Course Code: CS 4115

Course Title: Computational Biology

Take Home Assignment – Individual – Report
Gene Expression data analysis

Name : Nimna Alupotha Gamage (NIMNA A. G. T.)

Index No.: s14682

Reg. No. : 2019s17241

Degree : Bioinformatics

Date : 14/2/2024

Content

Data – Human Airway smooth muscle Transcriptome changes in response to Asthma medications (GSE52778)

1. Describe the data.

This dataset consists of the gene expression data of **64102 genes** from **8 samples**. Asthma is a chronic inflammatory airway disease. The most common medications target the airway smooth muscle in the treatment of asthma.

According to the overall design of the experiment, mRNA profiles obtained via RNA-Seq for **four** primary human airway smooth muscle **cell lines** that were treated with dexamethasone or were left untreated.

The data presented as a matrix. Genes are represented in rows and samples are in columns. Each and every row of the data matrix represents the expression data of a single gene whereas each column represents the expression data of a single sample.

The class of the experiment is RangedSummarizedExperiment. The dimensions of the data as represent in the code is, 64102 8. Counts are taken as assays.

```
data(airway)
airway

## class: RangedSummarizedExperiment
## dim: 64102 8
## metadata(1): ''
## assays(1): counts
## rownames(64102): ENSG00000000003 ENSG00000000005 ... LRG_98 LRG_99
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(9): SampleName cell ... Sample BioSample
```

Following shows the sample information extracted from the 'sample_info.csv' file. Four cell lines shown and each cell line has samples which are untreated and treated with 'dexamethasone'.

```
sample_info

##            cellLine dexamethasone
## SRR1039508   N61311     untreated
## SRR1039509   N61311       treated
## SRR1039512  N052611     untreated
## SRR1039513  N052611       treated
## SRR1039516  N080611     untreated
## SRR1039517  N080611       treated
## SRR1039520  N061011     untreated
## SRR1039521  N061011       treated
```

Counts data which contained in the ‘counts_data.csv’ file is shown below. Genes are represented in the rows and samples are represented in the columns.

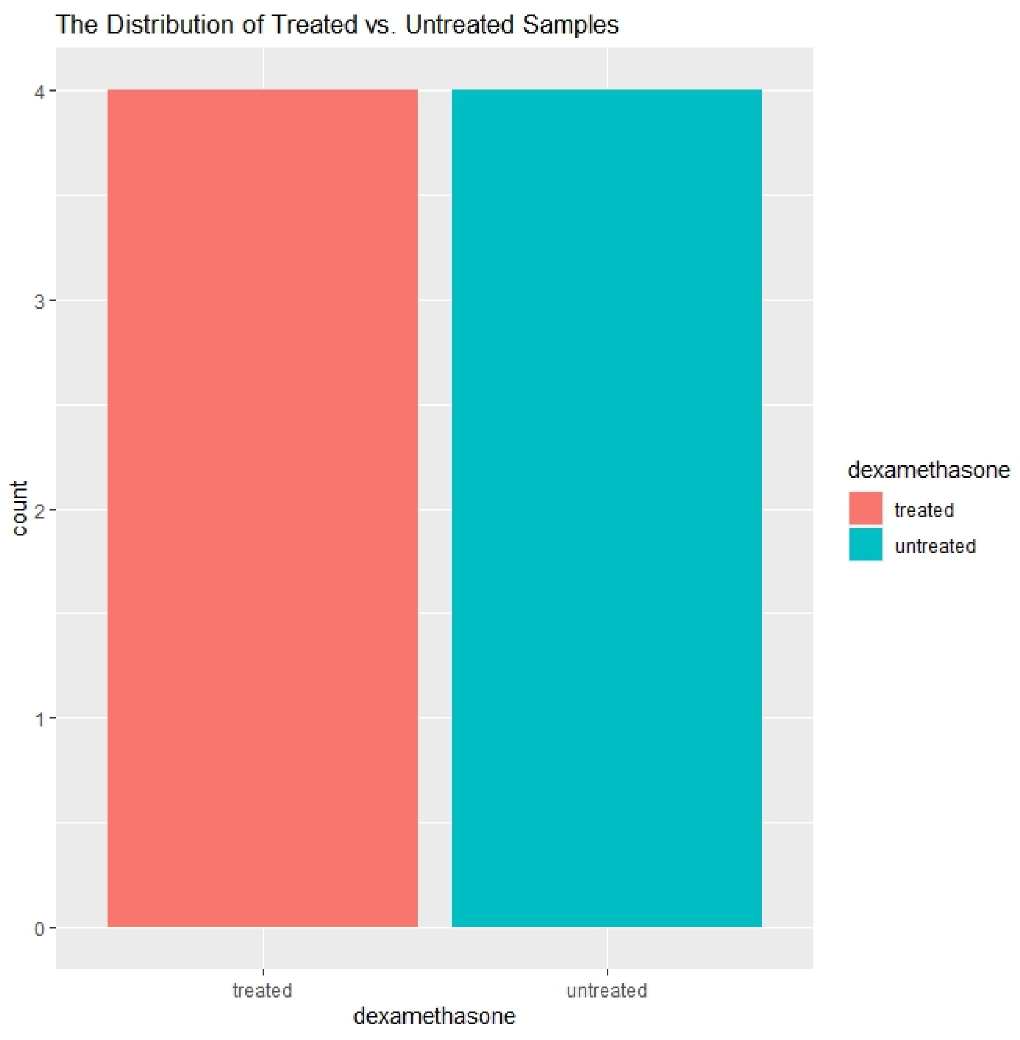
```
#countsData
head(countsData)

##                 SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003        679        448        873        408       1138
## ENSG00000000005          0          0          0          0          0
## ENSG00000000419        467        515        621        365        587
## ENSG00000000457        260        211        263        164        245
## ENSG00000000460         60         55         40         35         78
## ENSG00000000938          0          0          2          0          1
##                 SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003       1047        770        572
## ENSG00000000005          0          0          0
## ENSG00000000419        799        417        508
## ENSG00000000457        331        233        229
## ENSG00000000460         63         76         60
## ENSG00000000938          0          0          0
```

Following shows the summary statistics of the data.

```
# Summary statistics of the data

# Summary of sample information
summary(sample_info_csv)

##    cellLine          dexamethasone
##  Length:8           Length:8
##  Class :character   Class :character
##  Mode  :character   Mode  :character

# Summary of counts data
summary(countsData_csv)

##    SRR1039508       SRR1039509        SRR1039512        SRR1039513
##  Min.   :     0   Min.   :     0.0   Min.   :     0.0   Min.   :     0.0
##  1st Qu.:     0   1st Qu.:     0.0   1st Qu.:     0.0   1st Qu.:     0.0
##  Median :     0   Median :     0.0   Median :     0.0   Median :     0.0
##  Mean   :   322   Mean   :   293.4   Mean   :   395.4   Mean   :   236.6
##  3rd Qu.:    10   3rd Qu.:     8.0   3rd Qu.:    12.0   3rd Qu.:     6.0
##  Max.   :297906   Max.   :255662.0   Max.   :513766.0   Max.   :273878.0
##    SRR1039516        SRR1039517        SRR1039520        SRR1039521
##  Min.   :     0.0   Min.   :     0.0   Min.   :     0.0   Min.   :     0.0
##  1st Qu.:     0.0   1st Qu.:     0.0   1st Qu.:     0.0   1st Qu.:     0.0
##  Median :     0.0   Median :     0.0   Median :     0.0   Median :     0.0
##  Mean   :   381.4   Mean   :   480.8   Mean   :   298.4   Mean   :   330.2
##  3rd Qu.:    11.0   3rd Qu.:    12.0   3rd Qu.:     9.0   3rd Qu.:     8.0
##  Max.   :397791.0   Max.   :401539.0   Max.   :378834.0   Max.   :372489.0
```

Following bar plot shows the distribution of treated samples vs. untreated samples. Equal distribution can be seen as the number of treated and untreated samples are equal (4 samples for each)

```
# Bar plot - The distribution of treated vs. untreated samples

# Install the necessary packages
# install.packages("ggplot2")

# Load the packages into the R session
library(ggplot2)

ggplot(sample_info_csv, aes(x = dexamethasone, fill = dexamethasone)) +
  geom_bar() +
  labs(title = "The Distribution of Treated vs. Untreated Samples")
```



2. Use DESeq2 package in R to identify a list of differentially expressed genes.

Relevant packages are installed and loaded them to the R session.

```
#Q2. Use DESeq2 package in R to identify a list of differentially expressed
genes

# Install the necessary packages
# BiocManager::install("DESeq2")

# Load the packages into the R session
library(DESeq2)
```

All the columns of the data matrix of count_data should be in the rows of sample_info. That is checked below. And also, columns of the data matrix of count_data should be the same as the rows of sample_info.

```
# Ensure that the column names in countData matches the rownames in colData
all(colnames(countsData_csv) %in% rownames(sample_info_csv))

## [1] TRUE

# Ensure that the above data is in same order
all(colnames(countsData_csv) == rownames(sample_info_csv))

## [1] TRUE
```

`DESeqDataSet object` is created. The design factor is ‘`dexamethasone’`

‘dds’ is the object and it has the dimension of `64102 8`

```
# Create DESeqDataSet object from the count data
# Use 'dexamethasone' as the design factor
dds <- DESeqDataSetFromMatrix(countData = countsData_csv,
                              colData = sample_info_csv,
                              design = ~ dexamethasone)

dds

## class: DESeqDataSet
## dim: 64102 8
## metadata(1): version
## assays(1): counts
## rownames(64102): ENSG00000000003 ENSG00000000005 ... LRG_98 LRG_99
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(2): cellLine dexamethasone
```

The data of the DESeqDataSet object if filtered. The rows with the low gene counts were removed while keeping the gene with atleast 10 reads. After the filtering 'dds' object has the dimension of `22369 8`

```
# prefiltering on DESeqDataSet object
# Remove rows with low gene counts
# Keeping rows with at least 10 reades total
keep <- rowSums(counts(dds)) >= 10
dds <- dds[keep,]

dds

## class: DESeqDataSet
## dim: 22369 8
## metadata(1): version
## assays(1): counts
## rownames(22369): ENSG00000000003 ENSG00000000419 ... ENSG00000273487
##   ENSG00000273488
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(2): cellLine dexamethasone
```

The reference factor level was set and it is the 'untreated group'. 'Relevel()' function is used to do that.

```
# Set the factor level
# Reference level - untreated
# Compare untreated with other levels
dds$dexamethasone <- relevel(dds$dexamethasone, ref = "untreated")
```

In the Differential expression analysis using 'DESeq()' function, following steps are done. Those steps are; estimating size factors, estimating dispersions, gene-wise dispersion estimates, mean-dispersion relationship, final dispersion estimates, fitting model and testing.

```
# Differential expression analysis using 'DESeq()' function
dds <- DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

The results of the 'dds' object are as follows. The adjusted p-value is < 0.1. There are 1884 upregulated genes and 1502 downregulated genes.

