2
##
##
## $df.residual
## [1] 40 40 40 40 40
## 22278 more elements ...
##
## $sigma
## 1007_s_at   1053_at    117_at    121_at 1255_g_at
## 676.47958  39.20263  46.57555 186.41006  24.40483
## 22278 more elements ...
##
## $cov.coefficients
##                          Contrasts
## Contrasts                 Breast_Cancer-Controls
##   Breast_Cancer-Controls             0.09722222
##
## $stdev.unscaled
##            Contrasts
##             Breast_Cancer-Controls
##   1007_s_at              0.3118048
##   1053_at                0.3118048
##   117_at                 0.3118048
##   121_at                 0.3118048
##   1255_g_at              0.3118048
## 22278 more rows ...
##
```

```
## $pivot
## [1] 1 2
##
## $Amean
##  1007_s_at    1053_at     117_at     121_at  1255_g_at
## 3121.48095   71.58333  139.26667  803.70000   53.62143
## 22278 more elements ...
##
## $method
## [1] "ls"
##
## $design
##           Controls Breast_Cancer
## GSM512539        0             1
## GSM512540        0             1
## GSM512541        0             1
## GSM512542        0             1
## GSM512543        0             1
## 37 more rows ...
##
## $contrasts
##               Contrasts
## Levels          Breast_Cancer-Controls
##   Controls                          -1
##   Breast_Cancer                      1
```

```
fit2 <- eBayes(fit2)
fit2
```

```
## An object of class "MArrayLM"
## $coefficients
##            Contrasts
##             Breast_Cancer-Controls
##   1007_s_at            -143.329167
##   1053_at                 3.704167
##   117_at                  3.179167
##   121_at                 56.700000
##   1255_g_at               4.733333
## 22278 more rows ...
##
## $rank
## [1] 2
##
## $assign
## [1] 1 1
##
## $qr
## $qr
##            Controls Breast_Cancer
## GSM512539 -4.242641     0.0000000
## GSM512540  0.000000    -4.8989795
## GSM512541  0.000000     0.2041241
## GSM512542  0.000000     0.2041241
## GSM512543  0.000000     0.2041241
## 37 more rows ...
```

```
##
## $qraux
## [1] 1.000000 1.204124
##
## $pivot
## [1] 1 2
##
## $tol
## [1] 1e-07
##
## $rank
## [1] 2
##
##
## $df.residual
## [1] 40 40 40 40 40
## 22278 more elements ...
##
## $sigma
## 1007_s_at   1053_at    117_at    121_at 1255_g_at
## 676.47958  39.20263  46.57555 186.41006  24.40483
## 22278 more elements ...
##
## $cov.coefficients
##                          Contrasts
## Contrasts                 Breast_Cancer-Controls
##   Breast_Cancer-Controls             0.09722222
##
## $stdev.unscaled
##            Contrasts
##             Breast_Cancer-Controls
##   1007_s_at              0.3118048
##   1053_at                0.3118048
##   117_at                 0.3118048
##   121_at                 0.3118048
##   1255_g_at              0.3118048
## 22278 more rows ...
##
## $pivot
## [1] 1 2
##
## $Amean
##  1007_s_at    1053_at     117_at     121_at  1255_g_at
## 3121.48095   71.58333  139.26667  803.70000   53.62143
## 22278 more elements ...
##
## $method
## [1] "ls"
##
## $design
##           Controls Breast_Cancer
## GSM512539        0             1
## GSM512540        0             1
## GSM512541        0             1
```

```
## GSM512542        0             1
## GSM512543        0             1
## 37 more rows ...
##
## $contrasts
##                Contrasts
## Levels          Breast_Cancer-Controls
##   Controls                          -1
##   Breast_Cancer                      1
##
## $df.prior
## [1] 0.7758139
##
## $s2.prior
## [1] 1774.513
##
## $var.prior
## [1] 0.009016559
##
## $proportion
## [1] 0.01
##
## $s2.post
##   1007_s_at     1053_at      117_at      121_at   1255_g_at
## 448951.4715   1541.3682   2161.7709  34121.3327    618.0261
## 22278 more elements ...
##
## $t
##            Contrasts
##             Breast_Cancer-Controls
##   1007_s_at             -0.6860442
##   1053_at                0.3025900
##   117_at                 0.2192935
##   121_at                 0.9844356
##   1255_g_at              0.6106339
## 22278 more rows ...
##
## $df.total
## [1] 40.77581 40.77581 40.77581 40.77581 40.77581
## 22278 more elements ...
##
## $p.value
##            Contrasts
##             Breast_Cancer-Controls
##   1007_s_at              0.4965678
##   1053_at                0.7637406
##   117_at                 0.8275155
##   121_at                 0.3307069
##   1255_g_at              0.5448308
## 22278 more rows ...
##
## $lods
##            Contrasts
##             Breast_Cancer-Controls
```

```
##   1007_s_at              -4.619226
##   1053_at                -4.635493
##   117_at                 -4.637376
##   121_at                 -4.598269
##   1255_g_at              -4.623394
## 22278 more rows ...
##
## $F
## [1] 0.47065666 0.09156070 0.04808962 0.96911338 0.37287376
## 22278 more elements ...
##
## $F.p.value
## [1] 0.4965678 0.7637406 0.8275155 0.3307069 0.5448308
## 22278 more elements ...
```