Genomic Data Science Capstone Analysis Description Report

Contents

1. Getting the Data (Reads)

The data was obtained from SRA in the form of a set of FASTQ files.

The FASTQ files were downloaded using sratoolkit.2.9.6-1w1.

The following bash script was employed:

```
#!/bin/bash

# FETAL SAMPLES

fastq-dump -v --gzip --split-files -O /data/sra  SRR2071348
printf "\n****** SRR2071348 is downloaded  ...\n\n"

fastq-dump -v --gzip --split-files -O /data/sra  SRR2071349
printf "\n****** SRR2071349 is downloaded  ...\n\n"

fastq-dump -v --gzip --split-files -O /data/sra  SRR2071352
printf "\n****** SRR2071352 is downloaded  ...\n\n"

# ADULT SAMPLES

fastq-dump -v --gzip --split-files -O /data/sra  SRR2071346
printf "\n****** SRR2071346 is downloaded  ...\n\n"

fastq-dump -v --gzip --split-files -O /data/sra  SRR2071347
printf "\n****** SRR2071347 is downloaded  ...\n\n"
```

```
fastq-dump -v --gzip --split-files -O /data/sra  SRR2071350
printf "\n****** SRR2071350 is downloaded  ...\n\n"
```

After saving the above script as `download-sra-reads.sh` in `bash-scripts` directory, the following bash command was used for running it:

```
nohup sh bash-scripts/download-sra-reads.sh > download-sra-reads.out &
```

1.1. Quality Control

The quality of the reads was checked using FastQC v0.11.8.

The following bash script was employed:

```
#!/bin/bash

# ALL SAMPLES

fastqc -o fastqc  /data/sra/*fastq.gz
```

2. Reads Alignment

The FASTQ files were aligned (mapped) to the reference genome hg38 using hisat2-2.1.0.

The following bash script was employed:

```
#!/bin/bash

# FETAL SAMPLES

echo "Aligning the reads of run SRR2071348"
hisat2 -p 8  --dta -x /data/hg38/genome -1 /data/sra/SRR2071348_1.fastq.gz -2 /data/sra/SRR2071348_2.fas
        | samtools sort -@ 8 -n -o /data/align/SRR2071348.bam
echo "Finished the aignment of reads of run SRR2071348"

echo "Aligning the reads of run SRR2071349"
hisat2 -p 8  --dta -x /data/hg38/genome -1 /data/sra/SRR2071349_1.fastq.gz -2 /data/sra/SRR2071349_2.fas
        | samtools sort -@ 8 -n -o /data/align/SRR2071349.bam
echo "Finished the aignment of reads of run SRR2071349"

echo "Aligning the reads of run SRR2071352"
hisat2 -p 8  --dta -x /data/hg38/genome -1 /data/sra/SRR2071352_1.fastq.gz -2 /data/sra/SRR2071352_2.fas
        | samtools sort -@ 8 -n -o /data/align/SRR2071352.bam
echo "Finished the aignment of reads of run SRR2071352"

# ADULT SAMPLES

echo "Aligning the reads of run SRR2071346"
hisat2 -p 8  --dta -x /data/hg38/genome -1 /data/sra/SRR2071346_1.fastq.gz -2 /data/sra/SRR2071346_2.fas
        | samtools sort -@ 8 -n -o /data/align/SRR2071346.bam
```

```
echo "Finished the aignment of reads of run SRR2071346"

echo "Aligning the reads of r SRR2071347"
hisat2 -p 8  --dta -x /data/hg38/genome -1 /data/sra/SRR2071347_1.fastq.gz -2 /data/sra/SRR2071347_2.fas
        | samtools sort -@ 8 -n -o /data/align/SRR2071347.bam
echo "Finished the aignment of reads of run SRR2071347"

echo "Aligning the reads of run SRR2071350"
hisat2 -p 8  --dta -x /data/hg38/genome -1 /data/sra/SRR2071350_1.fastq.gz -2 /data/sra/SRR2071350_2.fas
        | samtools sort -@ 8 -n -o /data/align/SRR2071350.bam
echo "Finished the aignment of reads of run SRR2071350"
```

After saving the above script as `hisat2.sh` in `bash-scripts` directory, the following bash command was used for running it:

```
nohup sh bash-scripts/hisat2.sh > hisat2.out &
```

3. Counting Reads in Genes

For quantifying the abundance of genes in each sample the number of reads corresponding to each gene in the genome in that sample is counted using HTSeq.0.11.1.

gencode.v28.annotation.gtf provides hg38 genome annotations used in the counting. Gene annotations describe the structure of transcripts (model) expressed from those gene loci. A transcript model consists of the coordinates of the exons of a transcript on a reference genome.

The following bash script was employed:

```
#!/bin/bash

ANNOT=/data/gencode.v28.annotation.gtf

# FETAL SAMPLES

printf "\nn***** Counting reads of run SRR2071348 ...\n\n"
htseq-count -f bam -s no -r name -m union --nonunique all /data/align/SRR2071348.bam $ANNOT > htseq/SRR

printf "\nn***** Counting reads of run SRR2071349 ...\n\n"
htseq-count -f bam -s no -r name -m union --nonunique all /data/align/SRR2071349.bam $ANNOT > htseq/SRR

printf "\nn***** Counting reads of run SRR2071352 ...\n\n"
htseq-count -f bam -s no -r name -m union --nonunique all /data/align/SRR2071352.bam $ANNOT > htseq/SRR

# ADULT SAMPLES

printf "\nn***** Counting reads of run SRR2071346 ...\n\n"
htseq-count -f bam -s no -r name -m union --nonunique all /data/align/SRR2071346.bam $ANNOT > htseq/SRR

printf "\nn***** Counting reads of run SRR2071347  ...\n\n"
htseq-count -f bam -s no -r name -m union --nonunique all /data/align/SRR2071347.bam $ANNOT > htseq/SRR

printf "\nn***** Counting reads of run SRR2071350 ...\n\n"
htseq-count -f bam -s no -r name -m union --nonunique all /data/align/SRR2071350.bam $ANNOT > htseq/SRR
```

