Expression Data Clustering and Analysis of Expression Clusters

BMEG 310 Final Report Code

Laura Ing (55616957), Benjamin Green (99276917), Aadesh Mehra (39288733)

```
library(tidyverse)
library(ggplot2)
library(ggbiplot)
library(RColorBrewer)
library("survival")
library("survminer")
library(DESeq2)
library("AnnotationDbi")
library("org.Hs.eg.db")
library(pathview)
library(gage)
library(gageData)
library(pheatmap)
```

```
# Load Data
data.clinical <- read.delim("data/data_clinical_patient.txt",
                            sep = "\t", header = TRUE, comment.char = "#")
data.mutation <- read.delim("data/data_mutations.txt",
                            sep = "\t", header = TRUE, comment.char = "#")
data.expression <- read.delim("data/RNAseq_BRCA.csv",
                              sep = ",", header = TRUE, comment.char = "#")
```

```
# ------------- GET LIST OF ALL PATIENTS WITH FULL DATA -------------

# Build the patient-mutation matrix
mutation.patients <- unique(substr(data.mutation$Tumor_Sample_Barcode, 1, 12))
clinical.patients <- data.clinical$PATIENT_ID
expression.patients <- gsub("\\.", "-",
                            substr(colnames(data.expression), 1, 12)[-1])

# Get the patients that we have full data for
unique.patients.full.data <- Reduce(intersect,
                                    list(mutation.patients,
                                         clinical.patients,
                                         expression.patients))
```

```
# ------------- CLEAN AND FORMAT EXPRESSION DATA -------------
# Set gene IDs as rownames
rownames(data.expression) <- data.expression[, 1]
# Remove excess gene IDs column
data.expression <- data.expression[, -1]

# Clean up patient tags for expression data
patient.ids <- colnames(data.expression)
patient.ids.shortened <- substr(patient.ids, 1, 12)
patient.ids.shortened.sub <- gsub("\\.", "-", patient.ids.shortened)
colnames(data.expression) <- patient.ids.shortened.sub
```

```r
# Get the data from patients with full data
data.expression <- data.expression[, unique.patients.full.data]
```

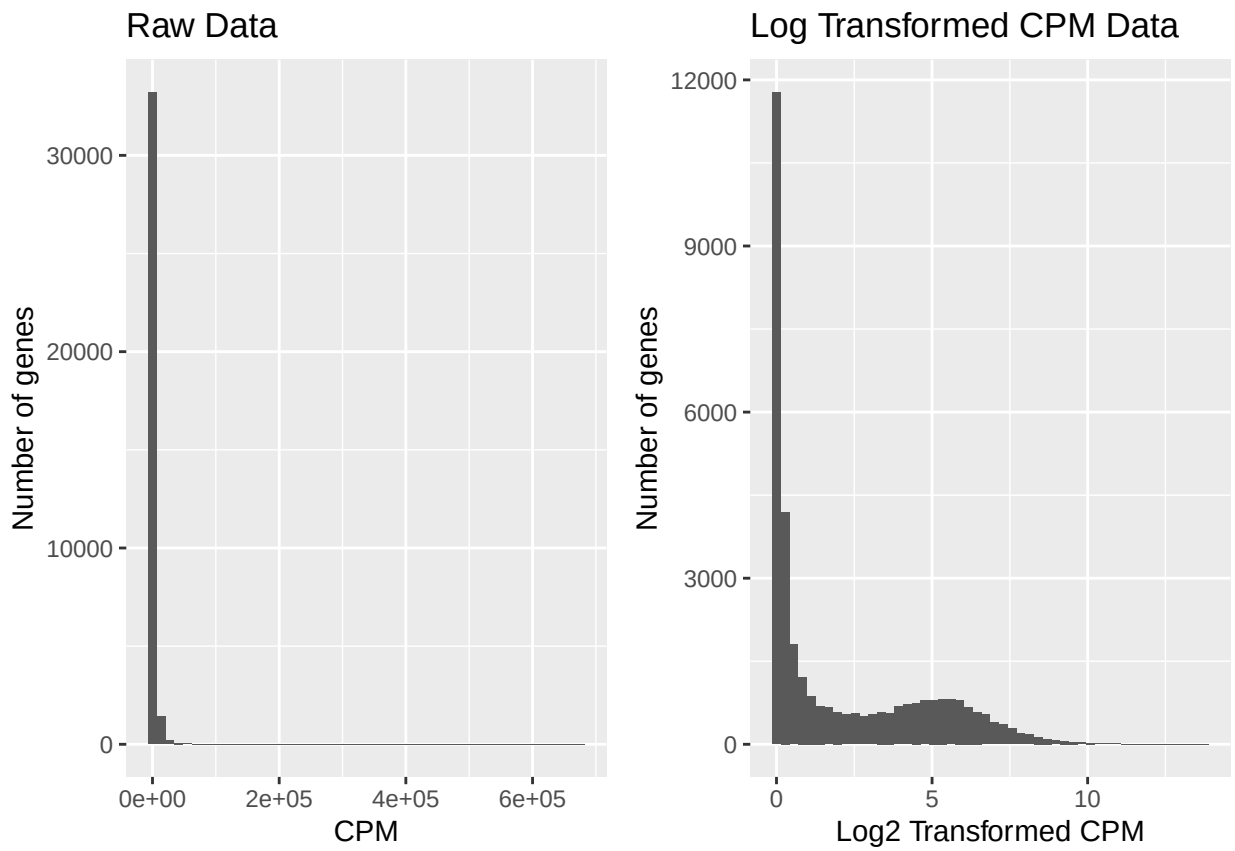
```r
# Filter out genes that have less counts than the total number of patients
# (1006)across all samples
counts <- data.expression[rowSums(data.expression) > 1006, ]
counts.df <- as.data.frame(counts)
```

```r
# ------------- NORMALIZE EXPRESSION DATA -------------
# Perform CPM normalization
cpm <- counts.df * 10^6  / colSums(counts.df)
```

```r
# Plot raw versus log-transformed data for a single patient
p1 <- ggplot(data = counts.df, aes(x = counts.df$`TCGA-3C-AAAU`)) +
  geom_histogram(bins = 50) +
  labs(x = "CPM", title = "Raw Data",
       y = "Number of genes")

log2.cpm.df <- as.data.frame(log(cpm + 1, base = 2))
p2 <- ggplot(data = log2.cpm.df, aes(x = log2.cpm.df$`TCGA-3C-AAAU`)) +
  geom_histogram(bins = 50) +
  labs(x = "Log2 Transformed CPM", title = "Log Transformed CPM Data",
       y = "Number of genes")

library("gridExtra")
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:Biobase':
##
##     combine

## The following object is masked from 'package:BiocGenerics':
##
##     combine

## The following object is masked from 'package:dplyr':
##
##     combine
```

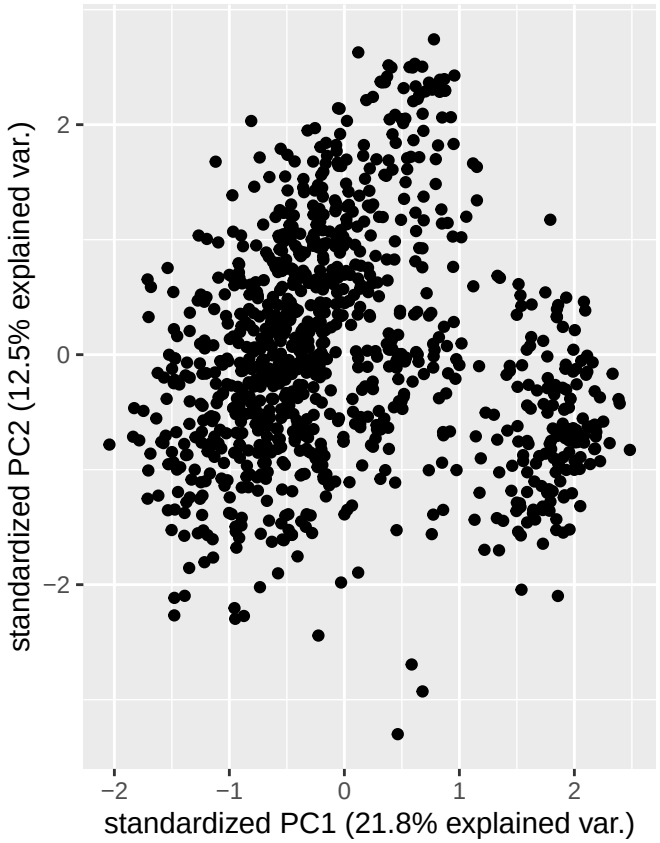
```r
grid.arrange(p1, p2, ncol = 2)
```



Above, you can see the effect of log-transforming the raw RNA-seq counts data. Before using PCA on the expression data, you can see that log-transforming the counts makes it more like a Gaussian distribution. However, most of the data is still centered around 0. This means that the expression of most genes is very low (close to 0).

```
# Log-transform the CPM counts
log.cpm <- log(cpm + 1, base = 2)
# Select the top 100 variable genes
log.cpm.t <- as.data.frame(t(log.cpm))
most.variable.genes <- order(sapply(log.cpm.t, sd), decreasing = TRUE)[1:100]
```

```
# ---------- PCA ON CPM NORMALIZED LOG TRANSFORMED EXPRESSION DATA ----------
# Perform PCA Analysis on the expression data
log.cpm.genes <- as.data.frame(t(log.cpm[most.variable.genes, ]))
log.cpm.pca <- prcomp(log.cpm.genes, scale = TRUE, center = TRUE)
ggbiplot(log.cpm.pca, var.axes = FALSE, ellipse = TRUE)
```