```
# Extract differential expression results
res = results(dds)


# Explore Results
summary(res)

##
## out of 22369 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 1884, 8.4%
## LFC < 0 (down)     : 1502, 6.7%
## outliers [1]       : 51, 0.23%
## low counts [2]     : 3903, 17%
## (mean count < 4)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

If the alpha value is changed, the numbers of upregulated and downregulated genes are also changed. Following the adjusted p-value is < 0.01.

```
# optional-Changing the alpha value
res0.01 <- results(dds, alpha = 0.01)
summary(res0.01)

##
## out of 22369 with nonzero total read count
## adjusted p-value < 0.01
## LFC > 0 (up)       : 1030, 4.6%
## LFC < 0 (down)     : 708, 3.2%
## outliers [1]       : 51, 0.23%
## low counts [2]     : 5200, 23%
## (mean count < 6)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results

# lists the coefficients
resultsNames(dds)

## [1] "Intercept"
"dexamethasone_treated_vs_untreated"
```

Significantly differentially expressed genes are filtered using the p-value threshold of 0.05 and absolute log2FoldChange of 1. The 'DE_genes' data frame consists with 872 rows and 6 columns. Initially, there were **64102 genes** and out of those genes **872 genes** are selected as significantly differentially expressed genes. The list of those 872 genes were written to a csv file. ('s14682_De_genes.csv').

```
# Filter significantly differentially expressed genes
# Adjust p-value threshold = 0.05
DE_genes = subset(res, padj < 0.05 & abs(log2FoldChange) > 1)


# Get the differentially expressed genes
head(DE_genes)

## log2 fold change (MLE): dexamethasone treated vs untreated
## Wald test p-value: dexamethasone treated vs untreated
## DataFrame with 6 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE      stat      pvalue
##                 <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000003402 2546.6093        1.18345  0.163539   7.23648 4.60482e-13
## ENSG00000004799  914.3706        2.54406  0.901176   2.82304 4.75701e-03
## ENSG00000004846   17.9989       -1.88130  0.699835  -2.68821 7.18368e-03
## ENSG00000005471   33.6637       -1.21869  0.434789  -2.80294 5.06394e-03
## ENSG00000006788   10.2534        3.16910  1.067378   2.96905 2.98719e-03
## ENSG00000008256 3716.5654        1.20463  0.204288   5.89671 3.70817e-09
##                        padj
##                   <numeric>
## ENSG00000003402 5.47082e-11
## ENSG00000004799 3.59682e-02
## ENSG00000004846 4.93426e-02
## ENSG00000005471 3.77235e-02
## ENSG00000006788 2.50954e-02
## ENSG00000008256 1.88635e-07

tail(DE_genes)

## log2 fold change (MLE): dexamethasone treated vs untreated
## Wald test p-value: dexamethasone treated vs untreated
## DataFrame with 6 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE      stat      pvalue
##                 <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000272796  17.64008       -1.66348  0.495293  -3.35857 7.83474e-04
## ENSG00000272841 114.90320       -2.05355  0.520234  -3.94735 7.90198e-05
## ENSG00000272870 130.63591        1.05392  0.197213   5.34406 9.08880e-08
## ENSG00000273162   9.98609       -1.66989  0.583558  -2.86157 4.21546e-03
## ENSG00000273179 167.91931       -1.40771  0.458272  -3.07178 2.12790e-03
## ENSG00000273259   8.86088        3.85950  1.071082   3.60336 3.14126e-04
##                        padj
##                   <numeric>
## ENSG00000272796 8.68091e-03
## ENSG00000272841 1.25228e-03
## ENSG00000272870 3.29469e-06
## ENSG00000273162 3.26716e-02
## ENSG00000273179 1.93206e-02
## ENSG00000273259 4.05086e-03
```

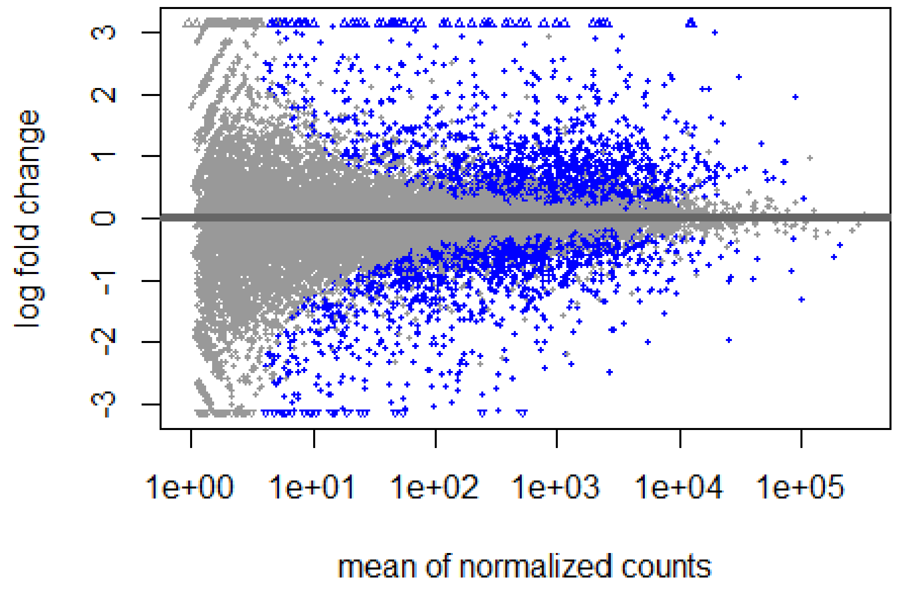
```
print(DE_genes)

## log2 fold change (MLE): dexamethasone treated vs untreated
## Wald test p-value: dexamethasone treated vs untreated
## DataFrame with 872 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE      stat      pvalue
##                 <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000003402 2546.6093        1.18345  0.163539   7.23648 4.60482e-13
## ENSG00000004799  914.3706        2.54406  0.901176   2.82304 4.75701e-03
## ENSG00000004846   17.9989       -1.88130  0.699835  -2.68821 7.18368e-03
## ENSG00000005471   33.6637       -1.21869  0.434789  -2.80294 5.06394e-03
## ENSG00000006788   10.2534        3.16910  1.067378   2.96905 2.98719e-03
## ...                   ...            ...       ...       ...         ...
## ENSG00000272841 114.90320       -2.05355  0.520234  -3.94735 7.90198e-05
## ENSG00000272870 130.63591        1.05392  0.197213   5.34406 9.08880e-08
## ENSG00000273162   9.98609       -1.66989  0.583558  -2.86157 4.21546e-03
## ENSG00000273179 167.91931       -1.40771  0.458272  -3.07178 2.12790e-03
## ENSG00000273259   8.86088        3.85950  1.071082   3.60336 3.14126e-04
##                        padj
##                   <numeric>
## ENSG00000003402 5.47082e-11
## ENSG00000004799 3.59682e-02
## ENSG00000004846 4.93426e-02
## ENSG00000005471 3.77235e-02
## ENSG00000006788 2.50954e-02
## ...                     ...
## ENSG00000272841 1.25228e-03
## ENSG00000272870 3.29469e-06
## ENSG00000273162 3.26716e-02
## ENSG00000273179 1.93206e-02
## ENSG00000273259 4.05086e-03
```

```
#Write differentially expressed genes into a csv file
write.table(DE_genes, file = "s14682_De_genes.csv", sep = ',')
```

```
## Above list of differentially expressed genes are retrieved based on the comparison
## between treated and untreated samples using DESeq2 package in R.
## The significance threshold (padj < 0.05) can be adjusted based on the specific analysis requirements.
```

The significantly differencially expressed gene data are represented using the ‘MA plot’.

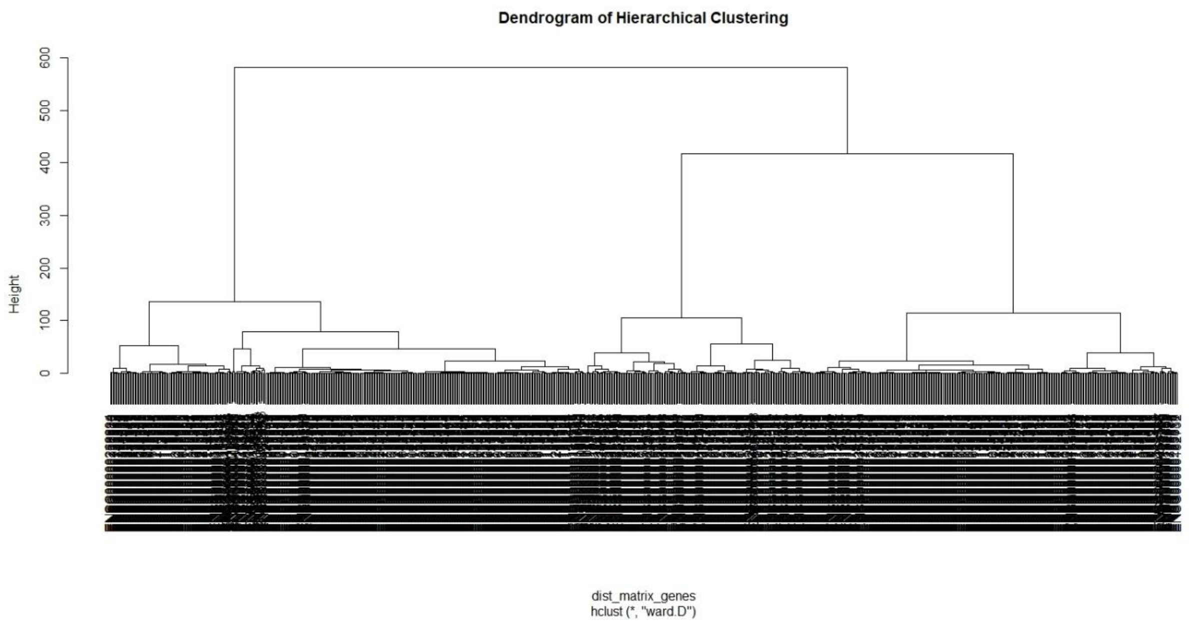
```
# Visualize the data - MA plot
# Plots the log-fold-change between experimental groups against the mean
expression across all the samples for each gene.
plotMA(res)
```



3. Identify the subgroups of differentially expressed genes by hierarchical clustering. – get the Clustering image, Members of each group

Above identified differentially expressed genes are clustered using hierarchical clustering method. The normalized data are used to compute a distance matrix. The distances matrix is used to do the hierarchical clustering using the ‘hclust()’ function. The ‘ward.D’ method is used. The dendrogram was constructed.