After saving the above script as `htseq.sh` in `bash-scripts` directory, the following bash command was used for running it:

```
nohup sh bash-scripts/htseq.sh > htseq.out &
```

3.1. Merging the Count Files

The following R script was employed to merge the counts files into one file (`merged_counts.txt`):

```
# Load fetal samples
SRX683795 <- read.table("SRR2071348_counts.txt", header = FALSE)
SRX683796 <- read.table("SRR2071349_counts.txt", header = FALSE)
SRX683799 <- read.table("SRR2071352_counts.txt", header = FALSE)

# Load adult samples
SRX683793 <- read.table("SRR2071346_counts.txt", header = FALSE)
SRX683794 <- read.table("SRR2071347_counts.txt", header = FALSE)
SRX683797 <- read.table("SRR2071350_counts.txt", header = FALSE)

# check to see if all elements of the first column (transcripts) are the same across all 6 samples
all(SRX683797[,1] == SRX683794[,1])
all(SRX683793[,1] == SRX683794[,1])
all(SRX683799[,1] == SRX683793[,1])
all(SRX683797[,1] == SRX683794[,1])
all(SRX683795[,1] == SRX683797[,1])

# create a merged_counts table
merged_counts <- data.frame(row.names = SRX683795[,1], SRX683795 = SRX683795[,2], SRX683796 = SRX683796
                            , SRX683799 = SRX683799[,2], SRX683793 = SRX683793[,2]
                            , SRX683794 = SRX683794[,2], SRX683797 = SRX683797[,2])
write.table(merged_counts, "merged_counts.tsv", quote = FALSE, sep = '\t')
```

4. Exploratory Analysis

In this section, we explore the data (read counts) to get an idea of what the distribution of the data will look like.

Load the necessary libraries and data.

```
library(SummarizedExperiment)
library(AnnotationDbi)
library(org.Hs.eg.db)
library(DESeq2)
library(dplyr)
library(ggplot2)
library(gplots)
library(ggthemes)
library(EnhancedVolcano)
par(pch = 19)
tropical = c("darkorange", "dodgerblue", "hotpink", "limegreen", "yellow")
palette(tropical)
```

```r
# read a merged_counts file
merged_counts = read.table("merged_counts.tsv", quote = "", sep = '\t')

# Read phenotype sample data
pheno_data = read.csv("phenotype_data.tsv", quote = "", sep = '\t')
```

Normalization of expression (read counts) according to the Read Per Million (RPM) unit.

```r
x = as.matrix(merged_counts)
counts_RPM = t(t(x) * 1e6 / colSums(x))
```

Checking the distribution of the data.

```r
table(pheno_data$sex)
```

```
##
## F M
## 4 2
```

```r
table(pheno_data$age_group, useNA = "ifany")
```

```
##
## adult fetal
##     3     3
```

```r
table(pheno_data$sex, pheno_data$age_group)
```

```
##
##     adult fetal
##   F     2     2
##   M     1     1
```

```r
# Make the distribution of NA's by genes
gene_na = rowSums(is.na(counts_RPM))
gene_na[5]
```

```
## ENSG00000000460.16
##                  0
```

```r
gene_na[1:5]
```

```
## ENSG00000000003.14  ENSG00000000005.5 ENSG00000000419.12 ENSG00000000457.13
##                  0                  0                  0                  0
## ENSG00000000460.16
##                  0
```

```
# Make the distribution of NA's by samples
sample_na = colSums(is.na(counts_RPM))
sample_na[6]
```

```
## SRX683797
##         0
```

```
table(sample_na)
```

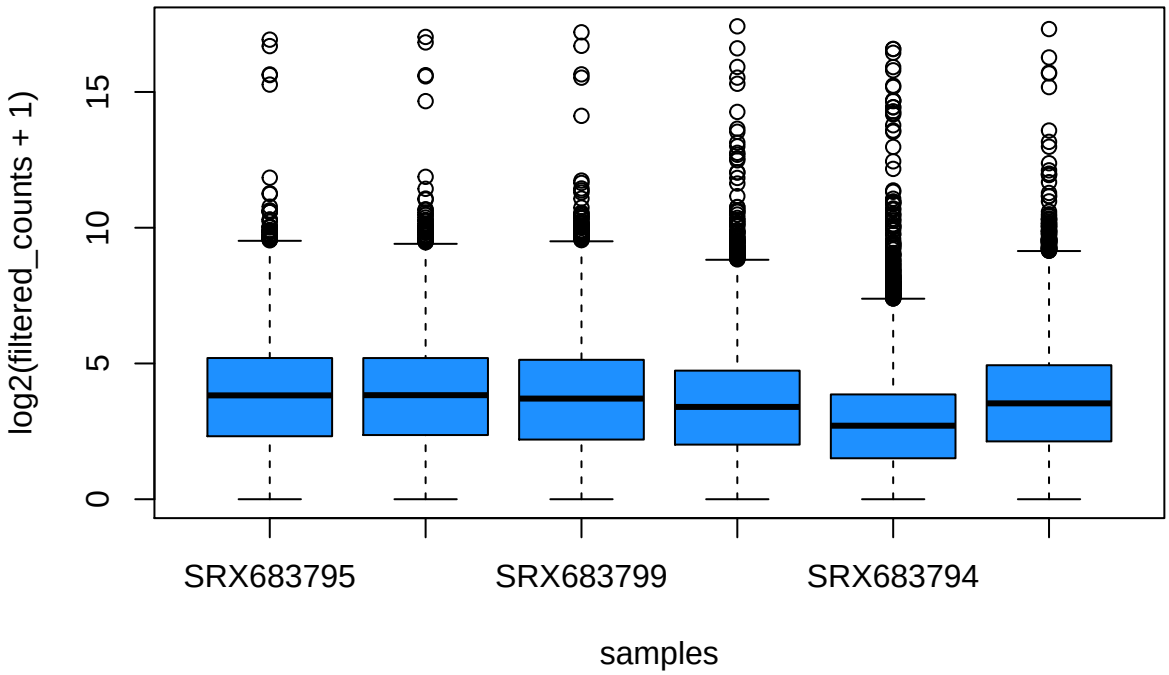
```
## sample_na
## 0
## 6
```

Boxplot all samples with log2 transformation of the filtered data.

```
counts_RPM = as.data.frame(counts_RPM)
fil_counts_RPM = filter(counts_RPM, rowMeans(counts_RPM) > 1)
dim(fil_counts_RPM)
```