As seen in the above PCA plot, the PCs do not distinctly separate the data. However, you can see about two groups split by PC1.

```
summary(log.cpm.pca)
```

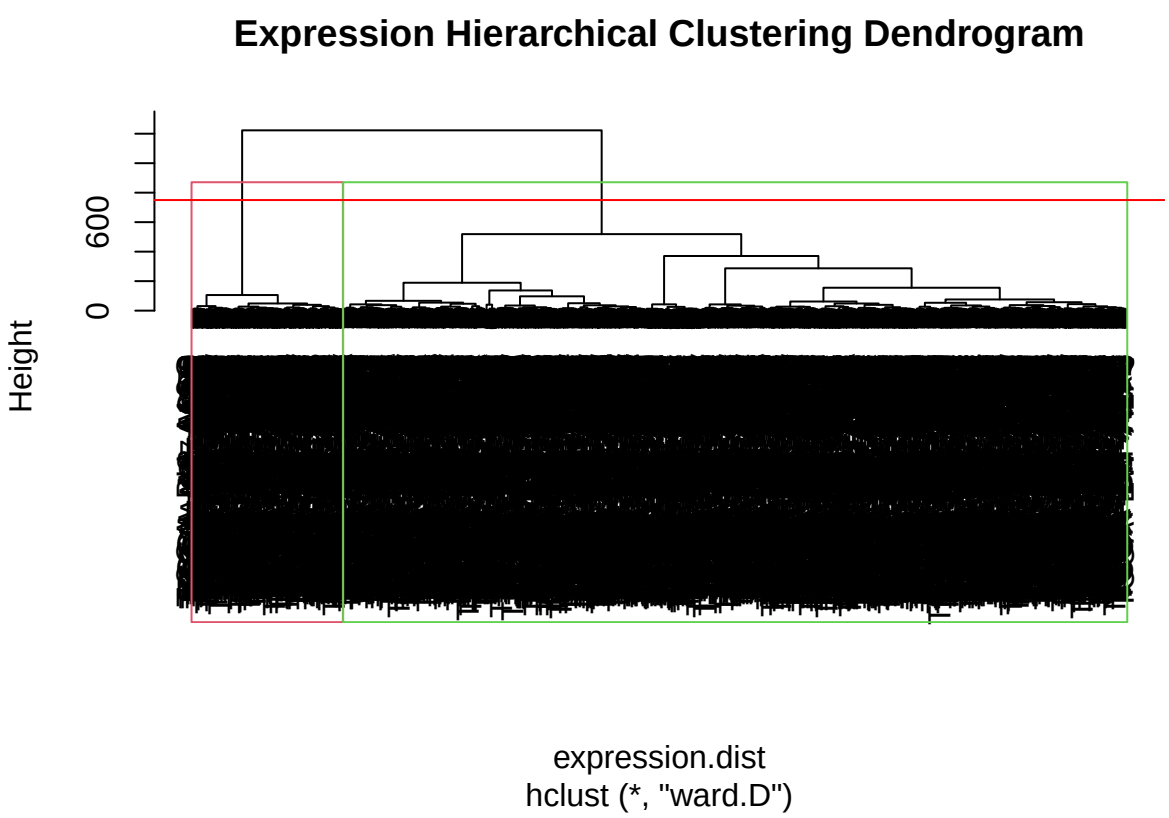
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     4.6672 3.5409 2.62263 2.26420 2.09891 1.67275 1.57924
## Proportion of Variance 0.2178 0.1254 0.06878 0.05127 0.04405 0.02798 0.02494
## Cumulative Proportion  0.2178 0.3432 0.41199 0.46325 0.50731 0.53529 0.56023
##                           PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     1.4800 1.36889 1.33576 1.27007 1.21900 1.17761 1.10857
## Proportion of Variance 0.0219 0.01874 0.01784 0.01613 0.01486 0.01387 0.01229
## Cumulative Proportion  0.5821 0.60087 0.61872 0.63485 0.64971 0.66357 0.67586
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     1.08513 1.07021 1.04043 1.00769 0.99286 0.95714 0.94062
## Proportion of Variance 0.01178 0.01145 0.01083 0.01015 0.00986 0.00916 0.00885
## Cumulative Proportion  0.68764 0.69909 0.70992 0.72007 0.72993 0.73909 0.74794
##                           PC22    PC23    PC24   PC25    PC26    PC27    PC28
## Standard deviation     0.91832 0.90385 0.89022 0.8776 0.87023 0.84178 0.83240
## Proportion of Variance 0.00843 0.00817 0.00792 0.0077 0.00757 0.00709 0.00693
## Cumulative Proportion  0.75637 0.76454 0.77246 0.7802 0.78774 0.79483 0.80175
##                           PC29    PC30    PC31    PC32    PC33    PC34    PC35
## Standard deviation     0.82273 0.80704 0.79247 0.77161 0.76405 0.75762 0.75637
## Proportion of Variance 0.00677 0.00651 0.00628 0.00595 0.00584 0.00574 0.00572
## Cumulative Proportion  0.80852 0.81504 0.82132 0.82727 0.83311 0.83885 0.84457
```

```
##                            PC36    PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     0.73829 0.72745 0.71560 0.70880 0.69751 0.68334 0.67741
## Proportion of Variance 0.00545 0.00529 0.00512 0.00502 0.00487 0.00467 0.00459
## Cumulative Proportion  0.85002 0.85531 0.86043 0.86546 0.87032 0.87499 0.87958
##                            PC43    PC44    PC45    PC46    PC47    PC48    PC49
## Standard deviation     0.67011 0.66380 0.65506 0.64206 0.63474 0.62156 0.61842
## Proportion of Variance 0.00449 0.00441 0.00429 0.00412 0.00403 0.00386 0.00382
## Cumulative Proportion  0.88407 0.88848 0.89277 0.89689 0.90092 0.90478 0.90861
##                            PC50    PC51    PC52    PC53    PC54    PC55    PC56
## Standard deviation     0.60531 0.60187 0.59485 0.58687 0.58260 0.57603 0.56515
## Proportion of Variance 0.00366 0.00362 0.00354 0.00344 0.00339 0.00332 0.00319
## Cumulative Proportion  0.91227 0.91589 0.91943 0.92288 0.92627 0.92959 0.93278
##                            PC57    PC58    PC59    PC60    PC61    PC62    PC63
## Standard deviation     0.55971 0.54837 0.54003 0.53585 0.51862 0.51259 0.50841
## Proportion of Variance 0.00313 0.00301 0.00292 0.00287 0.00269 0.00263 0.00258
## Cumulative Proportion  0.93592 0.93892 0.94184 0.94471 0.94740 0.95003 0.95261
##                            PC64    PC65    PC66    PC67    PC68    PC69   PC70
## Standard deviation     0.49310 0.49111 0.48099 0.47173 0.46392 0.44504 0.4360
## Proportion of Variance 0.00243 0.00241 0.00231 0.00223 0.00215 0.00198 0.0019
## Cumulative Proportion  0.95504 0.95746 0.95977 0.96199 0.96415 0.96613 0.9680
##                            PC71    PC72   PC73   PC74    PC75    PC76    PC77
## Standard deviation     0.43198 0.43030 0.4242 0.4129 0.40387 0.40226 0.39501
## Proportion of Variance 0.00187 0.00185 0.0018 0.0017 0.00163 0.00162 0.00156
## Cumulative Proportion  0.96989 0.97175 0.9735 0.9752 0.97688 0.97850 0.98006
##                            PC78    PC79    PC80    PC81    PC82    PC83    PC84
## Standard deviation     0.38273 0.37980 0.36628 0.35703 0.35119 0.34139 0.33566
## Proportion of Variance 0.00146 0.00144 0.00134 0.00127 0.00123 0.00117 0.00113
## Cumulative Proportion  0.98152 0.98297 0.98431 0.98558 0.98682 0.98798 0.98911
##                            PC85    PC86    PC87    PC88    PC89    PC90    PC91
## Standard deviation     0.33354 0.32504 0.31538 0.30254 0.28777 0.27951 0.27796
## Proportion of Variance 0.00111 0.00106 0.00099 0.00092 0.00083 0.00078 0.00077
## Cumulative Proportion  0.99022 0.99128 0.99227 0.99319 0.99402 0.99480 0.99557
##                            PC92    PC93    PC94    PC95    PC96    PC97    PC98
## Standard deviation     0.26863 0.25992 0.24892 0.22989 0.21095 0.20271 0.19739
## Proportion of Variance 0.00072 0.00068 0.00062 0.00053 0.00045 0.00041 0.00039
## Cumulative Proportion  0.99629 0.99697 0.99759 0.99811 0.99856 0.99897 0.99936
##                            PC99   PC100
## Standard deviation     0.19358 0.16291
## Proportion of Variance 0.00037 0.00027
## Cumulative Proportion  0.99973 1.00000
```

To stay standardized with the previous mutation PCA analysis, we are capturing 85% of the variance with 36 PCs.

