MutationAnalysis

Theo-60985751, Hannah-88788427 , Jingxuan-85741635

2023-11-23

# Data processing

### Including the libraries

library(ggplot2)

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

library(pheatmap)

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

library("TCGAbiolinks")  
library("survival")

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

library("survminer")

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

## Loading required package: ggpubr

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

##   
## Attaching package: 'survminer'

## The following object is masked from 'package:survival':  
##   
## myeloma

library("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

library(DESeq2)  
library("gridExtra")

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

##   
## Attaching package: 'gridExtra'

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

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

library("AnnotationDbi")  
library("org.Hs.eg.db")

##

library(pathview)

## ##############################################################################  
## Pathview is an open source software package distributed under GNU General  
## Public License version 3 (GPLv3). Details of GPLv3 is available at  
## http://www.gnu.org/licenses/gpl-3.0.html. Particullary, users are required to  
## formally cite the original Pathview paper (not just mention it) in publications  
## or products. For details, do citation("pathview") within R.  
##   
## The pathview downloads and uses KEGG data. Non-academic uses may require a KEGG  
## license agreement (details at http://www.kegg.jp/kegg/legal.html).  
## ##############################################################################

library(gage)

##

### Opening the datafiles

data\_mutation = read.delim("data\_mutations.txt", header = TRUE, sep = '\t')  
data\_clinical = read.table("data\_clinical\_patient.txt",header = TRUE, sep = '\t')  
data\_rna = read.csv("RNAseq\_LIHC.csv", header = TRUE, row.names = "X")

### Finding the patients that we have data for clinical, genome, and RNAseq data

unique\_patient\_clinical = unique(data\_clinical$PATIENT\_ID)  
unique\_patient\_rna = unique(colnames(data\_rna))  
unique\_patient\_mutation = unique(data\_mutation$Tumor\_Sample\_Barcode)  
  
shortened\_rna = substr(unique\_patient\_rna, start = 1, stop = 12)  
shortened\_rna = gsub("\\.", "-", shortened\_rna)  
  
shortened\_mutation = substr(unique\_patient\_mutation, start = 1, stop = 12)  
  
common\_names1 <- intersect(unique\_patient\_clinical, shortened\_mutation)  
common\_names2 <- intersect(common\_names1, shortened\_rna)

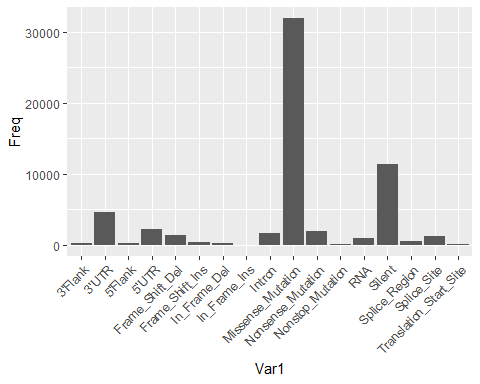
### Adding the patient ID at the last column and get the common datas

data\_clinical\_common = subset(data\_clinical, PATIENT\_ID %in% common\_names2)  
  
data\_mutation$PATIENT\_ID = substr(data\_mutation$Tumor\_Sample\_Barcode, start = 1, stop = 12)  
data\_mutation\_common = subset(data\_mutation, PATIENT\_ID %in% common\_names2)  
  
data\_mutation\_common$Tumor\_Sample\_Barcode = data\_mutation\_common$PATIENT\_ID  
  
data\_rna\_shortened = data\_rna  
colnames(data\_rna\_shortened) = substr(colnames(data\_rna\_shortened), start = 1, stop = 12)  
colnames(data\_rna\_shortened) = gsub("\\.", "-", colnames(data\_rna\_shortened))  
data\_rna\_common = data\_rna\_shortened[,common\_names2]