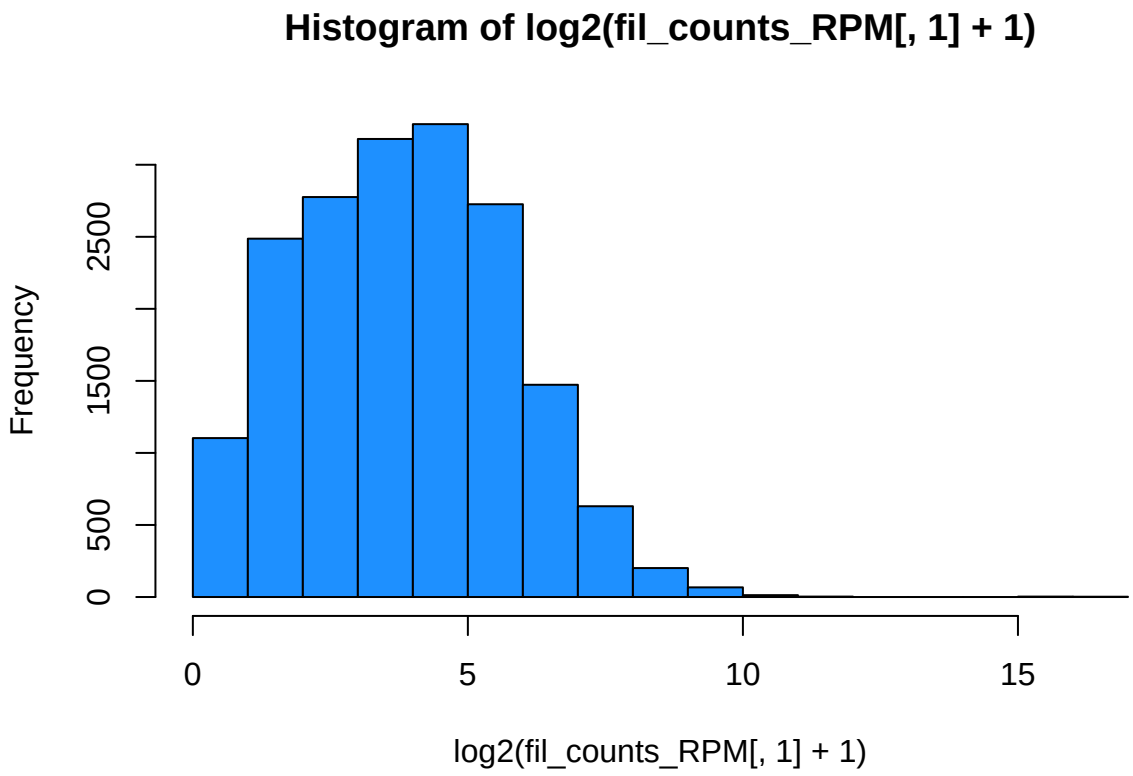
```
## [1] 17945     6
```

```
boxplot(as.matrix(log2(fil_counts_RPM+1)), col=2, xlab = "samples",  ylab = "log2(filtered_counts + 1)")
```



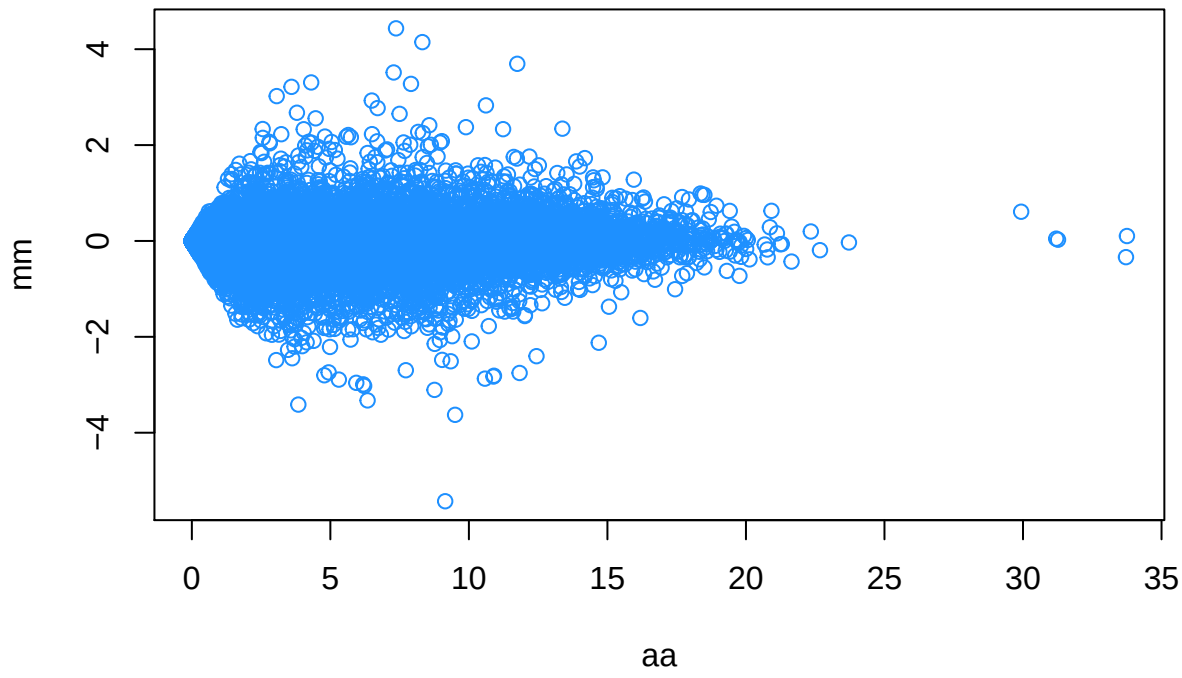
Plot a histogram to show the probability/frequency distribution of the filtered counts data of the first sample.

```
hist(log2(fil_counts_RPM[,1]+1), col = 2, Xlab = "log2(filtered_counts[,1] + 1)")
```