```
# ---------- HIERARCHICAL CLUSTERING ON EXPRESSION DATA ----------
# Cluster genes based on expression of top 100 variable genes
expression.dist <- dist(log.cpm.pca$x[, 1:36])
expression.ward <- hclust(expression.dist, method = 'ward.D')
```

```
CUT_HEIGHT <- 750
cut_ward_expression <- cutree(expression.ward, h = CUT_HEIGHT)
plot(expression.ward, main = "Expression Hierarchical Clustering Dendrogram")
rect.hclust(expression.ward, h = CUT_HEIGHT, border = 2:6)
abline(h = CUT_HEIGHT, col = 'red')
```

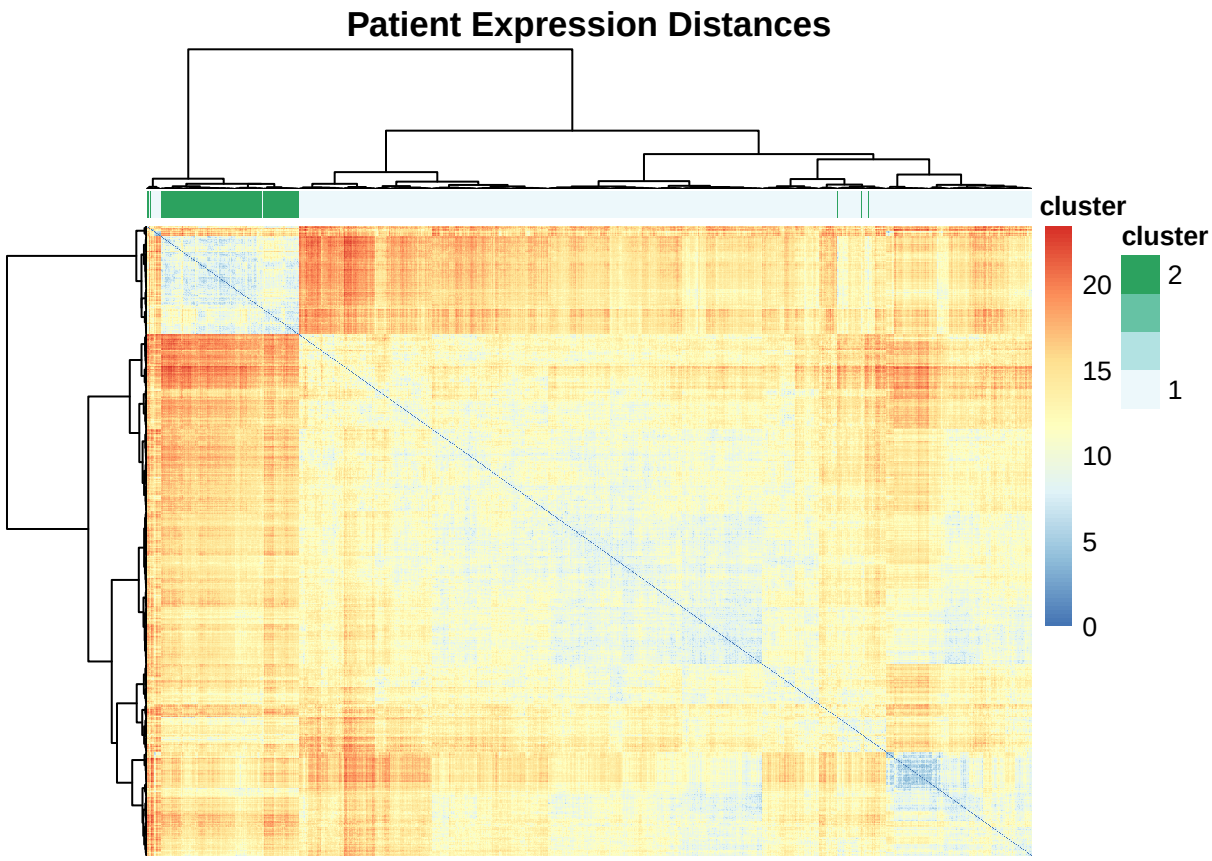


2 clusters were chosen for heirarchial clustering because the PC analysis had 2 groups. Also, 2 clusters splits the patients quite well as seen in the dendogram above (the second split is much lower in the tree). Unfortunately, one cluster is much larger than the second cluster.

```
# ------------- VISUALIZE CLUSTERS VERSUS TUMOUR-SUBTYPES -------------
# Find indices of all patients in the clinical data that have been clustered
# (1006 patients)
idx <- which(data.clinical$PATIENT_ID %in% unique.patients.full.data)

# Create a data frame containing the patient ID, their BRCA subtype and their
# cluster assignment
clust.sub.data <- data.frame(Patient = data.clinical$PATIENT_ID[idx],
                             Subtype = data.clinical$SUBTYPE[idx],
                             Clusters = cut_ward_expression)

# Visualize the BRCA suptypes within each cluster using a bar plot
ggplot(clust.sub.data, aes(x = cut_ward_expression, fill = Subtype)) +
  geom_bar(position = "dodge") +
  labs(title = "Subtypes within Clusters",
       x = "Cluster",
       y = "Count") +
  scale_fill_brewer(palette = "Set2", name = "Subtype")
```

Above the subtypes of cancer in each expression cluster is plotted. Cluster 2 includes almost exclusively the Basal subtype, and cluster 1 contains all the other subtypes (most frequently Luminal A and B).

```
# ------------- VISUALIZE EXPRESSION DISTANCE BETWEEN PATIENTS -------------
annotation <- as.data.frame(cut_ward_expression)
rownames(annotation) <- rownames(log.cpm.t)
colnames(annotation) <- c("cluster")

expression.dist.mat <- as.matrix(expression.dist)
rownames(expression.dist.mat) <- rownames(log.cpm.t)
colnames(expression.dist.mat) <- rownames(log.cpm.t)

pheatmap(expression.dist.mat,
         cluster_rows=TRUE,
         show_rownames=FALSE,
         cluster_cols=TRUE,
         show_colnames = FALSE,
         annotation_col=annotation,
         clustering_method = "ward.D",
         main = "Patient Expression Distances")
```



Above you can see a heatmap of the distance between samples based on expression data. The clusters generated from PCA analysis and then hierarchical clustering produced good clusters that have distinct expression patterns (the clusters have a large distance apart from each other).