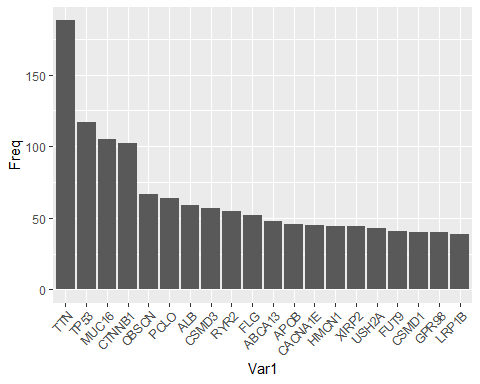
# Analysis of mutation data

Generate a plot for distribution of variant classifications

data\_oncoplot = data\_mutation\_common  
  
hugo <- as.data.frame(table(data\_oncoplot$Hugo\_Symbol))  
var.class <- as.data.frame(table(data\_oncoplot$Variant\_Classification))  
ggplot(data=var.class, aes(x=Var1, y=Freq))+  
 geom\_col()+  
 theme(axis.text.x = element\_text(angle = 45,hjust=1))

 Generate a plot for distribution of mutation events

hugo <- as.data.frame(table(data\_mutation\_common$Hugo\_Symbol))  
  
hugo.ordered <- hugo[order(-hugo$Freq),]  
  
ggplot(data=hugo.ordered[1:20,], aes(x=Var1, y=Freq))+  
 geom\_col()+  
 theme(axis.text.x = element\_text(angle = 45,hjust=1))+  
 scale\_x\_discrete(limits = hugo.ordered[1:20,]$Var1)

 Generate an oncoplot matrix of all genes

cnv\_events = unique(data\_oncoplot$Variant\_Classification)  
oncomat = reshape2::dcast(  
 data = data\_oncoplot,  
 formula = Hugo\_Symbol ~ Tumor\_Sample\_Barcode,  
 fun.aggregate = function(x, cnv = cnv\_events) {  
 x = as.character(x) # >= 2 same/distinct variant classification = Multi\_Hit  
 xad = x[x %in% cnv]  
 xvc = x[!x %in% cnv]  
   
 if (length(xvc) > 0) {  
 xvc = ifelse(test = length(xvc) > 1,  
 yes = 'Multi\_Hit',  
 no = xvc)  
 }  
   
 x = ifelse(  
 test = length(xad) > 0,  
 yes = paste(xad, xvc, sep = ';'),  
 no = xvc  
 )  
 x = gsub(pattern = ';$',  
 replacement = '',  
 x = x)  
 x = gsub(pattern = '^;',  
 replacement = '',  
 x = x)  
 return(x)  
 },  
 value.var = 'Variant\_Classification',  
 fill = '',  
 drop = FALSE  
)  
  
rownames(oncomat) = oncomat$Hugo\_Symbol  
oncomat <- oncomat[,-1]  
  
oncomat.ordered <- oncomat[order(-hugo$Freq),]

Transform the matrix into a binary matrix

mat <- oncomat.ordered  
mat[mat== "Silent"] = 0  
mat[mat == "Intron"] = 0  
mat[mat == "Missense\_Mutation"] = 0  
mat[mat == ""] = 0  
  
mat <- apply(mat, 2 ,as.numeric)  
mat <- as.matrix(mat)  
mat[is.na(mat)]=1  
  
rownames(mat) <- row.names(oncomat.ordered)

Finding the top 20 most mutated genes

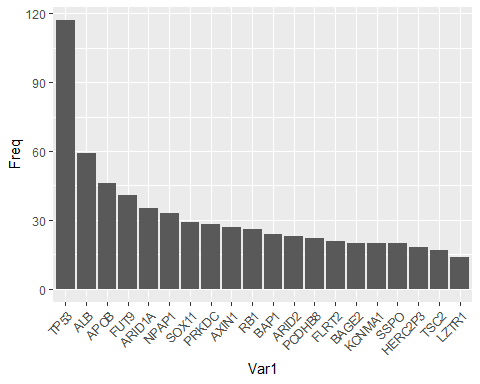
genes = rowSums(mat)  
genes.ordered = sort(genes, decreasing = TRUE)  
  
  
genes.ordered.top = genes.ordered[1:20]  
genes.ordered.top.names = names(genes.ordered.top)  
  
data\_oncoplot.top = subset(data\_oncoplot, Hugo\_Symbol %in% genes.ordered.top.names)