MA-plot between 2 samples to visualize the differences between measurements taken in the 2 samples.

```
aa = log2(counts_RPM[,1]+1) + log2(counts_RPM[,2]+1)
mm = log2(counts_RPM[,1]+1) - log2(counts_RPM[,2]+1)
plot(aa, mm, col=2)
```

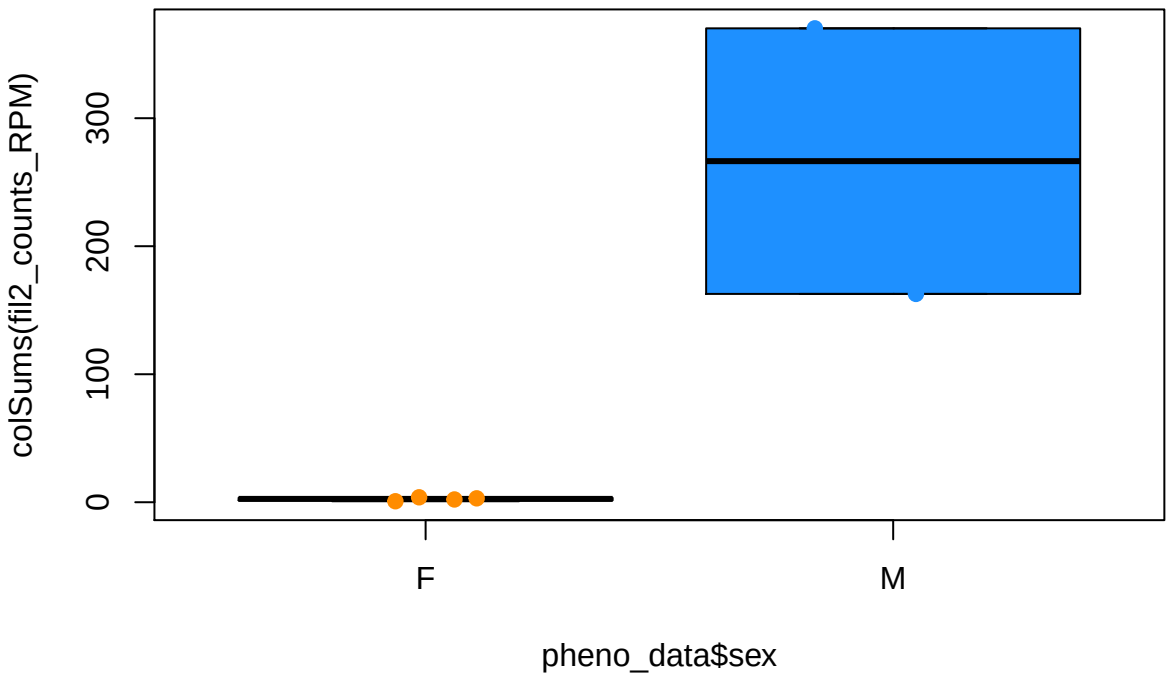


Confirm that male samples have more genes on chromosome Y than female samples.

```
par(pch = 19)
rownames(counts_RPM) = sub("\\.\\d+$", "", rownames(counts_RPM))

chr = AnnotationDbi::select(org.Hs.eg.db, keys = rownames(counts_RPM), keytype = "ENSEMBL", columns = "
chr = chr[!duplicated(chr[,1]),]

# Confirm that the annotations still have the same sort as the counts
all(chr[,1] == rownames(counts_RPM))
```

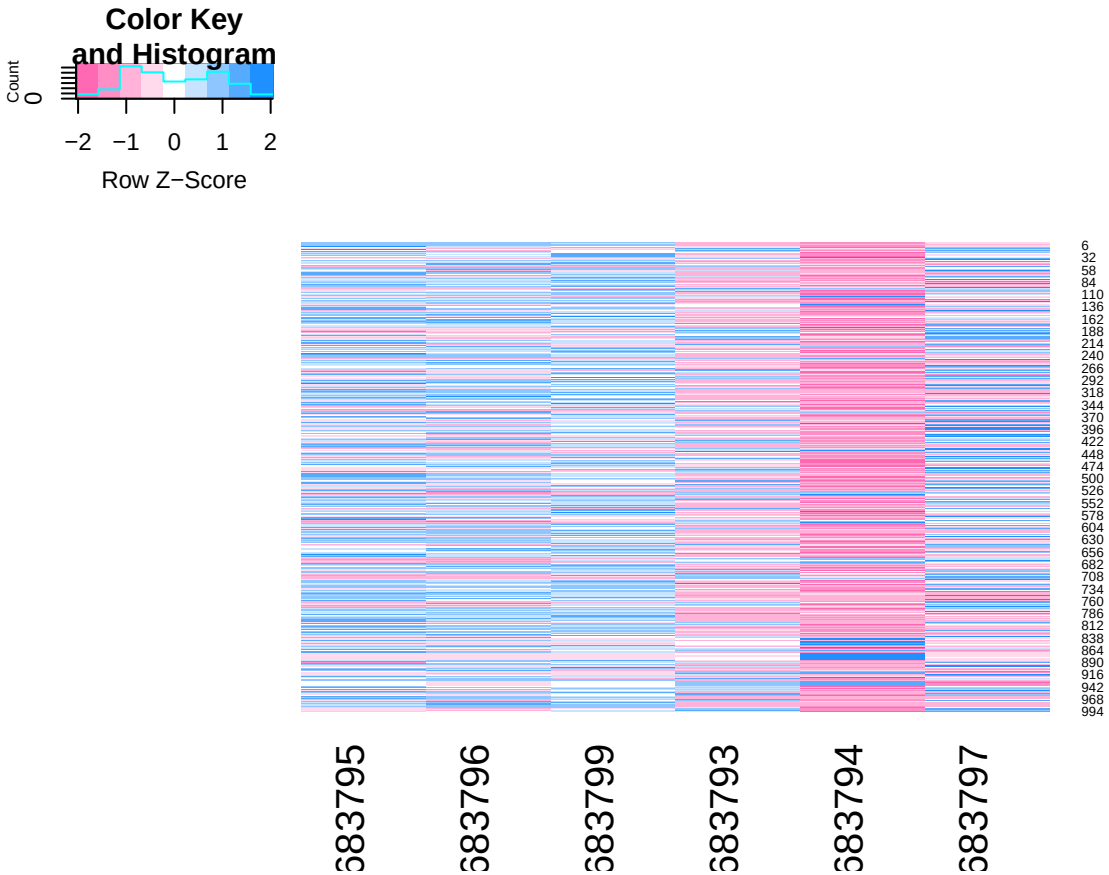
```
## [1] TRUE
```

```
# Select the chromosome Y samples
fil2_counts_RPM = filter(counts_RPM, chr$CHR == "Y")

# Male samples have more genes on chromosome Y than females
boxplot(colSums(fil2_counts_RPM) ~ pheno_data$sex, col=2)
points(colSums(fil2_counts_RPM) ~ jitter(as.numeric(pheno_data$sex)), col = as.numeric(pheno_data$sex))
```