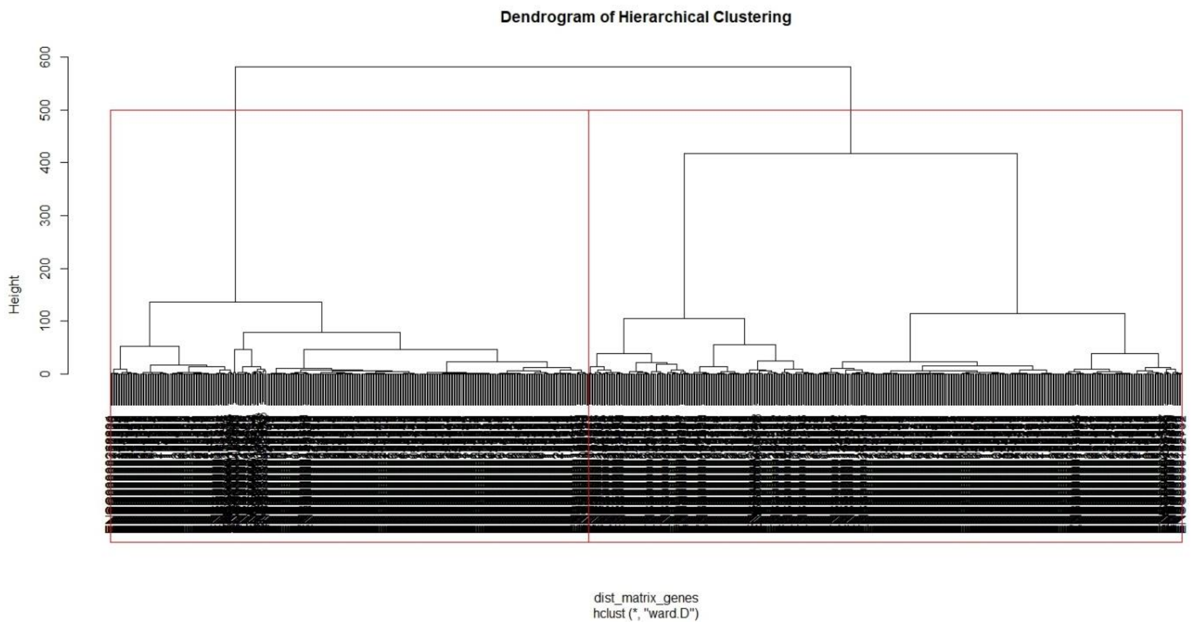
```
### Hierarchical clustering on differentially expressed genes

# Standardize/normalize the data
DE_genes = scale(DE_genes)

# Obtain the distance matrix
dist_matrix_genes = dist(DE_genes)

# Using of 'hclust()' function
hc_genes = hclust(dist_matrix_genes, method = "ward.D")

# Visualize the Dendrogram using the 'plot()' function
plot(hc_genes, main = "Dendrogram of Hierarchical Clustering")
```



The cluster assignment were extracted using ‘cutree()’ function. As initially states there were two groups as ‘treated’ and ‘untreated’. Therefore, the k is selected as 2.

To represent the clusters in dendrogram, ‘rect.hclust()’ function is used.

```
# Extract cluster assignments using 'cutree()' function
# Specified desired no. of clusters(e. g. k = 2; treated, untreated)
cluster_assignments_genes = cutree(hc_genes, k = 2)

# devide the 2 clusters using 2 rectangles for the visualization
rect.hclust(hc_genes,k=2,border = "red")
```



The members of each cluster is printed according to the following code.

Code

```
# Extract members of each gene cluster
group_members <- lapply(unique(cluster_assignments_genes), function(group) {
  genes_in_group <- rownames(gene_expr_genes)[cluster_assignments_genes == group]
  return(genes_in_group)
})

# Display members of each group
print("Member genes of each group:")
for (i in seq_along(group_members)) {
  cat("Group", i, ": ", paste(group_members[[i]], collapse = ", "), "\n")
}
```

Console output

```
> # Extract members of each gene cluster
> group_members <- lapply(unique(cluster_assignments_genes), function(group) {
+   genes_in_group <- rownames(gene_expr_genes)[cluster_assignments_genes == group]
+   return(genes_in_group)
+ })
> # Display members of each group
> print("Member genes of each group:")
[1] "Member genes of each group:"
> for (i in seq_along(group_members)) {
+   cat("Group", i, ": ", paste(group_members[[i]], collapse = ", "), "\n")
+ }
```

Group 1 : ENSG00000003402, ENSG00000008256, ENSG00000008311, ENSG00000009413, ENSG00000011198, ENSG00000020577, ENSG00000023909, ENSG00000025708, ENSG00000035664, ENSG00000048540, ENSG00000057657, ENSG00000060718, ENSG00000067082, ENSG00000067798, ENSG00000068383, ENSG00000068831, ENSG00000069431, ENSG00000070388, ENSG00000070404, ENSG00000071282, ENSG00000072163, ENSG00000072571, ENSG00000072958, ENSG00000074590, ENSG00000074660, ENSG00000077684, ENSG00000077943, ENSG00000078053, ENSG00000079691, ENSG00000081052, ENSG00000081320, ENSG00000083223, ENSG00000083290, ENSG00000084090, ENSG00000085117, ENSG00000087448, ENSG00000095637, ENSG00000096060, ENSG00000097096, ENSG00000099204, ENSG00000099337, ENSG00000099840, ENSG00000099849, ENSG00000099860, ENSG00000100033, ENSG00000100206, ENSG00000100242, ENSG00000100767, ENSG00000101342, ENSG00000101347, ENSG00000102466, ENSG00000102554, ENSG00000102760, ENSG00000102804, ENSG00000102996, ENSG00000103064, ENSG00000103175, ENSG00000103196, ENSG00000105835, ENSG00000105889, ENSG00000106123, ENSG00000106617, ENSG00000107104, ENSG00000107562, ENSG00000107796, ENSG00000107968, ENSG00000108387, ENSG00000108604, ENSG00000108821, ENSG00000108950, ENSG00000108960, ENSG00000109861, ENSG00000109906, ENSG00000110756, ENSG00000111859, ENSG00000112936, ENSG00000114098, ENSG00000114270, ENSG00000115419, ENSG00000115828, ENSG00000116194, ENSG00000116285, ENSG00000116675, ENSG00000116962, ENSG00000117479, ENSG00000118257, ENSG00000118507, ENSG00000118689, ENSG00000119138, ENSG00000119139, ENSG00000119508, ENSG00000119711, ENSG00000120129, ENSG00000120162, ENSG00000122035, ENSG00000123358, ENSG00000123562, ENSG00000123685, ENSG00000124151, ENSG00000124374, ENSG00000124440, ENSG00000125148, ENSG00000126803, ENSG00000127324, ENSG00000127954, ENSG00000128045, ENSG00000128311, ENSG00000128699, ENSG00000128923, ENSG00000130066, ENSG00000131386, ENSG00000131459, ENSG00000131979, ENSG00000132170, ENSG00000132518, ENSG00000132970, ENSG00000133142, ENSG00000133401, ENSG00000133816, ENSG00000134243, ENSG00000134294, ENSG00000134686, ENSG00000135362, ENSG00000135604, ENSG00000135678, ENSG00000135821, ENSG00000135917, ENSG00000136237, ENSG00000136383, ENSG00000136436, ENSG00000136478, ENSG00000136546, ENSG00000137393, ENSG00000137672, ENSG00000137673, ENSG00000137767, ENSG00000137801, ENSG00000137869, ENSG00000137880, ENSG00000137959, ENSG00000137962, ENSG00000138073, ENSG00000138074, ENSG00000138166, ENSG00000138356, ENSG00000138483, ENSG00000138615, ENSG00000138678, ENSG00000138829, ENSG00000139132, ENSG00000140511, ENSG00000140545, ENSG00000140807, ENSG00000141150, ENSG00000141298, ENSG00000141401, ENSG00000142871, ENSG00000143127, ENSG00000143869, ENSG00000143878, ENSG00000144362, ENSG00000145244, ENSG00000145390, ENSG00000145569, ENSG00000145675, ENSG00000146122, ENSG00000146373, ENSG00000147027, ENSG00000147119, ENSG00000147576, ENSG00000148120, ENSG00000148175, ENSG00000149218, ENSG00000149591, ENSG00000150907,

ENSG00000150938, ENSG00000151690, ENSG00000151726, ENSG00000152463,
ENSG00000152583, ENSG00000152779, ENSG00000153207, ENSG00000153904,
ENSG00000154127, ENSG00000154262, ENSG00000154734, ENSG00000154736,
ENSG00000154930, ENSG00000155324, ENSG00000156675, ENSG00000156804,
ENSG00000157150, ENSG00000157152, ENSG00000157214, ENSG00000157510,
ENSG00000157514, ENSG00000157617, ENSG00000158246, ENSG00000158716,
ENSG00000158813, ENSG00000159212, ENSG00000160200, ENSG00000160256,
ENSG00000161267, ENSG00000161647, ENSG00000162407, ENSG00000162426,
ENSG00000162614, ENSG00000162616, ENSG00000162630, ENSG00000162772,
ENSG00000162878, ENSG00000162998, ENSG00000163083, ENSG00000163110,
ENSG00000163171, ENSG00000163251, ENSG00000163378, ENSG00000163431,
ENSG00000163513, ENSG00000163661, ENSG00000163697, ENSG00000163803,
ENSG00000163884, ENSG00000164104, ENSG00000164105, ENSG00000164125,
ENSG00000164292, ENSG00000164330, ENSG00000164442, ENSG00000164647,
ENSG00000165030, ENSG00000165507, ENSG00000165644, ENSG00000165899,
ENSG00000165995, ENSG00000166260, ENSG00000166741, ENSG00000166825,
ENSG00000166979, ENSG00000167191, ENSG00000167549, ENSG00000167641,
ENSG00000167645, ENSG00000168309, ENSG00000168481, ENSG00000168556,
ENSG00000168621, ENSG00000168646, ENSG00000168994, ENSG00000169031,
ENSG00000169218, ENSG00000169271, ENSG00000169715, ENSG00000169738,
ENSG00000169750, ENSG00000169908, ENSG00000170214, ENSG00000170323,
ENSG00000170485, ENSG00000171793, ENSG00000171819, ENSG00000172260,
ENSG00000172403, ENSG00000172465, ENSG00000173838, ENSG00000173918,
ENSG00000174306, ENSG00000174437, ENSG00000174680, ENSG00000174697,
ENSG00000174944, ENSG00000175471, ENSG00000175741, ENSG00000175946,
ENSG00000176928, ENSG00000176971, ENSG00000177283, ENSG00000177575,
ENSG00000177666, ENSG00000177674, ENSG00000178015, ENSG00000178723,
ENSG00000179094, ENSG00000179294, ENSG00000179300, ENSG00000179593,
ENSG00000179820, ENSG00000179862, ENSG00000180672, ENSG00000181061,
ENSG00000182552, ENSG00000182836, ENSG00000183044, ENSG00000184156,
ENSG00000184307, ENSG00000185022, ENSG00000185112, ENSG00000185432,
ENSG00000185813, ENSG00000185950, ENSG00000186575, ENSG00000187193,
ENSG00000187288, ENSG00000187498, ENSG00000188916, ENSG00000189221,
ENSG00000196507, ENSG00000196569, ENSG00000196616, ENSG00000196850,
ENSG00000196975, ENSG00000197301, ENSG00000197312, ENSG00000197381,
ENSG00000198108, ENSG00000198431, ENSG00000198624, ENSG00000203685,
ENSG00000205364, ENSG00000206190, ENSG00000206538, ENSG00000211445,
ENSG00000211448, ENSG00000213160, ENSG00000213626, ENSG00000213639,
ENSG00000213763, ENSG00000214274, ENSG00000214944, ENSG00000215481,
ENSG00000219565, ENSG00000221869, ENSG00000221968, ENSG00000223401,
ENSG00000224080, ENSG00000224468, ENSG00000225313, ENSG00000226121,
ENSG00000226950, ENSG00000229644, ENSG00000229647, ENSG00000230018,
ENSG00000231246, ENSG00000233117, ENSG00000235927, ENSG00000237697,
ENSG00000237928, ENSG00000240445, ENSG00000240859, ENSG00000241399,