Making a matrix of the top 20 mutated genes

cnv\_events = unique(data\_oncoplot.top$Variant\_Classification)  
oncomat.top = reshape2::dcast(  
 data = data\_oncoplot.top,  
 formula = Hugo\_Symbol ~ Tumor\_Sample\_Barcode,  
 fun.aggregate = function(x, cnv = cnv\_events) {  
 x = as.character(x) # >= 2 same/distinct variant classification = Multi\_Hit  
 xad = x[x %in% cnv]  
 xvc = x[!x %in% cnv]  
   
 if (length(xvc) > 0) {  
 xvc = ifelse(test = length(xvc) > 1,  
 yes = 'Multi\_Hit',  
 no = xvc)  
 }  
   
 x = ifelse(  
 test = length(xad) > 0,  
 yes = paste(xad, xvc, sep = ';'),  
 no = xvc  
 )  
 x = gsub(pattern = ';$',  
 replacement = '',  
 x = x)  
 x = gsub(pattern = '^;',  
 replacement = '',  
 x = x)  
 return(x)  
 },  
 value.var = 'Variant\_Classification',  
 fill = '',  
 drop = FALSE  
)  
hugo <- as.data.frame(table(data\_oncoplot.top$Hugo\_Symbol))  
  
rownames(oncomat.top) = oncomat.top$Hugo\_Symbol  
oncomat.top <- oncomat.top[,-1]  
oncomat.top.ordered <- oncomat.top[order(-hugo$Freq),]

Generate a plot for distribution of the top 20 most mutated genes

hugo.ordered <- hugo[order(-hugo$Freq),]  
ggplot(hugo.ordered, aes(x=Var1, y=Freq))+  
 geom\_col()+  
 theme(axis.text.x = element\_text(angle = 45,hjust=1))+  
 scale\_x\_discrete(limits = hugo.ordered$Var1)

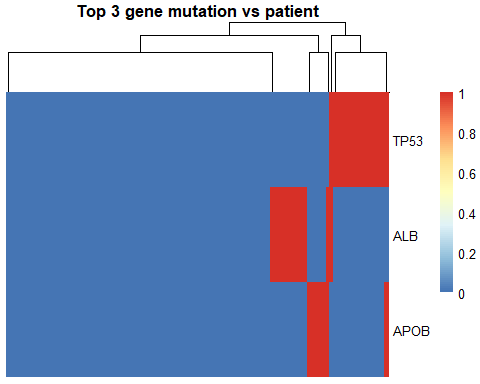


Transforming the top 20 gene matrix into binary

mat.top <- oncomat.top.ordered  
mat.top[mat.top== "Silent"] = 0  
mat.top[mat.top == "Intron"] = 0  
mat.top[mat.top == "Missense\_Mutation"] = 0  
  
mat.top[mat.top == ""] = 0  
  
  
mat.top <- apply(mat.top, 2 ,as.numeric)  
mat.top <- as.matrix(mat.top)  
mat.top[is.na(mat.top)]=1  
  
rownames(mat.top) <- row.names(oncomat.top.ordered)

Generate a pheatmap for top 3 mutated genes

reduce.mat <- mat.top[1:3,]  
res <- pheatmap(reduce.mat,  
 cluster\_rows = FALSE,  
 show\_colnames = FALSE, main = "Top 3 gene mutation vs patient")