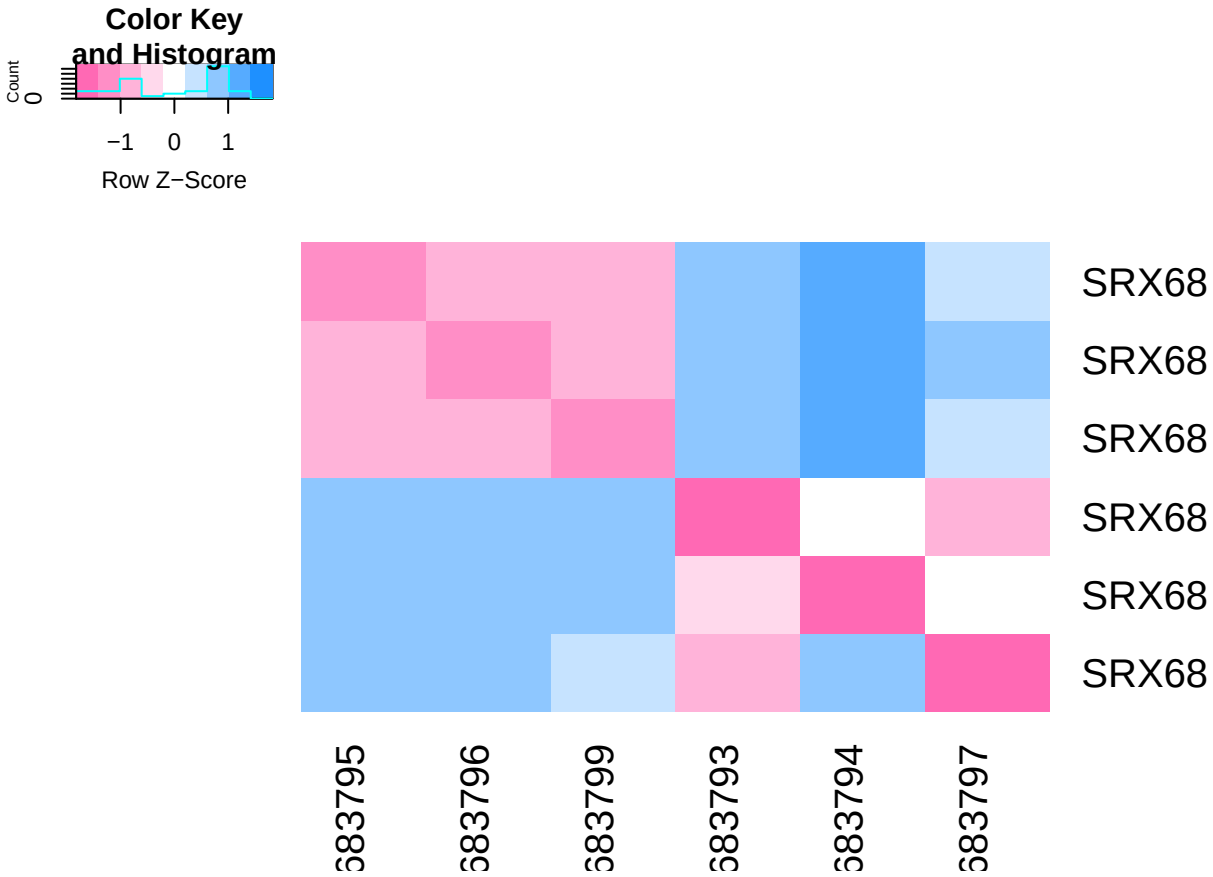


Plot Heatmaps; this gives us anoverview over similarities and dissimilarities between samples.

```
fil3_counts_RPM = filter(counts_RPM, rowMeans(counts_RPM) > 100)
coloramp =  colorRampPalette(c(3, "white", 2))(9)
# heatmap of the expression data itself
heatmap.2(as.matrix(fil3_counts_RPM), col = coloramp, Rowv = NA, Colv = NA,
          dendogram="none", scale="row", trace="none", title = "Heatmap of the Expression Data without
```




```
#heatmap of the distances between samples
fil3_counts_RPM = log2(fil3_counts_RPM + 1)
dist.samples = dist(t(fil3_counts_RPM))
heatmap.2(as.matrix(dist.samples), col = coloramp, Rowv = NA, Colv = NA,
          dendogram="none", scale="row", trace="none", title = "Heatmap Distances Between Samples of th
```



5. Differential Expression Analysis

Differential expression analysis between the sample age groups (adult vs. fetal) while adjusting for RIN was carried out using DESeq2_1.24.0.