```
# ---------- DIFFERENTIAL EXPRESSION ON EXPRESSION CLUSTERS ----------
expression.clusters <- as.data.frame(cut_ward_expression)
rownames(expression.clusters) <- rownames(log.cpm.t)
colnames(expression.clusters) <- c("cluster")
dds.expression = DESeqDataSetFromMatrix(countData=counts,
                                colData=expression.clusters,
                                design=~cluster)
```

```
dds.expression <- DESeq(dds.expression)
```

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
## -- replacing outliers and refitting for 6612 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

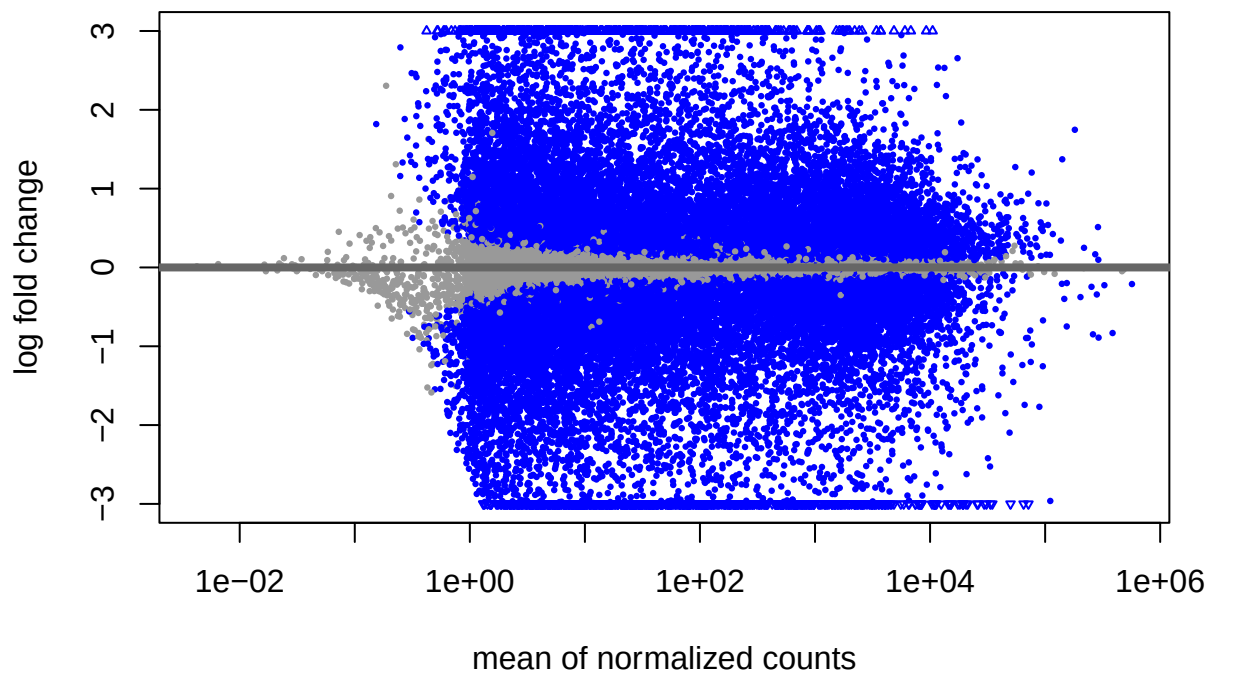
```
res.expression <- results(dds.expression)
res.expression
```

```
## log2 fold change (MLE): cluster
## Wald test p-value: cluster
## DataFrame with 34965 rows and 6 columns
##                     baseMean log2FoldChange     lfcSE      stat      pvalue
##                    <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000000003.15 3126.1978       0.516371 0.0818151   6.31144 2.76453e-10
## ENSG00000000005.6    80.4192      -1.906234 0.2143350  -8.89371 5.91006e-19
## ENSG00000000419.13 2377.3111       0.332442 0.0488410   6.80661 9.99253e-12
## ENSG00000000457.14 1580.9323      -0.380614 0.0478830  -7.94883 1.88279e-15
## ENSG00000000460.17  719.0585       0.899496 0.0631801  14.23702 5.39833e-46
## ...                      ...            ...       ...       ...         ...
## ENSG00000288658.1   23.14281      1.3369077 0.1792645  7.457740 8.80193e-14
## ENSG00000288663.1   28.67196     -0.5602812 0.0720059 -7.781047 7.19266e-15
## ENSG00000288670.1  425.30940     -0.0643582 0.0581319 -1.107107 2.68247e-01
## ENSG00000288674.1    8.08483     -0.0114023 0.0840998 -0.135581 8.92152e-01
## ENSG00000288675.1   33.14620      0.3210702 0.0939609  3.417063 6.33006e-04
##                          padj
##                     <numeric>
## ENSG00000000003.15 6.93513e-10
## ENSG00000000005.6  2.32081e-18
## ENSG00000000419.13 2.72322e-11
## ENSG00000000457.14 6.26550e-15
## ENSG00000000460.17 5.87283e-45
## ...                      ...
## ENSG00000288658.1  2.68083e-13
## ENSG00000288663.1  2.31960e-14
## ENSG00000288670.1  3.07074e-01
## ENSG00000288674.1  9.07742e-01
## ENSG00000288675.1  9.94566e-04
```

```
summary(res.expression)
```

```
##
## out of 34965 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 14308, 41%
## LFC < 0 (down)     : 13804, 39%
## outliers [1]       : 0, 0%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
plotMA(res.expression, ylim = c(-3, 3),
       main = "Differentially expressed genes comparing expression clusters 1 and 2")
```




```
# ------------ VISUALIZE HEATMAP OF DIFFERENTIALLY EXPRESSED GENES ------------
# Get top 20 DE Genes from expression clusters
expression.genes <- order(res.expression$padj, decreasing = FALSE)[1:20]
```

```
# Filter for significant genes
res.expression.sig <- subset(res.expression, padj < 0.0001)
# Select the top 10 over-expressed genes
de.genes.up <- rownames(res.expression.sig)[order(
  res.expression.sig$log2FoldChange, decreasing = TRUE)[1:10]]
de.genes.up <- which(rownames(res.expression) %in% de.genes.up)
# Select the top 10 under-expressed genes
de.genes.down <- rownames(res.expression.sig)[order(
  res.expression.sig$log2FoldChange, decreasing = FALSE)[1:10]]
de.genes.down <- which(rownames(res.expression) %in% de.genes.down)
# Bind significant genes together
de.genes <- c(de.genes.up, de.genes.down)
```

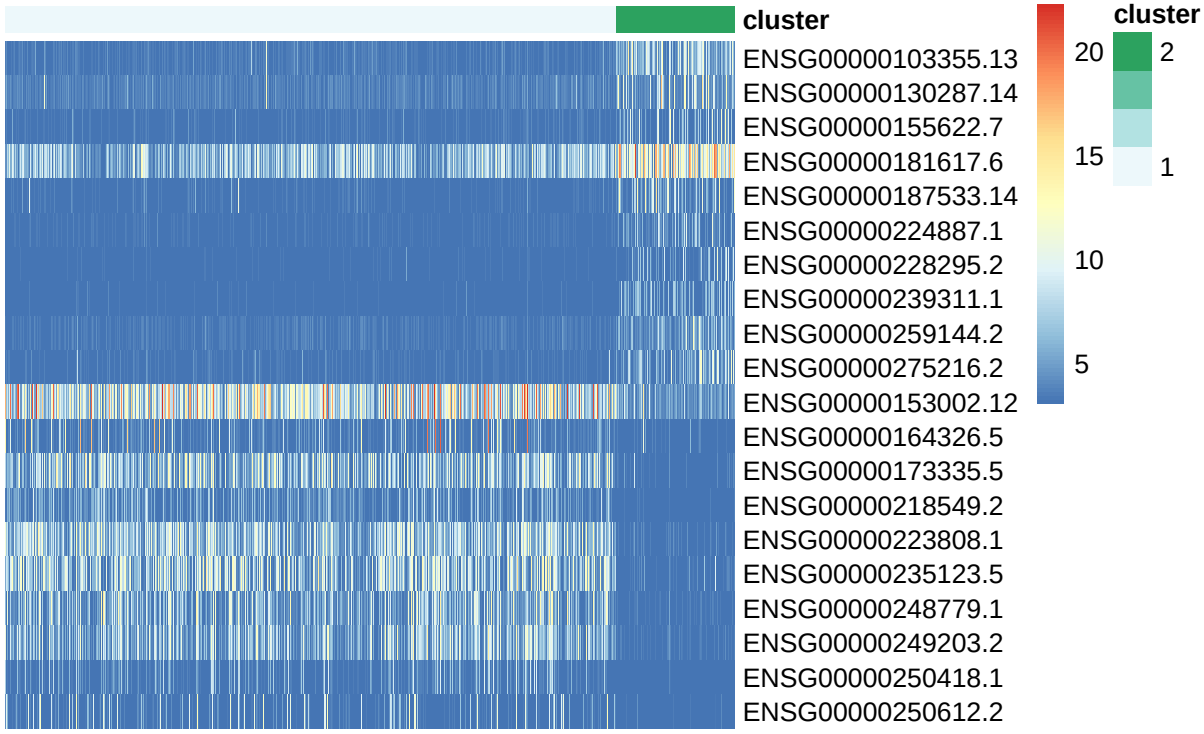
```
# Variance Stabalizing transform
expression.vsd <- vst(dds.expression, blind = TRUE)
```

```
# Plot the log2 fold expression of the top 20 differentially expressed genes
expression.mat <- assay(expression.vsd)[de.genes,]
colnames(expression.mat) <- colnames(counts)