 Performing k-mean clustering

cluster = as.data.frame(cutree(res$tree\_col, k = 2))  
# cluster

Finding the patientID in each group

mutation\_group1\_patientID = subset(cluster,`cutree(res$tree\_col, k = 2)` == 1)  
mutation\_group2\_patientID = subset(cluster,`cutree(res$tree\_col, k = 2)` == 2)  
  
data\_clinical\_common\_group1 = subset(data\_clinical\_common,PATIENT\_ID %in% rownames(mutation\_group1\_patientID))  
  
data\_clinical\_common\_group2 = subset(data\_clinical\_common,PATIENT\_ID %in% rownames(mutation\_group2\_patientID))  
  
mutation\_TP53\_patientID = data\_mutation\_common$PATIENT\_ID[data\_mutation\_common$Hugo\_Symbol == "TP53"]  
  
data\_clinical\_common\_TP53 = subset(data\_clinical\_common, PATIENT\_ID %in% mutation\_TP53\_patientID)

# Survival analysis

## survival analysis (SA) on clinical data

data\_clinical\_common$deceased = data\_clinical\_common$PFS\_STATUS == "1:PROGRESSION"  
  
# create an "overall survival" variable that is equal to days\_to\_death  
# for dead patients, and to days\_to\_last\_follow\_up for patients who  
# are still alive  
data\_clinical\_common$overall\_survival = ifelse(data\_clinical\_common$deceased,  
 data\_clinical\_common$OS\_MONTHS,  
 data\_clinical\_common$DAYS\_LAST\_FOLLOWUP)

### SA by cluster

# Adding a cluster annotation column  
  
data\_clinical\_common$cluster = cluster[data\_clinical\_common$PATIENT\_ID,]  
  
Surv(data\_clinical\_common$overall\_survival, data\_clinical\_common$deceased) ~ data\_clinical\_common$cluster

## Surv(data\_clinical\_common$overall\_survival, data\_clinical\_common$deceased) ~   
## data\_clinical\_common$cluster

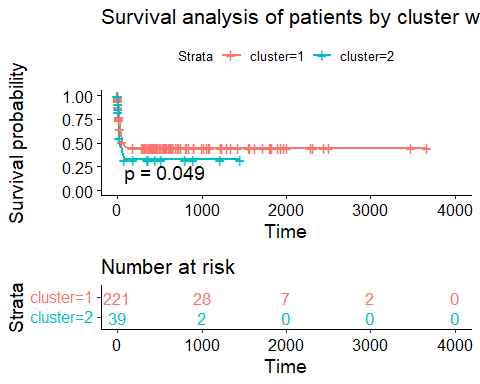
table(data\_clinical\_common$cluster)

##   
## 1 2   
## 240 45

fit = survfit(Surv(overall\_survival, deceased) ~ cluster, data = data\_clinical\_common)  
  
pval = surv\_pvalue(fit, data=data\_clinical\_common)$pval  
print(pval)

## [1] 0.04853337

ggsurvplot(fit, data=data\_clinical\_common, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35, title="Survival analysis of patients by cluster with mutation data")



### SA by sex

Surv(data\_clinical\_common$overall\_survival, data\_clinical\_common$deceased) ~ data\_clinical\_common$SEX

## Surv(data\_clinical\_common$overall\_survival, data\_clinical\_common$deceased) ~   
## data\_clinical\_common$SEX

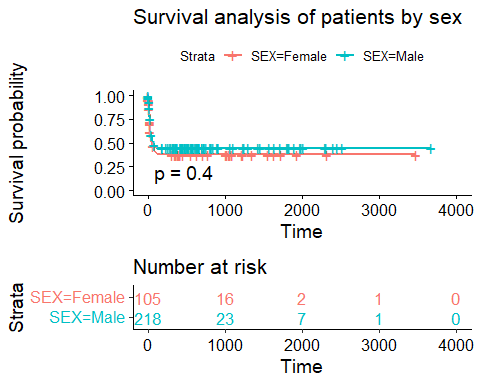
table(data\_clinical\_common$SEX)