ENSG00000242539, ENSG00000243244, ENSG00000244490, ENSG00000245812, ENSG00000246430, ENSG00000247311, ENSG00000248144, ENSG00000248187, ENSG00000249364, ENSG00000250899, ENSG00000250934, ENSG00000250978, ENSG00000253139, ENSG00000253276, ENSG00000253368, ENSG00000253833, ENSG00000254109, ENSG00000254254, ENSG00000254842, ENSG00000254851, ENSG00000258016, ENSG00000259426, ENSG00000260802, ENSG00000260841, ENSG00000261468, ENSG00000261490, ENSG00000261589, ENSG00000261685, ENSG00000264868, ENSG00000267480, ENSG00000267669, ENSG00000268894, ENSG00000268913, ENSG00000269289, ENSG00000269728, ENSG00000270689, ENSG00000272870, ENSG00000273259

Group 2 : ENSG00000004799, ENSG00000004846, ENSG00000005471, ENSG00000006788, ENSG00000012048, ENSG00000013293, ENSG00000013297, ENSG00000015520, ENSG00000016391, ENSG00000019186, ENSG00000021645, ENSG00000025423, ENSG00000028277, ENSG00000040731, ENSG00000041515, ENSG00000046653, ENSG00000049246, ENSG00000049759, ENSG00000054938, ENSG00000055163, ENSG00000056736, ENSG00000061337, ENSG00000064201, ENSG00000064309, ENSG00000065809, ENSG00000066468, ENSG00000069535, ENSG00000070808, ENSG00000070882, ENSG00000073756, ENSG00000075213, ENSG00000075240, ENSG00000077063, ENSG00000078114, ENSG00000079101, ENSG00000079435, ENSG00000079462, ENSG00000082126, ENSG00000084710, ENSG00000088756, ENSG00000089041, ENSG00000091262, ENSG00000091428, ENSG00000091831, ENSG00000092621, ENSG00000092853, ENSG00000092969, ENSG00000095585, ENSG00000099194, ENSG00000099998, ENSG00000100292, ENSG00000100302, ENSG00000100592, ENSG00000100739, ENSG00000100784, ENSG00000101255, ENSG00000101265, ENSG00000101825, ENSG00000101938, ENSG00000102385, ENSG00000102524, ENSG00000102935, ENSG00000102984, ENSG00000103257, ENSG00000103485, ENSG00000103647, ENSG00000103710, ENSG00000103742, ENSG00000104894, ENSG00000104951, ENSG00000105486, ENSG00000105516, ENSG00000105664, ENSG00000105711, ENSG00000105989, ENSG00000106003, ENSG00000106034, ENSG00000106484, ENSG00000106976, ENSG00000107611, ENSG00000107731, ENSG00000107821, ENSG00000108602, ENSG00000108684, ENSG00000108700, ENSG00000108830, ENSG00000109625, ENSG00000109674, ENSG00000109689, ENSG00000109881, ENSG00000110203, ENSG00000110900, ENSG00000111110, ENSG00000111728, ENSG00000111816, ENSG00000112137, ENSG00000112218, ENSG00000112715, ENSG00000112773, ENSG00000112837, ENSG00000114670, ENSG00000116106, ENSG00000116299, ENSG00000116584, ENSG00000116690, ENSG00000116711, ENSG00000116991, ENSG00000117152, ENSG00000117461, ENSG00000117600, ENSG00000119514, ENSG00000119630, ENSG00000119703, ENSG00000119714, ENSG00000120837, ENSG00000120899, ENSG00000121621, ENSG00000122641, ENSG00000122679, ENSG00000122877, ENSG00000122966, ENSG00000123405, ENSG00000123610, ENSG00000123612, ENSG00000123689, ENSG00000124134, ENSG00000124249, ENSG00000124466, ENSG00000124762, ENSG00000124766, ENSG00000125398, ENSG00000125657, ENSG00000125848, ENSG00000125965, ENSG00000126016,

ENSG00000126785, ENSG00000126860, ENSG00000126861, ENSG00000126878,
ENSG00000126882, ENSG00000126950, ENSG00000127083, ENSG00000127589,
ENSG00000127824, ENSG00000128165, ENSG00000128262, ENSG00000128285,
ENSG00000128342, ENSG00000128510, ENSG00000128594, ENSG00000128606,
ENSG00000128917, ENSG00000129048, ENSG00000129270, ENSG00000129467,
ENSG00000130487, ENSG00000130513, ENSG00000130592, ENSG00000131242,
ENSG00000131389, ENSG00000131730, ENSG00000131771, ENSG00000132321,
ENSG00000132326, ENSG00000132334, ENSG00000132622, ENSG00000132854,
ENSG00000132965, ENSG00000133069, ENSG00000133216, ENSG00000134070,
ENSG00000134253, ENSG00000134259, ENSG00000134321, ENSG00000134363,
ENSG00000134376, ENSG00000135069, ENSG00000135472, ENSG00000136267,
ENSG00000136999, ENSG00000137266, ENSG00000137331, ENSG00000137642,
ENSG00000137872, ENSG00000138135, ENSG00000138311, ENSG00000138316,
ENSG00000138669, ENSG00000138735, ENSG00000139055, ENSG00000139269,
ENSG00000139354, ENSG00000140105, ENSG00000140600, ENSG00000141469,
ENSG00000143226, ENSG00000143320, ENSG00000143344, ENSG00000143494,
ENSG00000143507, ENSG00000143786, ENSG00000143891, ENSG00000144369,
ENSG00000144648, ENSG00000144891, ENSG00000145242, ENSG00000145506,
ENSG00000145632, ENSG00000145685, ENSG00000145777, ENSG00000145861,
ENSG00000145911, ENSG00000146006, ENSG00000146250, ENSG00000146592,
ENSG00000147655, ENSG00000147883, ENSG00000148541, ENSG00000148677,
ENSG00000148848, ENSG00000149256, ENSG00000149403, ENSG00000149488,
ENSG00000149548, ENSG00000149633, ENSG00000150594, ENSG00000150636,
ENSG00000152495, ENSG00000152580, ENSG00000154263, ENSG00000154310,
ENSG00000154856, ENSG00000154864, ENSG00000155011, ENSG00000155130,
ENSG00000155749, ENSG00000155897, ENSG00000155962, ENSG00000157368,
ENSG00000157578, ENSG00000158125, ENSG00000158806, ENSG00000159023,
ENSG00000159167, ENSG00000159200, ENSG00000159713, ENSG00000160097,
ENSG00000160145, ENSG00000160223, ENSG00000160460, ENSG00000161381,
ENSG00000162493, ENSG00000162496, ENSG00000162643, ENSG00000162692,
ENSG00000163017, ENSG00000163072, ENSG00000163485, ENSG00000163491,
ENSG00000163823, ENSG00000164070, ENSG00000164106, ENSG00000164122,
ENSG00000164142, ENSG00000164171, ENSG00000164483, ENSG00000164484,
ENSG00000164619, ENSG00000164761, ENSG00000165072, ENSG00000165105,
ENSG00000165125, ENSG00000165244, ENSG00000165272, ENSG00000165388,
ENSG00000165495, ENSG00000165891, ENSG00000165895, ENSG00000166292,
ENSG00000166473, ENSG00000166592, ENSG00000166670, ENSG00000166762,
ENSG00000166793, ENSG00000167552, ENSG00000167642, ENSG00000167771,
ENSG00000167992, ENSG00000168398, ENSG00000168811, ENSG00000168918,
ENSG00000169129, ENSG00000169297, ENSG00000169744, ENSG00000169855,
ENSG00000170624, ENSG00000170647, ENSG00000170775, ENSG00000170873,
ENSG00000170989, ENSG00000171033, ENSG00000171132, ENSG00000171227,
ENSG00000171385, ENSG00000171502, ENSG00000171509, ENSG00000171617,
ENSG00000171817, ENSG00000171877, ENSG00000172399, ENSG00000172497,