# Get the gene symbol to plot instead of Ensembl ID
de.genes.cleaned <- sapply(strsplit(rownames(counts[de.genes,]),
                                    "\\."), "[", 1)
de.genes.sym = mapIds(org.Hs.eg.db,
                      keys=de.genes.cleaned,
                      column="SYMBOL",
                      keytype="ENSEMBL",
                      multiVals="first")
```

```
## 'select()' returned 1:1 mapping between keys and columns
```

```
# Commenting out because some gene symbols are NA
# rownames(expression.mat) <- de.genes.sym

# Order the expression df by cluster
cluster.order.df <- as.data.frame(sort(cut_ward_expression, decreasing = FALSE))
colnames(cluster.order.df) <- c(cluster)

patient.order.df <- as.data.frame(list(colnames(counts), cut_ward_expression))
colnames(patient.order.df) <- c("patient", "cluster")
patient.order.df <- patient.order.df[order(patient.order.df$cluster), ]

# Create the annotation df
annotation.df <- as.data.frame(patient.order.df$cluster)
rownames(annotation.df) <- patient.order.df$patient
colnames(annotation.df) <- c("cluster")

expression.mat <- expression.mat[, patient.order.df$patient]

library(pheatmap)
pheatmap(expression.mat,
         cluster_rows=FALSE,
         show_rownames=TRUE,
         show_colnames = FALSE,
         cluster_cols=FALSE,
         annotation_col = annotation.df,
         main = "Top 10 upregulated and top 10 downregulated significantly
         differentially expressed genes across clusters")
```

upregulated and top 10 downregulated significantly differentially expressed genes across clusters

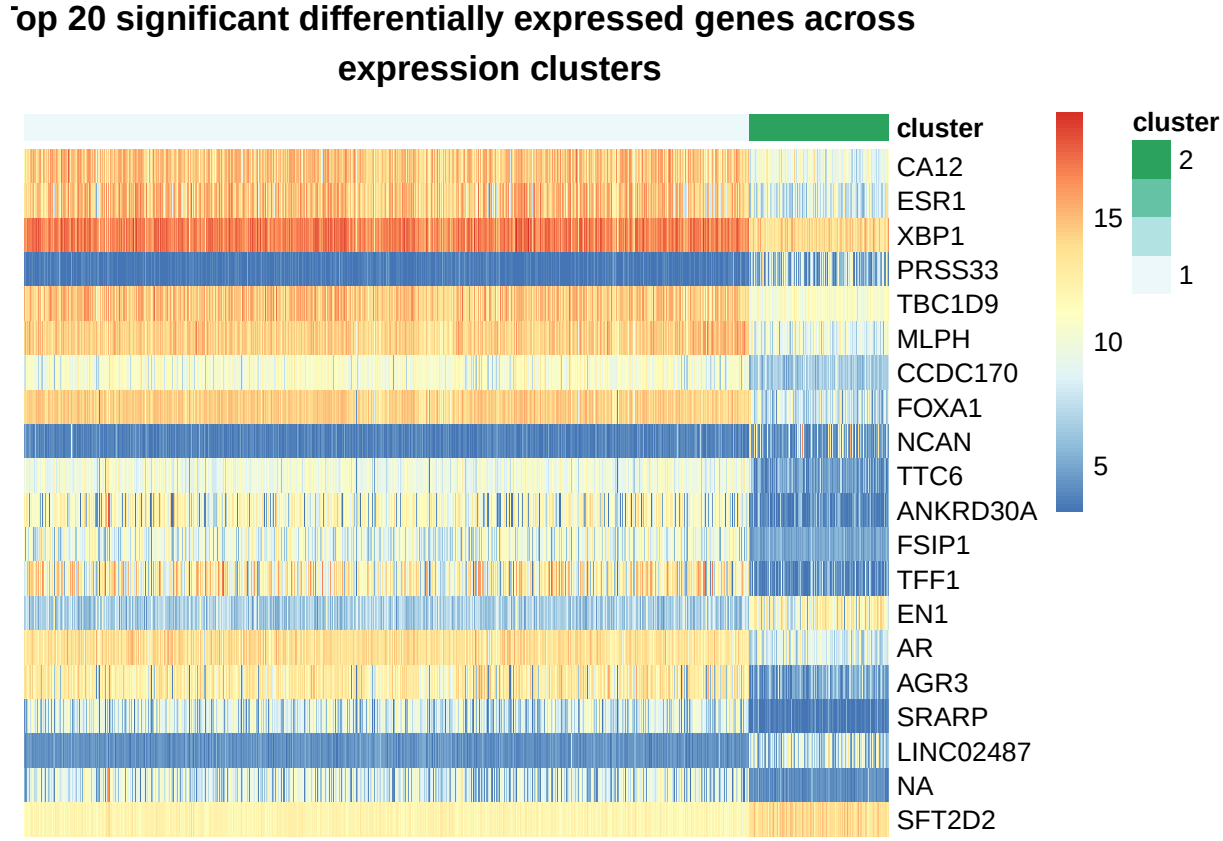



```
# Plot the expression of the top 20 differentially expressed genes
expression.vsd <- vst(dds.expression, blind = FALSE)

expression.mat <- assay(expression.vsd)[expression.genes, ]
colnames(expression.mat) = colnames(counts.df)

# Get the gene symbols instead of ENSEMBL gene IDs
expression.genes.cleaned <- sapply(strsplit(rownames(
  counts.df[expression.genes,]),
                                      "\\."), "[", 1)
expression.genes.sym = mapIds(org.Hs.eg.db,
                    keys=expression.genes.cleaned,
                    column="SYMBOL",
                    keytype="ENSEMBL",
                    multiVals="first")
```

```
## 'select()' returned 1:1 mapping between keys and columns
```

```
rownames(expression.mat) <- expression.genes.sym

# Data formatting
cluster.order.df <- as.data.frame(sort(cut_ward_expression, decreasing = FALSE))
colnames(cluster.order.df) <- c(cluster)

patient.order.df <- as.data.frame(list(colnames(counts), cut_ward_expression))
```

```r
colnames(patient.order.df) <- c("patient", "cluster")
patient.order.df <- patient.order.df[order(patient.order.df$cluster), ]

annotation.df <- as.data.frame(patient.order.df$cluster)
rownames(annotation.df) <- patient.order.df$patient
colnames(annotation.df) <- c("cluster")

# Order the matrix by the clusters
expression.mat <- expression.mat[, patient.order.df$patient]

pheatmap(expression.mat,
         cluster_rows=FALSE,
         show_rownames=TRUE,
         show_colnames = FALSE,
         cluster_cols=FALSE,
         annotation_col=annotation,
         main = "Top 20 significant differentially expressed genes across
         expression clusters")
```



Above it can be seen that clustering on the expression data generated two clusters with distinct expression patterns. This is expected as by clustering with the expression data it should achieve some level of separation.

```r
# ------------- ANNOTATE GENES -------------
gene.names.cleaned.expression <- sapply(strsplit(row.names(res.expression),
                                                 "\\."), "[", 1)
```

```
res.expression$symbol = mapIds(org.Hs.eg.db,
                     keys=gene.names.cleaned.expression,
                     column="SYMBOL",
                     keytype="ENSEMBL",
                     multiVals="first")
```

```
## 'select()' returned 1:many mapping between keys and columns
```

```
res.expression$entrez = mapIds(org.Hs.eg.db,
                       keys=gene.names.cleaned.expression,
                       column="ENTREZID",
                       keytype="ENSEMBL",
                       multiVals="first")
```

```
## 'select()' returned 1:many mapping between keys and columns
```

```
res.expression$name = mapIds(org.Hs.eg.db,
                     keys=gene.names.cleaned.expression,
                     column="GENENAME",
                     keytype="ENSEMBL",
                     multiVals="first")
```

```
## 'select()' returned 1:many mapping between keys and columns
```

```
# ------------- PATHWAY ANALYSIS ON EXPRESSION CLUSTERS -------------
data(kegg.sets.hs)
data(sigmet.idx.hs)
# Focus on signaling and metabolic pathways only
kegg.sets.hs = kegg.sets.hs[sigmet.idx.hs]
```

```
expression.fold.changes <- res.expression$log2FoldChange
names(expression.fold.changes) <- res.expression$entrez
```

```
# Gage pathway analysis
expression.kegg.res = gage(expression.fold.changes, gsets=kegg.sets.hs)
```