##   
## Female Male   
## 117 236

fit = survfit(Surv(overall\_survival, deceased) ~ SEX, data = data\_clinical\_common)  
  
pval = surv\_pvalue(fit, data=data\_clinical\_common)$pval  
print(pval)

## [1] 0.4000524

ggsurvplot(fit, data=data\_clinical\_common, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35, title = "Survival analysis of patients by sex")

 ### SA by Tumor Stage

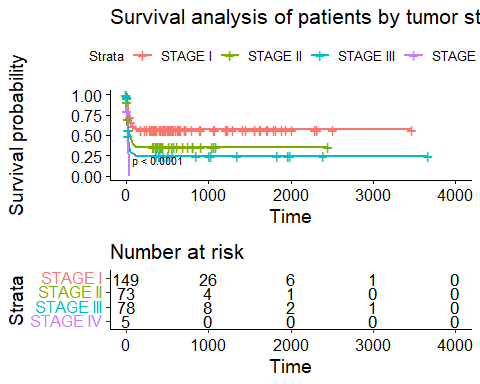
# remove any of the letters "a", "b" or "c", but only if they are at the end  
# of the name, eg "stage iiia" would become simply "stage iii"  
data\_clinical\_common$AJCC\_PATHOLOGIC\_TUMOR\_STAGE = gsub("[ABC]$", "", data\_clinical\_common$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)  
  
   
data\_clinical\_common[which(data\_clinical\_common$AJCC\_PATHOLOGIC\_TUMOR\_STAGE == ""), "AJCC\_PATHOLOGIC\_TUMOR\_STAGE"] = NA  
  
table(data\_clinical\_common$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)

##   
## STAGE I STAGE II STAGE III STAGE IV   
## 166 81 82 6

fit = survfit(Surv(overall\_survival, deceased) ~ AJCC\_PATHOLOGIC\_TUMOR\_STAGE, data=data\_clinical\_common)  
  
# we can extract the survival p-value and print it  
pval = surv\_pvalue(fit, data=data\_clinical\_common)$pval  
print(pval)

## [1] 9.488618e-09

ggsurvplot(fit, data=data\_clinical\_common, pval=T,pval.size = 3 ,risk.table=T, risk.table.height=0.39,legend.lab = c("STAGE I", "STAGE II", "STAGE III", "STAGE IV"), title = "Survival analysis of patients by tumor stage", title.fontsize = 2)

 ## Survival analysis on patients in each cluster (produced by mutation data) ### SA on cluster 1 by Tumor Stage

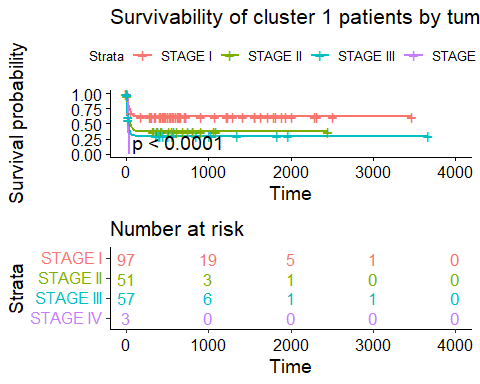
data\_clinical\_common\_group1$deceased = data\_clinical\_common\_group1$PFS\_STATUS == "1:PROGRESSION"  
  
# create an "overall survival" variable that is equal to days\_to\_death  
# for dead patients, and to days\_to\_last\_follow\_up for patients who  
# are still alive  
data\_clinical\_common\_group1$overall\_survival = ifelse(data\_clinical\_common\_group1$deceased,  
 data\_clinical\_common\_group1$OS\_MONTHS,  
 data\_clinical\_common\_group1$DAYS\_LAST\_FOLLOWUP)  
  