Load the necessary libraries and the data.

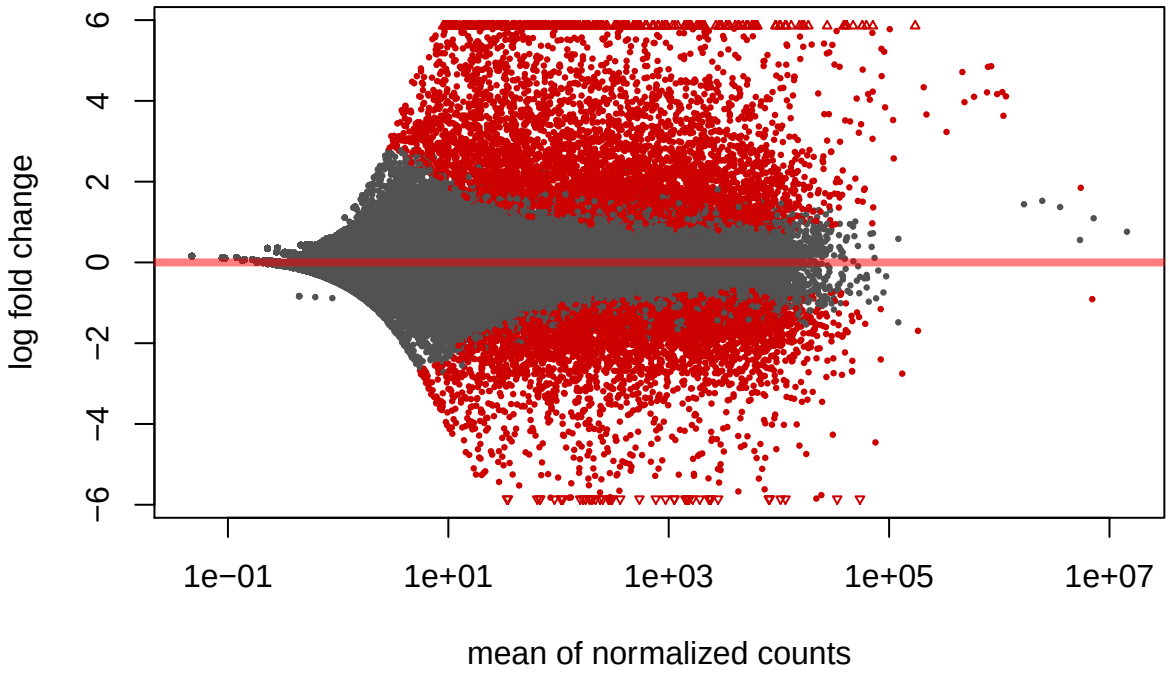
```
library(SummarizedExperiment)
library(AnnotationDbi)
library(org.Hs.eg.db)
library(DESeq2)
library(dplyr)
library(ggplot2)
library(ggthemes)
library(EnhancedVolcano)

par(pch = 19)

# read a merged_counts file
merged_counts = read.table("merged_counts.tsv", quote = "", sep = '\t')

# Read phenotype sample data
pheno_data = read.csv("phenotype_data.tsv", quote = "", sep = '\t')
```

Create DESeq2 object and get the results.

```
deseq.dat = DESeqDataSetFromMatrix(countData = merged_counts, colData = pheno_data, design = ~ age_grou
dds = DESeq(deseq.dat)
# DESeq2 results
res_deseq2 = results(dds, contrast = c("age_group", "adult", "fetal"))
res_deseq2_shrunk = lfcShrink(dds=dds, contrast = c("age_group", "adult", "fetal"), res = res_deseq2, ty
```

MA-plot between the sample age groups (adult vs. fetal) while adjusting for RIN: plot log2 fold-changes (on the y-axis) versus the mean of normalized counts (on the x-axis) for the differentially expressed genes.

```
pval_threshold = 10e-3
plotMA(res_deseq2_shrunk, alpha = pval_threshold)
```



Add gene symbols to the DESeq2 results table.

```
gene_symbol = read.table("gencode.v28.symbols.txt", header = TRUE, na.strings = "n/a", col.names = c("g
res_deseq2_shrunk = as.data.frame(res_deseq2_shrunk)
res_deseq2_shrunk$row = rownames(res_deseq2_shrunk)
res_deseq2_shrunk_annotated = merge(res_deseq2_shrunk,gene_symbol, by.x = "row", by.y = "gene", all = T
```

Identify genes with FDR < 0.05 and sort them by increasing adjusted p-value.