```
head(expression.kegg.res$less, 15)
```

```
##                                                      p.geomean stat.mean
## hsa00830 Retinol metabolism                         0.0005645007 -3.374672
## hsa00982 Drug metabolism - cytochrome P450          0.0016473146 -3.015050
## hsa00350 Tyrosine metabolism                        0.0055966047 -2.601872
## hsa00983 Drug metabolism - other enzymes            0.0094446578 -2.410636
## hsa00140 Steroid hormone biosynthesis               0.0108312296 -2.348543
## hsa03320 PPAR signaling pathway                     0.0151229935 -2.192071
## hsa00980 Metabolism of xenobiotics by cytochrome P450 0.0152925275 -2.195041
## hsa00500 Starch and sucrose metabolism              0.0174862841 -2.155624
## hsa04610 Complement and coagulation cascades        0.0201791932 -2.071716
## hsa00053 Ascorbate and aldarate metabolism          0.0205115525 -2.161637
## hsa02010 ABC transporters                           0.0284816156 -1.932508
```

```
## hsa00040 Pentose and glucuronate interconversions     0.0324649400 -1.909931
## hsa00071 Fatty acid metabolism                        0.0431405347 -1.735760
## hsa04972 Pancreatic secretion                         0.0456889334 -1.697793
## hsa00590 Arachidonic acid metabolism                  0.0487337580 -1.672281
##                                                              p.val      q.val
## hsa00830 Retinol metabolism                           0.0005645007 0.09257812
## hsa00982 Drug metabolism - cytochrome P450            0.0016473146 0.13507980
## hsa00350 Tyrosine metabolism                          0.0055966047 0.30594772
## hsa00983 Drug metabolism - other enzymes              0.0094446578 0.33638946
## hsa00140 Steroid hormone biosynthesis                 0.0108312296 0.33638946
## hsa03320 PPAR signaling pathway                       0.0151229935 0.33638946
## hsa00980 Metabolism of xenobiotics by cytochrome P450 0.0152925275 0.33638946
## hsa00500 Starch and sucrose metabolism                0.0174862841 0.33638946
## hsa04610 Complement and coagulation cascades          0.0201791932 0.33638946
## hsa00053 Ascorbate and aldarate metabolism            0.0205115525 0.33638946
## hsa02010 ABC transporters                             0.0284816156 0.42463500
## hsa00040 Pentose and glucuronate interconversions     0.0324649400 0.44368751
## hsa00071 Fatty acid metabolism                        0.0431405347 0.53282242
## hsa04972 Pancreatic secretion                         0.0456889334 0.53282242
## hsa00590 Arachidonic acid metabolism                  0.0487337580 0.53282242
##                                                      set.size         exp1
## hsa00830 Retinol metabolism                                56 0.0005645007
## hsa00982 Drug metabolism - cytochrome P450                 64 0.0016473146
## hsa00350 Tyrosine metabolism                               40 0.0055966047
## hsa00983 Drug metabolism - other enzymes                   45 0.0094446578
## hsa00140 Steroid hormone biosynthesis                      46 0.0108312296
## hsa03320 PPAR signaling pathway                            68 0.0151229935
## hsa00980 Metabolism of xenobiotics by cytochrome P450      62 0.0152925275
## hsa00500 Starch and sucrose metabolism                     42 0.0174862841
## hsa04610 Complement and coagulation cascades               65 0.0201791932
## hsa00053 Ascorbate and aldarate metabolism                 19 0.0205115525
## hsa02010 ABC transporters                                  44 0.0284816156
## hsa00040 Pentose and glucuronate interconversions          25 0.0324649400
## hsa00071 Fatty acid metabolism                             43 0.0431405347
## hsa04972 Pancreatic secretion                              96 0.0456889334
## hsa00590 Arachidonic acid metabolism                       54 0.0487337580
```

14 metabolic pathways are being downregulated in cluster 2 compared to cluster 1. These are mostly metabolic pathways without a clear connection to literature cancer signaling pathways.

```
upreg <- expression.kegg.res$greater
head(upreg, 15)
```

```
##                                                        p.geomean
## hsa04110 Cell cycle                                 9.282805e-06
## hsa04650 Natural killer cell mediated cytotoxicity  1.259109e-05
## hsa04612 Antigen processing and presentation        1.255417e-04
## hsa03008 Ribosome biogenesis in eukaryotes          2.023999e-03
## hsa04514 Cell adhesion molecules (CAMs)             3.259116e-03
## hsa04062 Chemokine signaling pathway                3.421695e-03
## hsa03030 DNA replication                            4.000221e-03
## hsa03040 Spliceosome                                4.500485e-03
## hsa03013 RNA transport                              8.023655e-03
```

```
## hsa04620 Toll-like receptor signaling pathway                       9.053310e-03
## hsa04660 T cell receptor signaling pathway                          1.116325e-02
## hsa04672 Intestinal immune network for IgA production               1.141030e-02
## hsa03050 Proteasome                                                 1.752441e-02
## hsa00601 Glycosphingolipid biosynthesis - lacto and neolacto series 1.968312e-02
## hsa04621 NOD-like receptor signaling pathway                        2.168624e-02
##                                                                        stat.mean
## hsa04110 Cell cycle                                                     4.380528
## hsa04650 Natural killer cell mediated cytotoxicity                      4.311996
## hsa04612 Antigen processing and presentation                            3.806439
## hsa03008 Ribosome biogenesis in eukaryotes                              2.950018
## hsa04514 Cell adhesion molecules (CAMs)                                 2.743176
## hsa04062 Chemokine signaling pathway                                    2.720339
## hsa03030 DNA replication                                                2.775044
## hsa03040 Spliceosome                                                    2.646668
## hsa03013 RNA transport                                                  2.426022
## hsa04620 Toll-like receptor signaling pathway                           2.393208
## hsa04660 T cell receptor signaling pathway                              2.306769
## hsa04672 Intestinal immune network for IgA production                   2.329269
## hsa03050 Proteasome                                                     2.157366
## hsa00601 Glycosphingolipid biosynthesis - lacto and neolacto series     2.123236
## hsa04621 NOD-like receptor signaling pathway                            2.047634
##                                                                             p.val
## hsa04110 Cell cycle                                                  9.282805e-06
## hsa04650 Natural killer cell mediated cytotoxicity                   1.259109e-05
## hsa04612 Antigen processing and presentation                         1.255417e-04
## hsa03008 Ribosome biogenesis in eukaryotes                           2.023999e-03
## hsa04514 Cell adhesion molecules (CAMs)                              3.259116e-03
## hsa04062 Chemokine signaling pathway                                 3.421695e-03
## hsa03030 DNA replication                                             4.000221e-03
## hsa03040 Spliceosome                                                 4.500485e-03
## hsa03013 RNA transport                                               8.023655e-03
## hsa04620 Toll-like receptor signaling pathway                        9.053310e-03
## hsa04660 T cell receptor signaling pathway                           1.116325e-02
## hsa04672 Intestinal immune network for IgA production                1.141030e-02
## hsa03050 Proteasome                                                  1.752441e-02
## hsa00601 Glycosphingolipid biosynthesis - lacto and neolacto series  1.968312e-02
## hsa04621 NOD-like receptor signaling pathway                         2.168624e-02
##                                                                             q.val
## hsa04110 Cell cycle                                                  0.001032470
## hsa04650 Natural killer cell mediated cytotoxicity                   0.001032470
## hsa04612 Antigen processing and presentation                         0.006862945
## hsa03008 Ribosome biogenesis in eukaryotes                           0.082983965
## hsa04514 Cell adhesion molecules (CAMs)                              0.092259937
## hsa04062 Chemokine signaling pathway                                 0.092259937
## hsa03030 DNA replication                                             0.092259937
## hsa03040 Spliceosome                                                 0.092259937
## hsa03013 RNA transport                                               0.146208820
## hsa04620 Toll-like receptor signaling pathway                        0.148474282
## hsa04660 T cell receptor signaling pathway                           0.155940779
## hsa04672 Intestinal immune network for IgA production                0.155940779
## hsa03050 Proteasome                                                  0.221077209
## hsa00601 Glycosphingolipid biosynthesis - lacto and neolacto series  0.230573686
## hsa04621 NOD-like receptor signaling pathway                         0.236541502
```