# remove any of the letters "a", "b" or "c", but only if they are at the end  
# of the name, eg "stage iiia" would become simply "stage iii"  
data\_clinical\_common\_group1$AJCC\_PATHOLOGIC\_TUMOR\_STAGE = gsub("[ABC]$", "", data\_clinical\_common\_group1$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)  
  
data\_clinical\_common\_group1[which(data\_clinical\_common\_group1$AJCC\_PATHOLOGIC\_TUMOR\_STAGE == ""), "AJCC\_PATHOLOGIC\_TUMOR\_STAGE"] = NA  
  
table(data\_clinical\_common\_group1$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)

##   
## STAGE I STAGE II STAGE III STAGE IV   
## 108 58 58 3

fit = survfit(Surv(overall\_survival, deceased) ~ AJCC\_PATHOLOGIC\_TUMOR\_STAGE , data=data\_clinical\_common\_group1)  
  
# we can extract the survival p-value and print it  
pval = surv\_pvalue(fit, data=data\_clinical\_common\_group1)$pval  
print(pval)

## [1] 7.978726e-06

ggsurvplot(fit, data=data\_clinical\_common\_group1, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.45,legend.lab = c("STAGE I", "STAGE II", "STAGE III", "STAGE IV"), title = "Survivability of cluster 1 patients by tumor stage")



### SA on cluster 1 by Sex

Surv(data\_clinical\_common\_group1$overall\_survival, data\_clinical\_common\_group1$deceased) ~ data\_clinical\_common\_group1$SEX

## Surv(data\_clinical\_common\_group1$overall\_survival, data\_clinical\_common\_group1$deceased) ~   
## data\_clinical\_common\_group1$SEX

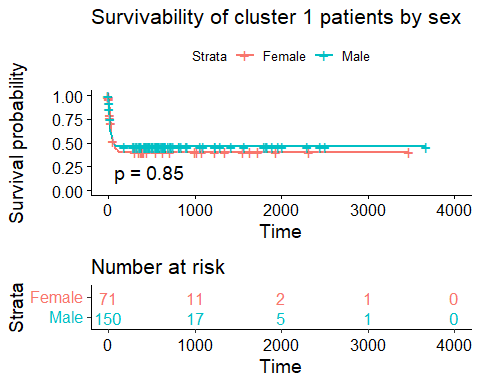
fit = survfit(Surv(overall\_survival, deceased) ~ SEX, data = data\_clinical\_common\_group1)  
  
table(data\_clinical\_common\_group1$SEX)

##   
## Female Male   
## 80 160

pval = surv\_pvalue(fit, data=data\_clinical\_common\_group1)$pval  
print(pval)

## [1] 0.8544197

ggsurvplot(fit, data=data\_clinical\_common\_group1, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35, legend.lab = c("Female", "Male"), title = "Survivability of cluster 1 patients by sex")



### SA on cluster 2 Tumor stage

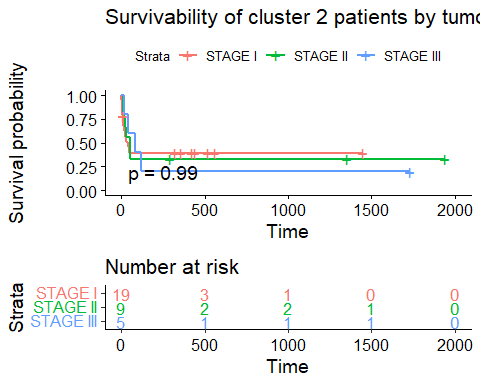
data\_clinical\_common\_group2$deceased = data\_clinical\_common\_group2$PFS\_STATUS == "1:PROGRESSION"  
  
# create an "overall survival" variable that is equal to days\_to\_death  
# for dead patients, and to days\_to\_last\_follow\_up for patients who  
# are still alive  
data\_clinical\_common\_group2$overall\_survival = ifelse(data\_clinical\_common\_group2$deceased,  
 data\_clinical\_common$OS\_MONTHS,  
 data\_clinical\_common$DAYS\_LAST\_FOLLOWUP)  
  