ENSG00000172602, ENSG00000172738, ENSG00000172828, ENSG00000172955, ENSG00000172986, ENSG00000173083, ENSG00000173110, ENSG00000173114, ENSG00000173320, ENSG00000173947, ENSG00000175130, ENSG00000175197, ENSG00000175489, ENSG00000175538, ENSG00000175928, ENSG00000176293, ENSG00000176771, ENSG00000176909, ENSG00000177570, ENSG00000177606, ENSG00000177614, ENSG00000178038, ENSG00000178184, ENSG00000178662, ENSG00000178695, ENSG00000178734, ENSG00000179082, ENSG00000179242, ENSG00000179388, ENSG00000180139, ENSG00000181467, ENSG00000181634, ENSG00000182010, ENSG00000182379, ENSG00000182575, ENSG00000182580, ENSG00000182732, ENSG00000183092, ENSG00000183160, ENSG00000183454, ENSG00000183496, ENSG00000183508, ENSG00000183801, ENSG00000183876, ENSG00000184564, ENSG00000184916, ENSG00000185338, ENSG00000185745, ENSG00000185972, ENSG00000186198, ENSG00000186314, ENSG00000186469, ENSG00000186998, ENSG00000187479, ENSG00000188176, ENSG00000188312, ENSG00000188501, ENSG00000188536, ENSG00000188596, ENSG00000189007, ENSG00000196155, ENSG00000196196, ENSG00000196230, ENSG00000196476, ENSG00000196511, ENSG00000196517, ENSG00000196932, ENSG00000196950, ENSG00000197046, ENSG00000197210, ENSG00000197594, ENSG00000197943, ENSG00000198203, ENSG00000198542, ENSG00000198691, ENSG00000198944, ENSG00000198947, ENSG00000203722, ENSG00000203727, ENSG00000203943, ENSG00000205208, ENSG00000205213, ENSG00000205978, ENSG00000206052, ENSG00000206172, ENSG00000206561, ENSG00000211574, ENSG00000213402, ENSG00000213420, ENSG00000213493, ENSG00000214212, ENSG00000214814, ENSG00000215018, ENSG00000215386, ENSG00000220563, ENSG00000221994, ENSG00000223458, ENSG00000223764, ENSG00000223802, ENSG00000223811, ENSG00000223820, ENSG00000223949, ENSG00000224124, ENSG00000225032, ENSG00000225217, ENSG00000225383, ENSG00000225415, ENSG00000225783, ENSG00000226887, ENSG00000227051, ENSG00000227120, ENSG00000227268, ENSG00000228798, ENSG00000229116, ENSG00000229474, ENSG00000230417, ENSG00000231574, ENSG00000232871, ENSG00000235109, ENSG00000235513, ENSG00000235649, ENSG00000235959, ENSG00000237886, ENSG00000240291, ENSG00000240498, ENSG00000240764, ENSG00000240809, ENSG00000241990, ENSG00000243742, ENSG00000244301, ENSG00000246763, ENSG00000250657, ENSG00000253540, ENSG00000254726, ENSG00000254987, ENSG00000256340, ENSG00000256616, ENSG00000258457, ENSG00000259319, ENSG00000259448, ENSG00000259583, ENSG00000259673, ENSG00000259886, ENSG00000260230, ENSG00000260396, ENSG00000260455, ENSG00000260807, ENSG00000261121, ENSG00000261425, ENSG00000261452, ENSG00000261597, ENSG00000261827, ENSG00000263567, ENSG00000264007, ENSG00000264745, ENSG00000267339, ENSG00000267528, ENSG00000267683, ENSG00000269959, ENSG00000270093, ENSG00000270605, ENSG00000271930, ENSG00000272168, ENSG00000272341, ENSG00000272349, ENSG00000272356, ENSG00000272384, ENSG00000272744, ENSG00000272796, ENSG00000272841, ENSG00000273162, ENSG00000273179

s14682-DEG.R

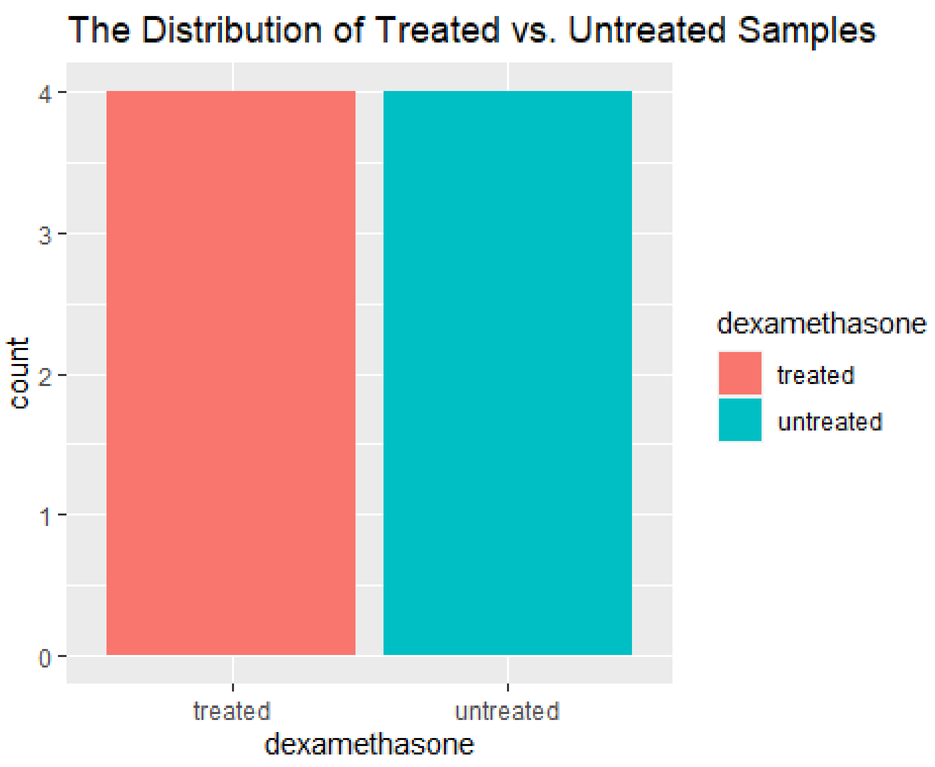
Complete code – compiled report

2024-02-20

```
#Take home Assignment - Individual
# Gene expression data analysis
# Author: Nimna A. G. T.
# Date: 14/2/2024

# Set working directory
setwd("D:/4th_yr_sem2/CS4115 Computational Biology/TH-Gene Expression data
analysis/Rcode_THA")

# Get working directory
getwd()

## [1] "D:/4th_yr_sem2/CS4115 Computational Biology/TH-Gene Expression data
analysis/Rcode_THA"

# Get data from airway package (run getData.R)

# Install the necessary packages
# BiocManager::install("airway")

# Load the packages into the R session
library(airway)

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

## Warning: package 'matrixStats' was built under R version 4.2.3

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: GenomicRanges

## Loading required package: stats4

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomeInfoDb
## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians

data(airway)
airway

## class: RangedSummarizedExperiment
## dim: 64102 8
## metadata(1): ''
## assays(1): counts
## rownames(64102): ENSG00000000003 ENSG00000000005 ... LRG_98 LRG_99
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(9): SampleName cell ... Sample BioSample

sample_info <- as.data.frame(colData(airway))
sample_info <- sample_info[,c(2,3)]
sample_info$dex <- gsub('trt', 'treated', sample_info$dex)
sample_info$dex <- gsub('untrt', 'untreated', sample_info$dex)
names(sample_info) <- c('cellLine', 'dexamethasone')
write.table(sample_info, file = "sample_info.csv", sep = ',', col.names = T,
row.names = T, quote = F)
sample_info

##            cellLine dexamethasone
## SRR1039508   N61311     untreated
## SRR1039509   N61311       treated
## SRR1039512  N052611     untreated
## SRR1039513  N052611       treated
## SRR1039516  N080611     untreated
## SRR1039517  N080611       treated
## SRR1039520  N061011     untreated
## SRR1039521  N061011       treated
```

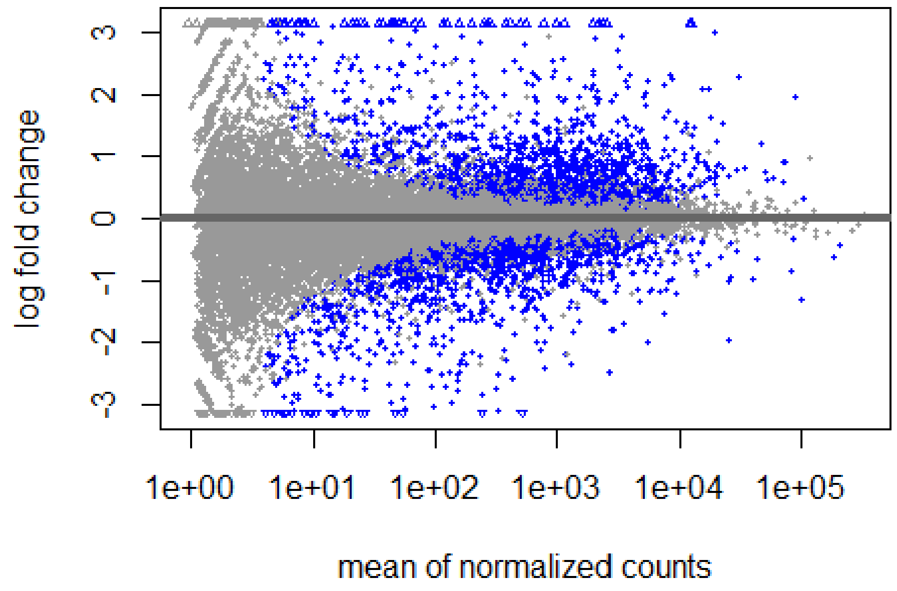
```
countsData <- assay(airway)
write.table(countsData, file = "counts_data.csv", sep = ',', col.names = T,
row.names = T, quote = F)
#countsData
head(countsData)

##                 SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003        679        448        873        408       1138
## ENSG00000000005          0          0          0          0          0
## ENSG00000000419        467        515        621        365        587
## ENSG00000000457        260        211        263        164        245
## ENSG00000000460         60         55         40         35         78
## ENSG00000000938          0          0          2          0          1
##                 SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003       1047        770        572
## ENSG00000000005          0          0          0
## ENSG00000000419        799        417        508
## ENSG00000000457        331        233        229
## ENSG00000000460         63         76         60
## ENSG00000000938          0          0          0

#Q1. Describe the data

# Read the csv files generated above

# Sample information
sample_info_csv <- read.csv("sample_info.csv", header = TRUE,
stringsAsFactors = FALSE)
sample_info_csv

##            cellLine dexamethasone
## SRR1039508   N61311     untreated
## SRR1039509   N61311       treated
## SRR1039512  N052611     untreated
## SRR1039513  N052611       treated
## SRR1039516  N080611     untreated
## SRR1039517  N080611       treated
## SRR1039520  N061011     untreated
## SRR1039521  N061011       treated

# Counts data
countsData_csv <- read.csv("counts_data.csv", header = TRUE, row.names = 1)
head(countsData_csv)

##                 SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003        679        448        873        408       1138
## ENSG00000000005          0          0          0          0          0
## ENSG00000000419        467        515        621        365        587
## ENSG00000000457        260        211        263        164        245
## ENSG00000000460        60         55         40         35         78
## ENSG00000000938         0          0          2          0          1
##                 SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003       1047        770        572
## ENSG00000000005          0          0          0
## ENSG00000000419        799        417        508
## ENSG00000000457        331        233        229
## ENSG00000000460         63         76         60
## ENSG00000000938          0          0          0

##1
# Summary statistics of the data

# Summary of sample information
summary(sample_info_csv)

##    cellLine          dexamethasone     
##  Length:8           Length:8          
##  Class :character   Class :character  
##  Mode  :character   Mode  :character  

# Summary of counts data
summary(countsData_csv)

##    SRR1039508       SRR1039509         SRR1039512         SRR1039513      
##  Min.   :     0   Min.   :     0.0   Min.   :     0.0   Min.   :     0.0  
##  1st Qu.:     0   1st Qu.:     0.0   1st Qu.:     0.0   1st Qu.:     0.0  
##  Median :     0   Median :     0.0   Median :     0.0   Median :     0.0  
##  Mean   :   322   Mean   :   293.4   Mean   :   395.4   Mean   :   236.6  
##  3rd Qu.:    10   3rd Qu.:     8.0   3rd Qu.:    12.0   3rd Qu.:     6.0  
##  Max.   :297906   Max.   :255662.0   Max.   :513766.0   Max.   :273878.0  
##    SRR1039516         SRR1039517         SRR1039520         SRR1039521      
##  Min.   :     0.0   Min.   :     0.0   Min.   :     0.0   Min.   :     0.0  
##  1st Qu.:     0.0   1st Qu.:     0.0   1st Qu.:     0.0   1st Qu.:     0.0  
##  Median :     0.0   Median :     0.0   Median :     0.0   Median :     0.0  
##  Mean   :   381.4   Mean   :   480.8   Mean   :   298.4   Mean   :   330.2  
##  3rd Qu.:    11.0   3rd Qu.:    12.0   3rd Qu.:     9.0   3rd Qu.:     8.0  
##  Max.   :397791.0   Max.   :401539.0   Max.   :378834.0   Max.   :372489.0  

##2
# Bar plot - The distribution of treated vs. untreated samples

# Install the necessary packages
# install.packages("ggplot2")

# Load the packages into the R session
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
ggplot(sample_info_csv, aes(x = dexamethasone, fill = dexamethasone)) +
  geom_bar() +
  labs(title = "The Distribution of Treated vs. Untreated Samples")
```