```
##                                                                 set.size
## hsa04110 Cell cycle                                                  124
## hsa04650 Natural killer cell mediated cytotoxicity                   117
## hsa04612 Antigen processing and presentation                          67
## hsa03008 Ribosome biogenesis in eukaryotes                            71
## hsa04514 Cell adhesion molecules (CAMs)                              128
## hsa04062 Chemokine signaling pathway                                 184
## hsa03030 DNA replication                                              36
## hsa03040 Spliceosome                                                 127
## hsa03013 RNA transport                                               148
## hsa04620 Toll-like receptor signaling pathway                         89
## hsa04660 T cell receptor signaling pathway                           107
## hsa04672 Intestinal immune network for IgA production                 44
## hsa03050 Proteasome                                                   43
## hsa00601 Glycosphingolipid biosynthesis - lacto and neolacto series   25
## hsa04621 NOD-like receptor signaling pathway                          58
##                                                                    exp1
## hsa04110 Cell cycle                                          9.282805e-06
## hsa04650 Natural killer cell mediated cytotoxicity           1.259109e-05
## hsa04612 Antigen processing and presentation                 1.255417e-04
## hsa03008 Ribosome biogenesis in eukaryotes                   2.023999e-03
## hsa04514 Cell adhesion molecules (CAMs)                      3.259116e-03
## hsa04062 Chemokine signaling pathway                         3.421695e-03
## hsa03030 DNA replication                                     4.000221e-03
## hsa03040 Spliceosome                                         4.500485e-03
## hsa03013 RNA transport                                       8.023655e-03
## hsa04620 Toll-like receptor signaling pathway                9.053310e-03
## hsa04660 T cell receptor signaling pathway                   1.116325e-02
## hsa04672 Intestinal immune network for IgA production        1.141030e-02
## hsa03050 Proteasome                                          1.752441e-02
## hsa00601 Glycosphingolipid biosynthesis - lacto and neolacto series 1.968312e-02
## hsa04621 NOD-like receptor signaling pathway                 2.168624e-02
```

Cluster 2 has 14 significant highly upregulated pathways, including cell cycle, DNA replication, and various immune cell signalling pathways.

Mutation Analysis with Expression Clusters

```r
# ------------- CLEAN MUTATION DATA -------------
# Make a new column with the cleaned patient ID (same as other datasets)
data.mutation$Tumor_Sample_Barcode_Cleaned <- substr(
  data.mutation$Tumor_Sample_Barcode, 1, 12)

# Non-coding transcript exon variant?
important_mutations <- data.mutation[
  which(data.mutation$IMPACT %in% c("HIGH", "MODERATE")), ]

# Get the important mutations for full-data patients
important_mutations_full_data <- important_mutations[
  which(important_mutations$Tumor_Sample_Barcode_Cleaned %in%
          unique.patients.full.data), ]
```

```r
# Make feature matrix
feature_mat <- table(important_mutations_full_data$Tumor_Sample_Barcode_Cleaned, important_mutations_ful

# Turn into binary matrix
feature_mat[feature_mat > 1] <- 1
```

```r
# Filter for the top mutated genes
feature_mat_filtered <- feature_mat[, colSums(feature_mat) >= 51]
```

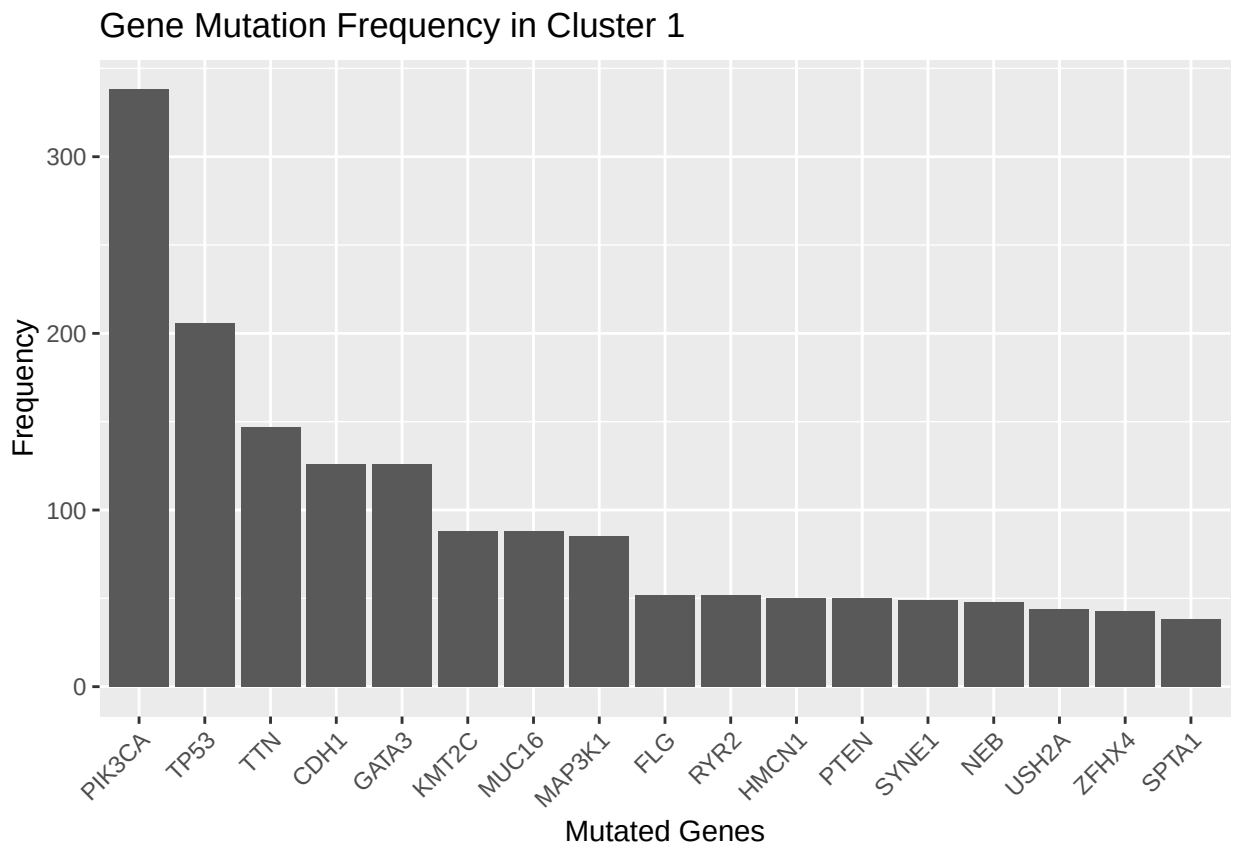
```r
# More formatting of the onco-matrix
feature_df_filtered <- as.data.frame(feature_mat_filtered)
feature_df_filtered <- pivot_wider(feature_df_filtered, names_from = Var2,
                                   values_from = Freq)
# Save the patients
patient.order <- feature_df_filtered$Var1
rownames(feature_df_filtered) <- patient.order
```

```
## Warning: Setting row names on a tibble is deprecated.
```

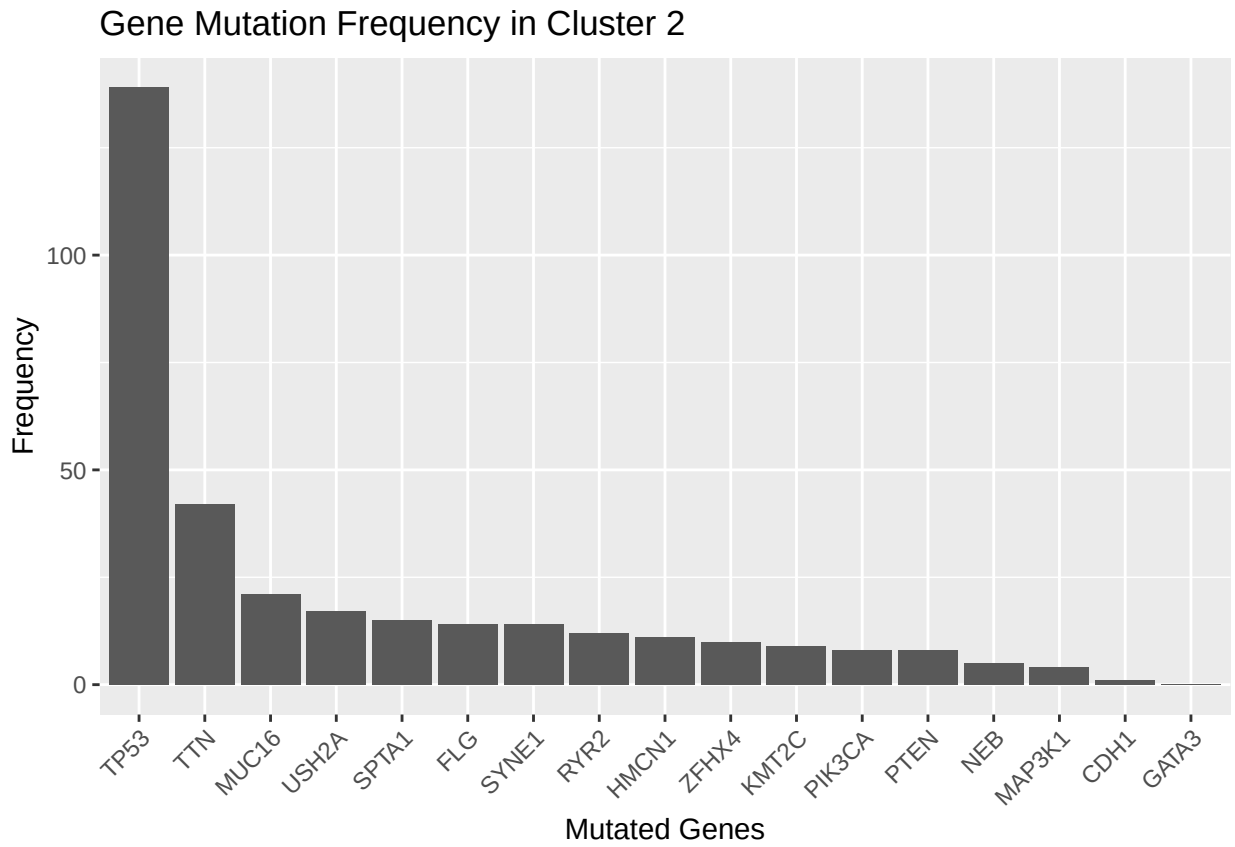
```r
# Remove the patients column
feature_df_filtered <- feature_df_filtered[, -1]
```