# remove any of the letters "a", "b" or "c", but only if they are at the end  
# of the name, eg "stage iiia" would become simply "stage iii"  
data\_clinical\_common\_group2$AJCC\_PATHOLOGIC\_TUMOR\_STAGE = gsub("[ABC]$", "", data\_clinical\_common\_group2$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)  
  
data\_clinical\_common\_group2[which(data\_clinical\_common\_group2$AJCC\_PATHOLOGIC\_TUMOR\_STAGE == ""), "AJCC\_PATHOLOGIC\_TUMOR\_STAGE"] = NA  
  
table(data\_clinical\_common\_group2$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)

##   
## STAGE I STAGE II STAGE III   
## 25 12 7

fit = survfit(Surv(overall\_survival, deceased) ~ AJCC\_PATHOLOGIC\_TUMOR\_STAGE, data=data\_clinical\_common\_group2)  
  
# we can extract the survival p-value and print it  
pval = surv\_pvalue(fit, data=data\_clinical\_common\_group2)$pval  
print(pval)

## [1] 0.9941215

ggsurvplot(fit, data=data\_clinical\_common\_group2, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35,legend.lab = c("STAGE I", "STAGE II", "STAGE III"), title = "Survivability of cluster 2 patients by tumor stage")

 ### SA on cluster 2 Sex

Surv(data\_clinical\_common\_group2$overall\_survival, data\_clinical\_common\_group2$deceased) ~ data\_clinical\_common\_group2$SEX

## Surv(data\_clinical\_common\_group2$overall\_survival, data\_clinical\_common\_group2$deceased) ~   
## data\_clinical\_common\_group2$SEX

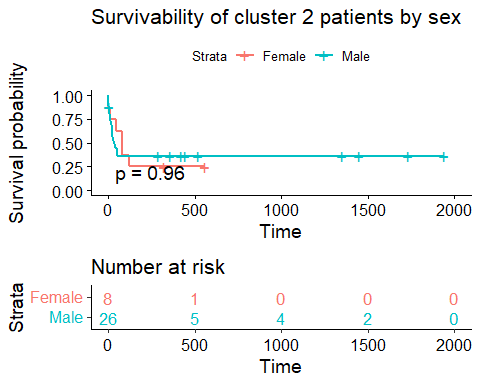
fit = survfit(Surv(overall\_survival, deceased) ~ SEX, data = data\_clinical\_common\_group2)  
  
table(data\_clinical\_common\_group2$SEX)

##   
## Female Male   
## 10 35

pval = surv\_pvalue(fit, data=data\_clinical\_common\_group2)$pval  
print(pval)

## [1] 0.9592107

ggsurvplot(fit, data=data\_clinical\_common\_group2, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35, legend.lab = c("Female", "Male"), title = "Survivability of cluster 2 patients by sex")

 ## SA on the most mutated gene (TP53) ### SA by sex

data\_clinical\_common\_TP53$deceased = data\_clinical\_common\_TP53$PFS\_STATUS == "1:PROGRESSION"  
  
# create an "overall survival" variable that is equal to days\_to\_death  
# for dead patients, and to days\_to\_last\_follow\_up for patients who  
# are still alive  
data\_clinical\_common\_TP53$overall\_survival = ifelse(data\_clinical\_common\_TP53$deceased,  
 data\_clinical\_common\_TP53$OS\_MONTHS,  
 data\_clinical\_common\_TP53$DAYS\_LAST\_FOLLOWUP)  
  
  
Surv(data\_clinical\_common\_TP53$overall\_survival, data\_clinical\_common\_TP53$deceased) ~ data\_clinical\_common\_TP53$SEX