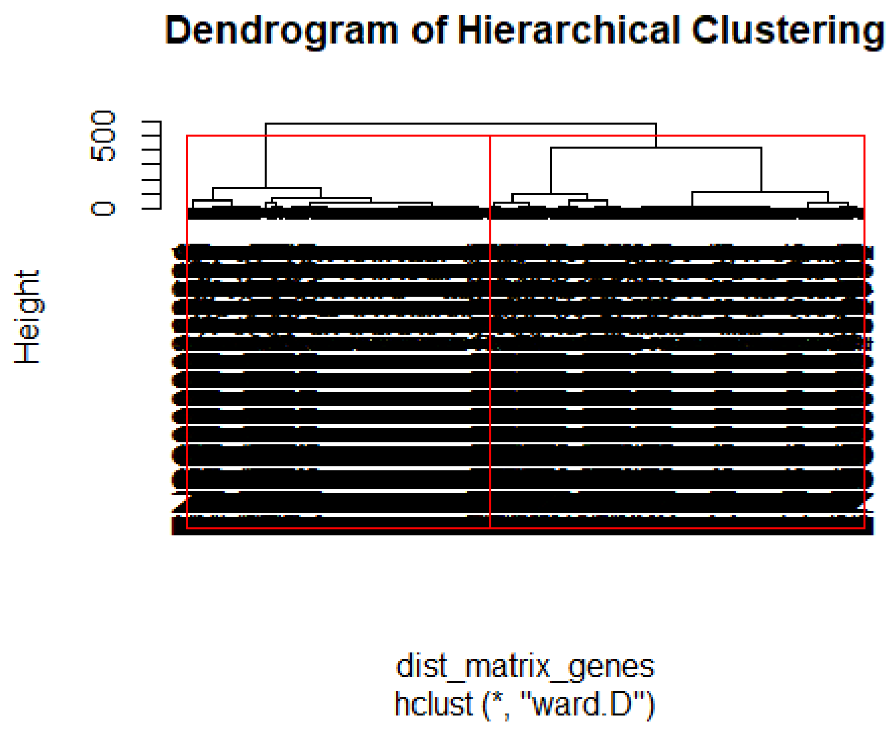
```
#Q2. Use DESeq2 package in R to identify a list of differentially expressed
genes

# Install the necessary packages
# BiocManager::install("DESeq2")

# Load the packages into the R session
library(DESeq2)

# Ensure that the column names in countData matches the rownames in colData
all(colnames(countsData_csv) %in% rownames(sample_info_csv))

## [1] TRUE

# Ensure that the above data is in same order
all(colnames(countsData_csv) == rownames(sample_info_csv))

## [1] TRUE

# Create DESeqDataSet object from the count data
# Use 'dexamethasone' as the design factor
dds <- DESeqDataSetFromMatrix(countData = countsData_csv,
                                    colData = sample_info_csv,
                                    design = ~ dexamethasone)

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables
in
## design formula are characters, converting to factors

dds

## class: DESeqDataSet
## dim: 64102 8
## metadata(1): version
## assays(1): counts
## rownames(64102): ENSG00000000003 ENSG00000000005 ... LRG_98 LRG_99
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(2): cellLine dexamethasone

# prefiltering on DESeqDataSet object
# Remove rows with low gene counts
# Keeping rows with at least 10 reades total
keep <- rowSums(counts(dds)) >= 10
dds <- dds[keep,]

dds

## class: DESeqDataSet
## dim: 22369 8
## metadata(1): version
## assays(1): counts
## rownames(22369): ENSG00000000003 ENSG00000000419 ... ENSG00000273487
##   ENSG00000273488
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(2): cellLine dexamethasone

# Set the factor level
# Reference level - untreated
# Compare untreated with other levels
dds$dexamethasone <- relevel(dds$dexamethasone, ref = "untreated")

# Differential expression analysis using 'DESeq()' function
dds <- DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates
## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

# Extract differential expression results
res = results(dds)


# Explore Results
summary(res)

##
## out of 22369 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 1884, 8.4%
## LFC < 0 (down)     : 1502, 6.7%
## outliers [1]       : 51, 0.23%
## low counts [2]     : 3903, 17%
## (mean count < 4)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results

# optional-Changing the alpha value
res0.01 <- results(dds, alpha = 0.01)
summary(res0.01)

##
## out of 22369 with nonzero total read count
## adjusted p-value < 0.01
## LFC > 0 (up)       : 1030, 4.6%
## LFC < 0 (down)     : 708, 3.2%
## outliers [1]       : 51, 0.23%
## low counts [2]     : 5200, 23%
## (mean count < 6)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results

# lists the coefficients
resultsNames(dds)

## [1] "Intercept"
"dexamethasone_treated_vs_untreated"

# Filter significantly differentially expressed genes
# Adjust p-value threshold = 0.05
DE_genes = subset(res, padj < 0.05 & abs(log2FoldChange) > 1)
```

```
# Get the differentially expressed genes
head(DE_genes)

## log2 fold change (MLE): dexamethasone treated vs untreated
## Wald test p-value: dexamethasone treated vs untreated
## DataFrame with 6 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE      stat      pvalue
##                 <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000003402 2546.6093        1.18345  0.163539   7.23648 4.60482e-13
## ENSG00000004799  914.3706        2.54406  0.901176   2.82304 4.75701e-03
## ENSG00000004846   17.9989       -1.88130  0.699835  -2.68821 7.18368e-03
## ENSG00000005471   33.6637       -1.21869  0.434789  -2.80294 5.06394e-03
## ENSG00000006788   10.2534        3.16910  1.067378   2.96905 2.98719e-03
## ENSG00000008256 3716.5654        1.20463  0.204288   5.89671 3.70817e-09
##                        padj
##                   <numeric>
## ENSG00000003402 5.47082e-11
## ENSG00000004799 3.59682e-02
## ENSG00000004846 4.93426e-02
## ENSG00000005471 3.77235e-02
## ENSG00000006788 2.50954e-02
## ENSG00000008256 1.88635e-07

tail(DE_genes)

## log2 fold change (MLE): dexamethasone treated vs untreated
## Wald test p-value: dexamethasone treated vs untreated
## DataFrame with 6 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE      stat      pvalue
##                 <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000272796  17.64008       -1.66348  0.495293  -3.35857 7.83474e-04
## ENSG00000272841 114.90320       -2.05355  0.520234  -3.94735 7.90198e-05
## ENSG00000272870 130.63591        1.05392  0.197213   5.34406 9.08880e-08
## ENSG00000273162   9.98609       -1.66989  0.583558  -2.86157 4.21546e-03
## ENSG00000273179 167.91931       -1.40771  0.458272  -3.07178 2.12790e-03
## ENSG00000273259   8.86088        3.85950  1.071082   3.60336 3.14126e-04
##                        padj
##                   <numeric>
## ENSG00000272796 8.68091e-03
## ENSG00000272841 1.25228e-03
## ENSG00000272870 3.29469e-06
## ENSG00000273162 3.26716e-02
## ENSG00000273179 1.93206e-02
## ENSG00000273259 4.05086e-03

print(DE_genes)

## log2 fold change (MLE): dexamethasone treated vs untreated
## Wald test p-value: dexamethasone treated vs untreated
## DataFrame with 872 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE      stat      pvalue
##                 <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000003402 2546.6093        1.18345  0.163539   7.23648 4.60482e-13
## ENSG00000004799  914.3706        2.54406  0.901176   2.82304 4.75701e-03
## ENSG00000004846   17.9989       -1.88130  0.699835  -2.68821 7.18368e-03
## ENSG00000005471   33.6637       -1.21869  0.434789  -2.80294 5.06394e-03
## ENSG00000006788   10.2534        3.16910  1.067378   2.96905 2.98719e-03
## ...                   ...            ...       ...       ...         ...
## ENSG00000272841 114.90320       -2.05355  0.520234  -3.94735 7.90198e-05
## ENSG00000272870 130.63591        1.05392  0.197213   5.34406 9.08880e-08
## ENSG00000273162   9.98609       -1.66989  0.583558  -2.86157 4.21546e-03
## ENSG00000273179 167.91931       -1.40771  0.458272  -3.07178 2.12790e-03
## ENSG00000273259   8.86088        3.85950  1.071082   3.60336 3.14126e-04
##                        padj
##                   <numeric>
## ENSG00000003402 5.47082e-11
## ENSG00000004799 3.59682e-02
## ENSG00000004846 4.93426e-02
## ENSG00000005471 3.77235e-02
## ENSG00000006788 2.50954e-02
## ...                     ...
## ENSG00000272841 1.25228e-03
## ENSG00000272870 3.29469e-06
## ENSG00000273162 3.26716e-02
## ENSG00000273179 1.93206e-02
## ENSG00000273259 4.05086e-03

#Write differentially expressed genes into a csv file
write.table(DE_genes, file = "s14682_De_genes.csv", sep = ',')

## Above list of differentially expressed genes are retrieved based on the
comparison
## between treated and untreated samples using DESeq2 package in R.
## The significance threshold (padj < 0.05) can be adjusted based on the
specific analysis requirements.

# Visualize the data - MA plot
# Plots the log-fold-change between experimental groups against the mean
expression across all the samples for each gene.
plotMA(res)
```



```
#Q3. Identify the subgroups of differentially expressed genes by hierarchical
clustering. - get the
#Clustering image, Members of each group

### Hierarchical clustering on differentially expressed genes

# Standardize/normalize the data
DE_genes = scale(DE_genes)

# Gete the transpose
# DE_genes = t(DE_genes)

# Obtain the distance matrix
dist_matrix_genes = dist(DE_genes)

# Using of 'hclust()' function
hc_genes = hclust(dist_matrix_genes, method = "ward.D")

# Visualize the Dendogram using the 'plot()' function
plot(hc_genes, main = "Dendrogram of Hierarchical Clustering")


# Extract cluster assignments using 'cutree()' function
# Specified desired no. of clusters(e. g. k = 2; treated, untreated)
```

```
cluster_assignments_genes = cutree(hc_genes, k = 2)

# devide the 2 clusters using 2 rectangles for the visualization
rect.hclust(hc_genes,k=2,border = "red")
```