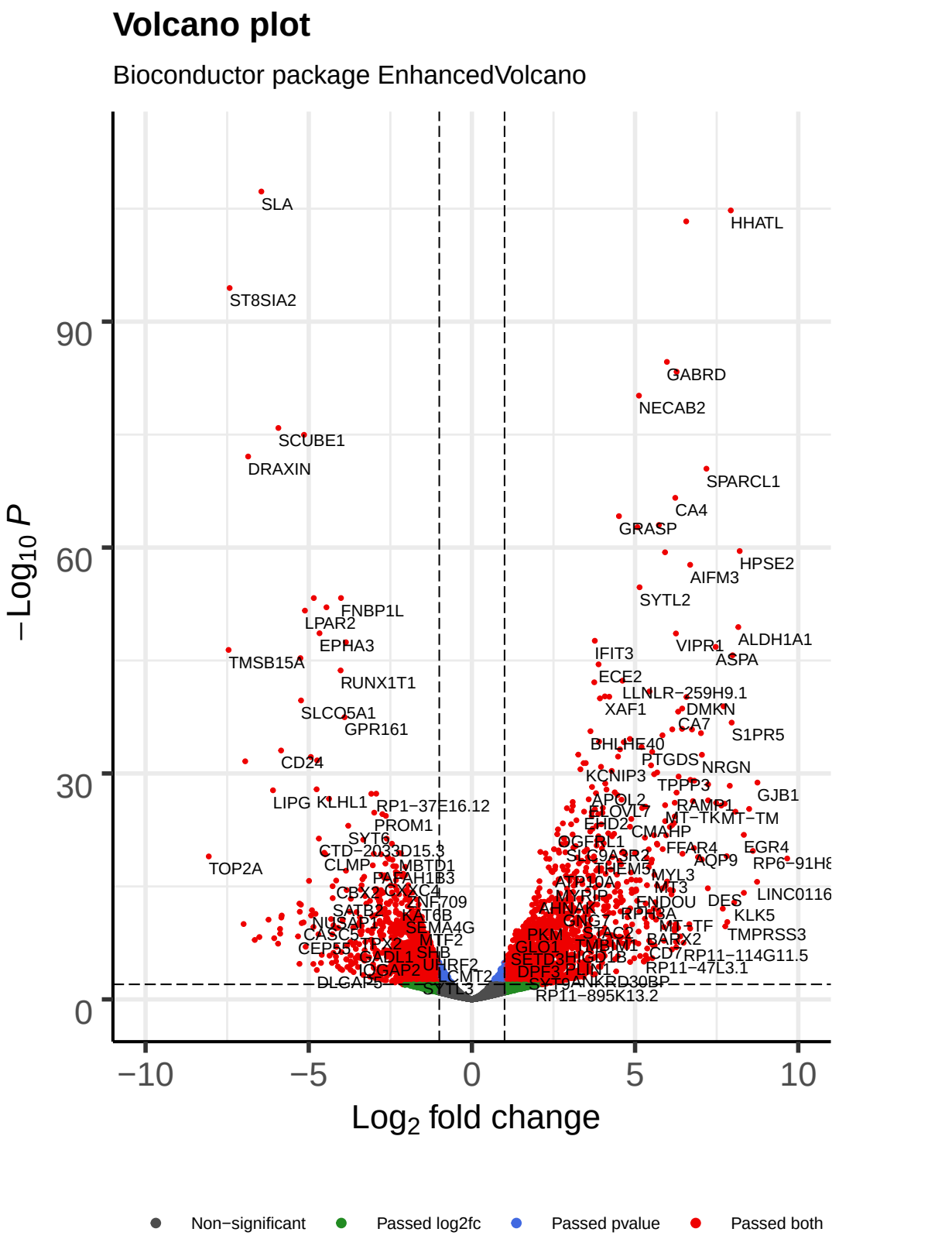
```
significant_differ_res_deseq2_shrunk = subset(res_deseq2_shrunk_annotated, res_deseq2_shrunk$padj < 0.05
dim(significant_differ_res_deseq2_shrunk)
```

```
## [1] 12941     7
```

```
sorted_significant_differ_res_deseq2_shrunk = significant_differ_res_deseq2_shrunk[sort.list(significan
```

Volcano plot: visualize significance and the magnitude of changes in genes.

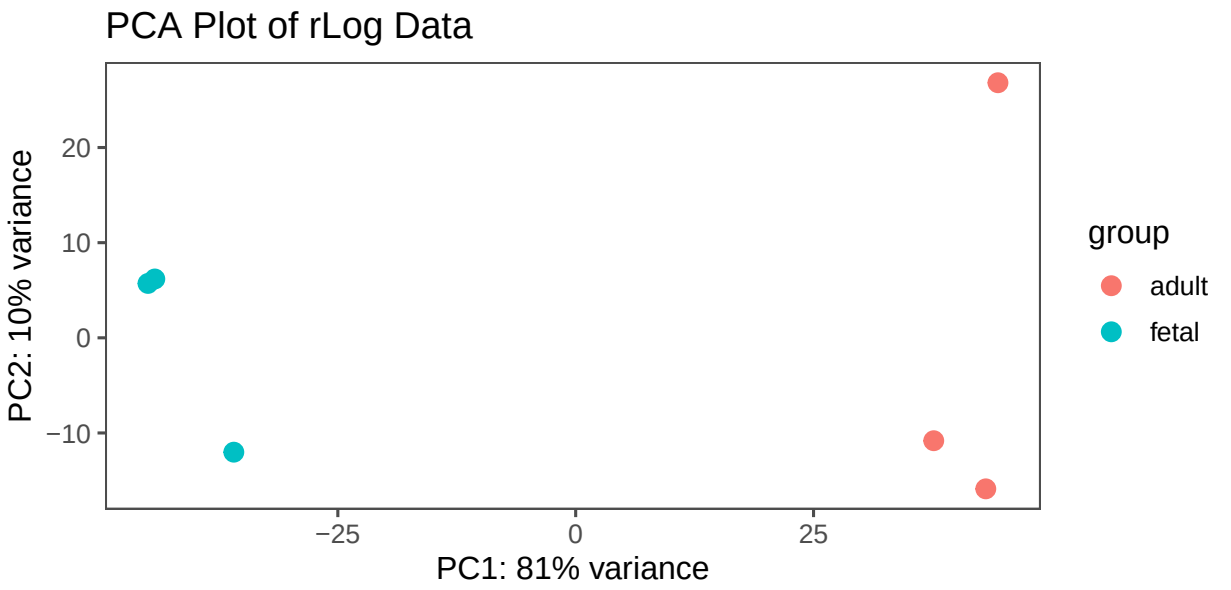
```
EnhancedVolcano(sorted_significant_differ_res_deseq2_shrunk, FCcutoff = 1, pCutoff = pval_threshold,
                sorted_significant_differ_res_deseq2_shrunk$symbol,
                x = 'log2FoldChange', y = 'pvalue',  xlim = c(-10, 10),
                legend = c("Non-significant", "Passed log2fc", "Passed pvalue", "Passed both"),
                legendPosition = 'bottom',
                legendLabSize = 9,
                legendIconSize = 2,
                widthConnectors = 0.2,
                colConnectors = 'grey30',
                colAlpha = 1)
```