```r
# ------------- VISUALIZE MUTATION FREQS IN CLUSTERS -------------
# This function takes the cluster number as an input and returns a frequency
# table data frame of mutated genes in the cluster
freqData <- function(clust) {
  cluster <- feature_df_filtered[which(cut_ward_expression == clust),]

  # Create frequency table
  freq <- colSums(cluster == 1)

  # Convert frequency table to a data frame
  data <- data.frame(column = names(freq), frequency = freq)

  # Sort the data frame by frequency in descending order
  data <- data[order(data$frequency, decreasing = TRUE), ]
}
```

```r
# Get mutation frequency data for each cluster
clust.data1 <- freqData(1)
clust.data2 <- freqData(2)

# Create bar plots to visualize the most frequently mutated gene in each cluster
ggplot(clust.data1, aes(x = column, y = frequency)) +
  geom_col() + theme(axis.text.x = element_text(angle = 45,hjust=1)) +
  scale_x_discrete(limits = clust.data1$column) +
  labs(x = "Mutated Genes", y = "Frequency",
       title = "Gene Mutation Frequency in Cluster 1")
```




```
ggplot(clust.data2, aes(x = column, y = frequency)) +
  geom_col() + theme(axis.text.x = element_text(angle = 45,hjust=1)) +
  scale_x_discrete(limits = clust.data2$column) +
  labs(x = "Mutated Genes", y = "Frequency",
       title = "Gene Mutation Frequency in Cluster 2")
```



Cluster 1 is characterized by a high mutation frequency of PIK3CA, while cluster 2 is characterized by a high mutation frequency of TP53. This clusters are very similar to clusters 2 and 3 that was found by clustering the mutation data.

Survival Analysis on Expression Clusters

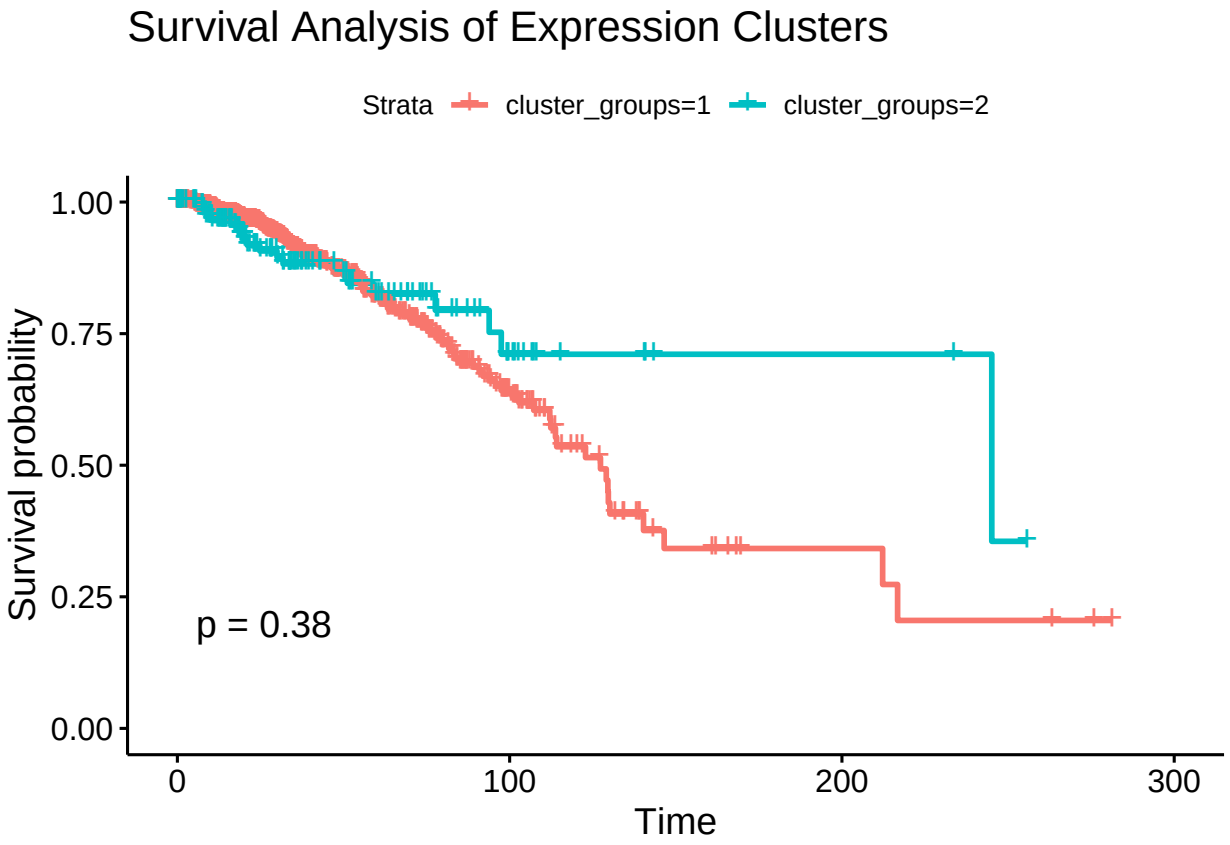
```
# ------------- CLEAN AND FORMAT SURVIVAL DATA -------------
# OS_MONTHS indicates the number of months from time of diagnosis to time of
# death or last follow up

# Create a subset of the clinicalData containing only the patients with mutated
# genes after filtering
clinical_cleaned <- data.clinical[which(data.clinical$PATIENT_ID
                                        %in% unique.patients.full.data) ,]

# Create a data frame called survival with a vector that contains TRUE =
# dead and FALSE = alive
survival_DF <- data.frame(deceased = clinical_cleaned$OS_STATUS == "1:DECEASED")

# Create a column of months to death from diagnosis
indices <- which(clinical_cleaned$OS_STATUS == "1:DECEASED")

# Set all to NA first since patients who are not dead should not have an
# OS_months value
```

```r
survival_DF$months_to_death = rep(NA, length(unique.patients.full.data))

for(i in indices) {
  survival_DF$months_to_death[i] = clinical_cleaned$OS_MONTHS[i]
}
```

```r
# ------------- SURVIVAL ANALYSIS ON EXPRESSION CLUSTERS -------------
survival_DF$progression_free = clinical_cleaned$PFS_MONTHS

# create an "overall survival" variable that is equal to days_to_death
# for dead patients, and to progression free disease for patients who
# are still alive
survival_DF$overall_survival = ifelse(survival_DF$deceased,
                                      survival_DF$months_to_death,
                                      survival_DF$progression_free )

# Create a vector within survival dataframe containing cluster groups for
# labeling
survival_DF$cluster_groups <- cut_ward_expression

# Now that the survival time has been tagged with the censoring, we can add the
# categorical independent variable `cluster groups`, and effectively create a
# formula

Surv(survival_DF$overall_survival,
     survival_DF$deceased) ~ survival_DF$cluster_groups
```

```
## Surv(survival_DF$overall_survival, survival_DF$deceased) ~ survival_DF$cluster_groups
```

```r
fit = survfit(Surv(overall_survival, deceased) ~ cluster_groups,
              data=survival_DF)

ggsurvplot(fit, data=survival_DF, pval=T,
           title = "Survival Analysis of Expression Clusters")
```



No significant results were found using surivial analysis on the two clusters generated from the expression data.