Code

```
# Extract members of each gene cluster
group_members <- lapply(unique(cluster_assignments_genes), function(group) {
  genes_in_group <- rownames(gene_expr_genes)[cluster_assignments_genes == group]
  return(genes_in_group)
})
# Display members of each group
print("Member genes of each group:")
for (i in seq_along(group_members)) {
  cat("Group", i, ": ", paste(group_members[[i]], collapse = ", "), "\n")
}
```

Console output

```
> # Extract members of each gene cluster

> group_members <- lapply(unique(cluster_assignments_genes), function(group) {

+   genes_in_group <- rownames(gene_expr_genes)[cluster_assignments_genes == group]

+   return(genes_in_group)

+ })

> # Display members of each group

> print("Member genes of each group:")

[1] "Member genes of each group:"

> for (i in seq_along(group_members)) {

+   cat("Group", i, ": ", paste(group_members[[i]], collapse = ", "), "\n")

+ }

Group 1 :  ENSG00000003402, ENSG00000008256, ENSG00000008311,
ENSG00000009413, ENSG00000011198, ENSG00000020577, ENSG00000023909,
ENSG00000025708, ENSG00000035664, ENSG00000048540, ENSG00000057657,
ENSG00000060718, ENSG00000067082, ENSG00000067798, ENSG00000068383,
ENSG00000068831, ENSG00000069431, ENSG00000070388, ENSG00000070404,
ENSG00000071282, ENSG00000072163, ENSG00000072571, ENSG00000072958,
ENSG00000074590, ENSG00000074660, ENSG00000077684, ENSG00000077943,
ENSG00000078053, ENSG00000079691, ENSG00000081052, ENSG00000081320,
ENSG00000083223, ENSG00000083290, ENSG00000084090, ENSG00000085117,
ENSG00000087448, ENSG00000095637, ENSG00000096060, ENSG00000097096,
ENSG00000099204, ENSG00000099337, ENSG00000099840, ENSG00000099849,
ENSG00000099860, ENSG00000100033, ENSG00000100206, ENSG00000100242,
ENSG00000100767, ENSG00000101342, ENSG00000101347, ENSG00000102466,
ENSG00000102554, ENSG00000102760, ENSG00000102804, ENSG00000102996,
ENSG00000103064, ENSG00000103175, ENSG00000103196, ENSG00000105835,
ENSG00000105889, ENSG00000106123, ENSG00000106617, ENSG00000107104,
ENSG00000107562, ENSG00000107796, ENSG00000107968, ENSG00000108387,
ENSG00000108604, ENSG00000108821, ENSG00000108950, ENSG00000108960,
ENSG00000109861, ENSG00000109906, ENSG00000110756, ENSG00000111859,
ENSG00000112936, ENSG00000114098, ENSG00000114270, ENSG00000115419,
ENSG00000115828, ENSG00000116194, ENSG00000116285, ENSG00000116675,
ENSG00000116962, ENSG00000117479, ENSG00000118257, ENSG00000118507,
ENSG00000118689, ENSG00000119138, ENSG00000119139, ENSG00000119508,
ENSG00000119711, ENSG00000120129, ENSG00000120162, ENSG00000122035,
ENSG00000123358, ENSG00000123562, ENSG00000123685, ENSG00000124151,
```

ENSG00000124374, ENSG00000124440, ENSG00000125148, ENSG00000126803, ENSG00000127324, ENSG00000127954, ENSG00000128045, ENSG00000128311, ENSG00000128699, ENSG00000128923, ENSG00000130066, ENSG00000131386, ENSG00000131459, ENSG00000131979, ENSG00000132170, ENSG00000132518, ENSG00000132970, ENSG00000133142, ENSG00000133401, ENSG00000133816, ENSG00000134243, ENSG00000134294, ENSG00000134686, ENSG00000135362, ENSG00000135604, ENSG00000135678, ENSG00000135821, ENSG00000135917, ENSG00000136237, ENSG00000136383, ENSG00000136436, ENSG00000136478, ENSG00000136546, ENSG00000137393, ENSG00000137672, ENSG00000137673, ENSG00000137767, ENSG00000137801, ENSG00000137869, ENSG00000137880, ENSG00000137959, ENSG00000137962, ENSG00000138073, ENSG00000138074, ENSG00000138166, ENSG00000138356, ENSG00000138483, ENSG00000138615, ENSG00000138678, ENSG00000138829, ENSG00000139132, ENSG00000140511, ENSG00000140545, ENSG00000140807, ENSG00000141150, ENSG00000141298, ENSG00000141401, ENSG00000142871, ENSG00000143127, ENSG00000143869, ENSG00000143878, ENSG00000144362, ENSG00000145244, ENSG00000145390, ENSG00000145569, ENSG00000145675, ENSG00000146122, ENSG00000146373, ENSG00000147027, ENSG00000147119, ENSG00000147576, ENSG00000148120, ENSG00000148175, ENSG00000149218, ENSG00000149591, ENSG00000150907, ENSG00000150938, ENSG00000151690, ENSG00000151726, ENSG00000152463, ENSG00000152583, ENSG00000152779, ENSG00000153207, ENSG00000153904, ENSG00000154127, ENSG00000154262, ENSG00000154734, ENSG00000154736, ENSG00000154930, ENSG00000155324, ENSG00000156675, ENSG00000156804, ENSG00000157150, ENSG00000157152, ENSG00000157214, ENSG00000157510, ENSG00000157514, ENSG00000157617, ENSG00000158246, ENSG00000158716, ENSG00000158813, ENSG00000159212, ENSG00000160200, ENSG00000160256, ENSG00000161267, ENSG00000161647, ENSG00000162407, ENSG00000162426, ENSG00000162614, ENSG00000162616, ENSG00000162630, ENSG00000162772, ENSG00000162878, ENSG00000162998, ENSG00000163083, ENSG00000163110, ENSG00000163171, ENSG00000163251, ENSG00000163378, ENSG00000163431, ENSG00000163513, ENSG00000163661, ENSG00000163697, ENSG00000163803, ENSG00000163884, ENSG00000164104, ENSG00000164105, ENSG00000164125, ENSG00000164292, ENSG00000164330, ENSG00000164442, ENSG00000164647, ENSG00000165030, ENSG00000165507, ENSG00000165644, ENSG00000165899, ENSG00000165995, ENSG00000166260, ENSG00000166741, ENSG00000166825, ENSG00000166979, ENSG00000167191, ENSG00000167549, ENSG00000167641, ENSG00000167645, ENSG00000168309, ENSG00000168481, ENSG00000168556, ENSG00000168621, ENSG00000168646, ENSG00000168994, ENSG00000169031, ENSG00000169218, ENSG00000169271, ENSG00000169715, ENSG00000169738, ENSG00000169750, ENSG00000169908, ENSG00000170214, ENSG00000170323, ENSG00000170485, ENSG00000171793, ENSG00000171819, ENSG00000172260, ENSG00000172403, ENSG00000172465, ENSG00000173838, ENSG00000173918, ENSG00000174306, ENSG00000174437, ENSG00000174680, ENSG00000174697, ENSG00000174944, ENSG00000175471, ENSG00000175741, ENSG00000175946, ENSG00000176928, ENSG00000176971, ENSG00000177283, ENSG00000177575, ENSG00000177666, ENSG00000177674, ENSG00000178015, ENSG00000178723, ENSG00000179094, ENSG00000179294, ENSG00000179300, ENSG00000179593, ENSG00000179820, ENSG00000179862, ENSG00000180672, ENSG00000181061, ENSG00000182552, ENSG00000182836, ENSG00000183044, ENSG00000184156, ENSG00000184307, ENSG00000185022, ENSG00000185112, ENSG00000185432, ENSG00000185813, ENSG00000185950, ENSG00000186575, ENSG00000187193, ENSG00000187288, ENSG00000187498, ENSG00000188916, ENSG00000189221, ENSG00000196507, ENSG00000196569, ENSG00000196616, ENSG00000196850, ENSG00000196975, ENSG00000197301, ENSG00000197312, ENSG00000197381, ENSG00000198108, ENSG00000198431, ENSG00000198624, ENSG00000203685, ENSG00000205364, ENSG00000206190, ENSG00000206538, ENSG00000211445, ENSG00000211448, ENSG00000213160, ENSG00000213626, ENSG00000213639, ENSG00000213763, ENSG00000214274, ENSG00000214944, ENSG00000215481, ENSG00000219565, ENSG00000221869, ENSG00000221968, ENSG00000223401, ENSG00000224080, ENSG00000224468, ENSG00000225313, ENSG00000226121, ENSG00000226950, ENSG00000229644, ENSG00000229647, ENSG00000230018, ENSG00000231246, ENSG00000233117, ENSG00000235927, ENSG00000237697, ENSG00000237928, ENSG00000240445, ENSG00000240859, ENSG00000241399, ENSG00000242539, ENSG00000243244, ENSG00000244490, ENSG00000245812, ENSG00000246430, ENSG00000247311, ENSG00000248144, ENSG00000248187, ENSG00000249364, ENSG00000250899, ENSG00000250934, ENSG00000250978, ENSG00000253139, ENSG00000253276, ENSG00000253368, ENSG00000253833, ENSG00000254109, ENSG00000254254, ENSG00000254842, ENSG00000254851, ENSG00000258016, ENSG00000259426, ENSG00000260802, ENSG00000260841, ENSG00000261468, ENSG00000261490, ENSG00000261589, ENSG00000261685, ENSG00000264868, ENSG00000267480, ENSG00000267669, ENSG00000268894, ENSG00000268913, ENSG00000269289, ENSG00000269728, ENSG00000270689, ENSG00000272870, ENSG00000273259