Volcano plot
Bioconductor package EnhancedVolcano
−Log10 P
Log2 fold change
Non−significant
Passed log2fc
Passed pvalue
Passed both
Total = 12941 variables

Principal

Components Analysis (PCA) plot. A PCA plot shows clusters of samples based on their similarity. It reduces the overwhelming number of dimensions by constructing principal components (PCs).

```
rld = rlogTransformation(dds)
plotPCA(rld, intgroup = "age_group") + ggtitle("PCA Plot of rLog Data") + theme_few()
```



6. Epigenetics and Expression Analysis

The following R script gets the percentage of overlap between the promoters of differentially expressed genes found in step 5. (Differential Expression Analysis) and the epigenetically marked (H3K4me3) promoters in the fetal and adult brain, and liver cell line.

```
library(AnnotationHub)
library(AnnotationDbi)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
library(EnsDb.Hsapiens.v86)
library(mygene)

# load list of of differentially expressed gene
diff.expressed.genes = read.table("results.tsv", quote = "", sep = '\t')

# check to see if FXN was in these differentially expressed genes
diff.expressed.genes[which(diff.expressed.genes$name == "FXN"), ]
```

```
## [1] gene   name   log2fc pval   padj
## <0 rows> (or 0-length row.names)
```

```
# AnnotationHub
ah = AnnotationHub()
ah = subset(ah, species == "Homo sapiens")

# get the narrow peaks data (promoter associated histone modification H3K4me3) for the fetal brain cell
fetal.brain = AnnotationHub::query(ah, c("EpigenomeRoadMap", "H3K4me3", "E081"))
fetal.brain.gr = fetal.brain[[2]]

# get the narrow peaks data (promoter associated histone modification H3K4me3) for the adult brain cell
adult.brain = AnnotationHub::query(ah, c("EpigenomeRoadMap", "H3K4me3", "E073"))
adult.brain.gr = adult.brain[[2]]

# get the narrow peak data (promoter associated histone modification H3K4me3) for liver cell line is ca
liver.line = AnnotationHub::query(ah, c("EpigenomeRoadMap", "H3K4me3", "Liver"))
liver.line.gr = liver.line[[2]]

# get the known genes from Tx database
txdb = TxDb.Hsapiens.UCSC.hg19.knownGene
txdb.genes = genes(txdb)

# change "ENSEMBL" ID to "ENTREZID" that matches gene_id
diff.expressed.genes$gene = sub("\\.\\d+$", "", as.character(diff.expressed.genes$gene))
AnnotationDbi::keytypes(EnsDb.Hsapiens.v86)
```

```
##  [1] "ENTREZID"            "EXONID"              "GENEBIOTYPE"
##  [4] "GENEID"              "GENENAME"            "PROTDOMID"
##  [7] "PROTEINDOMAINID"     "PROTEINDOMAINSOURCE" "PROTEINID"
## [10] "SEQNAME"             "SEQSTRAND"           "SYMBOL"
## [13] "TXBIOTYPE"           "TXID"                "TXNAME"
## [16] "UNIPROTID"
```

```
# diff.expressed.genes = diff.expressed.genes[!is.na(diff.expressed.genes$name), ]
# diff.expressed.genes.map = queryMany(diff.expressed.genes$name, scopes="symbol", fields="entrezgene",
diff.expressed.genes.map = AnnotationDbi::select(EnsDb.Hsapiens.v86, keys = as.character(diff.expressed

# get the promoters of differentially expressed genes
diff.expressed.gene.promoters = promoters(txdb.genes[txdb.genes$gene_id %in% diff.expressed.genes.map$E

# check to see if FXN (Entrez ID 2395) was in these promoters
diff.expressed.gene.promoters[which(diff.expressed.gene.promoters$gene_id == "2395")]
```

```
## GRanges object with 0 ranges and 1 metadata column:
##    seqnames    ranges strand |     gene_id
##       <Rle> <IRanges>  <Rle> | <character>
##   -------
##   seqinfo: 93 sequences (1 circular) from hg19 genome
```

The percentage of overlap between the promoters of differentially expressed genes found in step 5. (Differential Expression Analysis) and the epigenetically marked (H3K4me3) promoters in the **fetal brain cell line** is calculated as following:

```
# subsetByOverlaps() extracts the elements in the query (the first argument) that overlap at least one
fetal.brain.overlap.H3K4me3 = subsetByOverlaps(diff.expressed.gene.promoters, fetal.brain.gr)
(fetal.brain.overlap.percentage.H3K4me3 = length(fetal.brain.overlap.H3K4me3) / length(diff.expressed.g
```

```
## [1] 31.38979
```

The percentage of overlap between the promoters of differentially expressed genes found in step 5. (Differential Expression Analysis) and the epigenetically marked (H3K4me3) promoters in the **adult brain cell line** is calculated as following:

```
# subsetByOverlaps() extracts the elements in the query (the first argument) that overlap at least one
adult.brain.overlap.H3K4me3 = subsetByOverlaps(diff.expressed.gene.promoters, adult.brain.gr)
(adult.brain.overlap.percentage.H3K4me3 = length(adult.brain.overlap.H3K4me3) / length(diff.expressed.g
```

```
## [1] 45.14114
```

The percentage of overlap between the promoters of differentially expressed genes found in step 5. (Differential Expression Analysis) and the epigenetically marked (H3K4me3) promoters in the **liver cell line** is calculated as following:

```
# subsetByOverlaps() extracts the elements in the query (the first argument) that overlap at least one
liver.line.overlap.H3K4me3 = subsetByOverlaps(diff.expressed.gene.promoters, liver.line.gr)
(liver.line.overlap.percentage.H3K4me3 = length(liver.line.overlap.H3K4me3) / length(diff.expressed.gen
```

```
## [1] 42.32608
```