Group 2 : ENSG00000004799, ENSG00000004846, ENSG00000005471, ENSG00000006788, ENSG00000012048, ENSG00000013293, ENSG00000013297, ENSG00000015520, ENSG00000016391, ENSG00000019186, ENSG00000021645, ENSG00000025423, ENSG00000028277, ENSG00000040731, ENSG00000041515, ENSG00000046653, ENSG00000049246, ENSG00000049759, ENSG00000054938, ENSG00000055163, ENSG00000056736, ENSG00000061337, ENSG00000064201, ENSG00000064309, ENSG00000065809, ENSG00000066468, ENSG00000069535, ENSG00000070808, ENSG00000070882, ENSG00000073756, ENSG00000075213, ENSG00000075240, ENSG00000077063, ENSG00000078114, ENSG00000079101, ENSG00000079435, ENSG00000079462, ENSG00000082126, ENSG00000084710, ENSG00000088756, ENSG00000089041, ENSG00000091262, ENSG00000091428, ENSG00000091831, ENSG00000092621, ENSG00000092853, ENSG00000092969, ENSG00000095585, ENSG00000099194, ENSG00000099998, ENSG00000100292, ENSG00000100302, ENSG00000100592, ENSG00000100739, ENSG00000100784, ENSG00000101255, ENSG00000101265, ENSG00000101825, ENSG00000101938,

ENSG00000102385, ENSG00000102524, ENSG00000102935, ENSG00000102984,
ENSG00000103257, ENSG00000103485, ENSG00000103647, ENSG00000103710,
ENSG00000103742, ENSG00000104894, ENSG00000104951, ENSG00000105486,
ENSG00000105516, ENSG00000105664, ENSG00000105711, ENSG00000105989,
ENSG00000106003, ENSG00000106034, ENSG00000106484, ENSG00000106976,
ENSG00000107611, ENSG00000107731, ENSG00000107821, ENSG00000108602,
ENSG00000108684, ENSG00000108700, ENSG00000108830, ENSG00000109625,
ENSG00000109674, ENSG00000109689, ENSG00000109881, ENSG00000110203,
ENSG00000110900, ENSG00000111110, ENSG00000111728, ENSG00000111816,
ENSG00000112137, ENSG00000112218, ENSG00000112715, ENSG00000112773,
ENSG00000112837, ENSG00000114670, ENSG00000116106, ENSG00000116299,
ENSG00000116584, ENSG00000116690, ENSG00000116711, ENSG00000116991,
ENSG00000117152, ENSG00000117461, ENSG00000117600, ENSG00000119514,
ENSG00000119630, ENSG00000119703, ENSG00000119714, ENSG00000120837,
ENSG00000120899, ENSG00000121621, ENSG00000122641, ENSG00000122679,
ENSG00000122877, ENSG00000122966, ENSG00000123405, ENSG00000123610,
ENSG00000123612, ENSG00000123689, ENSG00000124134, ENSG00000124249,
ENSG00000124466, ENSG00000124762, ENSG00000124766, ENSG00000125398,
ENSG00000125657, ENSG00000125848, ENSG00000125965, ENSG00000126016,
ENSG00000126785, ENSG00000126860, ENSG00000126861, ENSG00000126878,
ENSG00000126882, ENSG00000126950, ENSG00000127083, ENSG00000127589,
ENSG00000127824, ENSG00000128165, ENSG00000128262, ENSG00000128285,
ENSG00000128342, ENSG00000128510, ENSG00000128594, ENSG00000128606,
ENSG00000128917, ENSG00000129048, ENSG00000129270, ENSG00000129467,
ENSG00000130487, ENSG00000130513, ENSG00000130592, ENSG00000131242,
ENSG00000131389, ENSG00000131730, ENSG00000131771, ENSG00000132321,
ENSG00000132326, ENSG00000132334, ENSG00000132622, ENSG00000132854,
ENSG00000132965, ENSG00000133069, ENSG00000133216, ENSG00000134070,
ENSG00000134253, ENSG00000134259, ENSG00000134321, ENSG00000134363,
ENSG00000134376, ENSG00000135069, ENSG00000135472, ENSG00000136267,
ENSG00000136999, ENSG00000137266, ENSG00000137331, ENSG00000137642,
ENSG00000137872, ENSG00000138135, ENSG00000138311, ENSG00000138316,
ENSG00000138669, ENSG00000138735, ENSG00000139055, ENSG00000139269,
ENSG00000139354, ENSG00000140105, ENSG00000140600, ENSG00000141469,
ENSG00000143226, ENSG00000143320, ENSG00000143344, ENSG00000143494,
ENSG00000143507, ENSG00000143786, ENSG00000143891, ENSG00000144369,
ENSG00000144648, ENSG00000144891, ENSG00000145242, ENSG00000145506,
ENSG00000145632, ENSG00000145685, ENSG00000145777, ENSG00000145861,
ENSG00000145911, ENSG00000146006, ENSG00000146250, ENSG00000146592,
ENSG00000147655, ENSG00000147883, ENSG00000148541, ENSG00000148677,
ENSG00000148848, ENSG00000149256, ENSG00000149403, ENSG00000149488,
ENSG00000149548, ENSG00000149633, ENSG00000150594, ENSG00000150636,
ENSG00000152495, ENSG00000152580, ENSG00000154263, ENSG00000154310,
ENSG00000154856, ENSG00000154864, ENSG00000155011, ENSG00000155130,

ENSG00000155749, ENSG00000155897, ENSG00000155962, ENSG00000157368, ENSG00000157578, ENSG00000158125, ENSG00000158806, ENSG00000159023, ENSG00000159167, ENSG00000159200, ENSG00000159713, ENSG00000160097, ENSG00000160145, ENSG00000160223, ENSG00000160460, ENSG00000161381, ENSG00000162493, ENSG00000162496, ENSG00000162643, ENSG00000162692, ENSG00000163017, ENSG00000163072, ENSG00000163485, ENSG00000163491, ENSG00000163823, ENSG00000164070, ENSG00000164106, ENSG00000164122, ENSG00000164142, ENSG00000164171, ENSG00000164483, ENSG00000164484, ENSG00000164619, ENSG00000164761, ENSG00000165072, ENSG00000165105, ENSG00000165125, ENSG00000165244, ENSG00000165272, ENSG00000165388, ENSG00000165495, ENSG00000165891, ENSG00000165895, ENSG00000166292, ENSG00000166473, ENSG00000166592, ENSG00000166670, ENSG00000166762, ENSG00000166793, ENSG00000167552, ENSG00000167642, ENSG00000167771, ENSG00000167992, ENSG00000168398, ENSG00000168811, ENSG00000168918, ENSG00000169129, ENSG00000169297, ENSG00000169744, ENSG00000169855, ENSG00000170624, ENSG00000170647, ENSG00000170775, ENSG00000170873, ENSG00000170989, ENSG00000171033, ENSG00000171132, ENSG00000171227, ENSG00000171385, ENSG00000171502, ENSG00000171509, ENSG00000171617, ENSG00000171817, ENSG00000171877, ENSG00000172399, ENSG00000172497, ENSG00000172602, ENSG00000172738, ENSG00000172828, ENSG00000172955, ENSG00000172986, ENSG00000173083, ENSG00000173110, ENSG00000173114, ENSG00000173320, ENSG00000173947, ENSG00000175130, ENSG00000175197, ENSG00000175489, ENSG00000175538, ENSG00000175928, ENSG00000176293, ENSG00000176771, ENSG00000176909, ENSG00000177570, ENSG00000177606, ENSG00000177614, ENSG00000178038, ENSG00000178184, ENSG00000178662, ENSG00000178695, ENSG00000178734, ENSG00000179082, ENSG00000179242, ENSG00000179388, ENSG00000180139, ENSG00000181467, ENSG00000181634, ENSG00000182010, ENSG00000182379, ENSG00000182575, ENSG00000182580, ENSG00000182732, ENSG00000183092, ENSG00000183160, ENSG00000183454, ENSG00000183496, ENSG00000183508, ENSG00000183801, ENSG00000183876, ENSG00000184564, ENSG00000184916, ENSG00000185338, ENSG00000185745, ENSG00000185972, ENSG00000186198, ENSG00000186314, ENSG00000186469, ENSG00000186998, ENSG00000187479, ENSG00000188176, ENSG00000188312, ENSG00000188501, ENSG00000188536, ENSG00000188596, ENSG00000189007, ENSG00000196155, ENSG00000196196, ENSG00000196230, ENSG00000196476, ENSG00000196511, ENSG00000196517, ENSG00000196932, ENSG00000196950, ENSG00000197046, ENSG00000197210, ENSG00000197594, ENSG00000197943, ENSG00000198203, ENSG00000198542, ENSG00000198691, ENSG00000198944, ENSG00000198947, ENSG00000203722, ENSG00000203727, ENSG00000203943, ENSG00000205208, ENSG00000205213, ENSG00000205978, ENSG00000206052, ENSG00000206172, ENSG00000206561, ENSG00000211574, ENSG00000213402, ENSG00000213420, ENSG00000213493, ENSG00000214212, ENSG00000214814, ENSG00000215018, ENSG00000215386, ENSG00000220563, ENSG00000221994, ENSG00000223458, ENSG00000223764, ENSG00000223802, ENSG00000223811, ENSG00000223820, ENSG00000223949, ENSG00000224124, ENSG00000225032, ENSG00000225217, ENSG00000225383, ENSG00000225415, ENSG00000225783, ENSG00000226887, ENSG00000227051, ENSG00000227120, ENSG00000227268, ENSG00000228798, ENSG00000229116, ENSG00000229474, ENSG00000230417, ENSG00000231574, ENSG00000232871, ENSG00000235109, ENSG00000235513, ENSG00000235649, ENSG00000235959, ENSG00000237886, ENSG00000240291, ENSG00000240498, ENSG00000240764, ENSG00000240809, ENSG00000241990, ENSG00000243742, ENSG00000244301, ENSG00000246763, ENSG00000250657, ENSG00000253540, ENSG00000254726, ENSG00000254987, ENSG00000256340, ENSG00000256616, ENSG00000258457, ENSG00000259319, ENSG00000259448, ENSG00000259583, ENSG00000259673, ENSG00000259886, ENSG00000260230, ENSG00000260396, ENSG00000260455, ENSG00000260807, ENSG00000261121, ENSG00000261425, ENSG00000261452, ENSG00000261597, ENSG00000261827, ENSG00000263567, ENSG00000264007, ENSG00000264745, ENSG00000267339, ENSG00000267528, ENSG00000267683, ENSG00000269959, ENSG00000270093, ENSG00000270605, ENSG00000271930, ENSG00000272168, ENSG00000272341, ENSG00000272349, ENSG00000272356, ENSG00000272384, ENSG00000272744, ENSG00000272796, ENSG00000272841, ENSG00000273162, ENSG00000273179