## Surv(data\_clinical\_common\_TP53$overall\_survival, data\_clinical\_common\_TP53$deceased) ~   
## data\_clinical\_common\_TP53$SEX

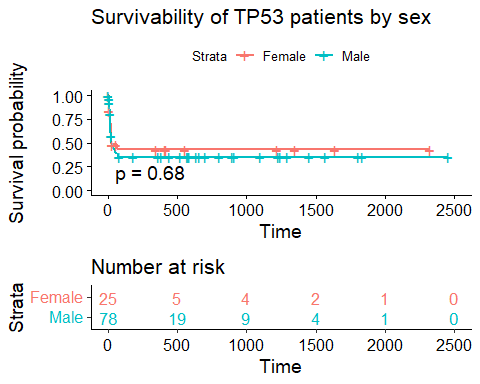
fit = survfit(Surv(overall\_survival, deceased) ~ SEX, data = data\_clinical\_common\_TP53)  
  
table(data\_clinical\_common\_TP53$SEX)

##   
## Female Male   
## 26 84

pval = surv\_pvalue(fit, data=data\_clinical\_common\_TP53)$pval  
print(pval)

## [1] 0.6845213

ggsurvplot(fit, data=data\_clinical\_common\_TP53, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35,legend.lab = c("Female", "Male"), title = "Survivability of TP53 patients by sex" )

 ### SA by tumor stage

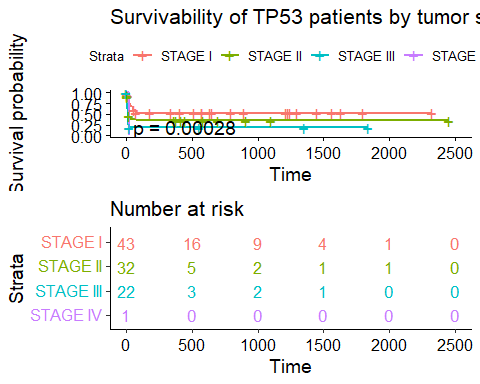
# remove any of the letters "a", "b" or "c", but only if they are at the end  
# of the name, eg "stage iiia" would become simply "stage iii"  
data\_clinical\_common\_TP53$AJCC\_PATHOLOGIC\_TUMOR\_STAGE = gsub("[ABC]$", "", data\_clinical\_common\_TP53$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)  
  
data\_clinical\_common\_TP53[which(data\_clinical\_common\_TP53$AJCC\_PATHOLOGIC\_TUMOR\_STAGE == ""), "AJCC\_PATHOLOGIC\_TUMOR\_STAGE"] = NA  
  
  
table(data\_clinical\_common\_TP53$AJCC\_PATHOLOGIC\_TUMOR\_STAGE)

##   
## STAGE I STAGE II STAGE III STAGE IV   
## 47 33 24 1

fit = survfit(Surv(overall\_survival, deceased) ~ AJCC\_PATHOLOGIC\_TUMOR\_STAGE, data=data\_clinical\_common\_TP53)  
  
# we can extract the survival p-value and print it  
pval = surv\_pvalue(fit, data=data\_clinical\_common\_TP53)$pval  
print(pval)

## [1] 0.0002790916

ggsurvplot(fit, data=data\_clinical\_common\_TP53, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.50, legend.labs = c("STAGE I", "STAGE II", "STAGE III", "STAGE IV"), title = "Survivability of TP53 patients by tumor stage")

 ### SA by cluster with mutation data

data\_clinical\_common\_TP53$cluster = cluster[data\_clinical\_common\_TP53$PATIENT\_ID,]  
  
Surv(data\_clinical\_common\_TP53$overall\_survival, data\_clinical\_common\_TP53$deceased) ~ data\_clinical\_common\_TP53$cluster

## Surv(data\_clinical\_common\_TP53$overall\_survival, data\_clinical\_common\_TP53$deceased) ~   
## data\_clinical\_common\_TP53$cluster

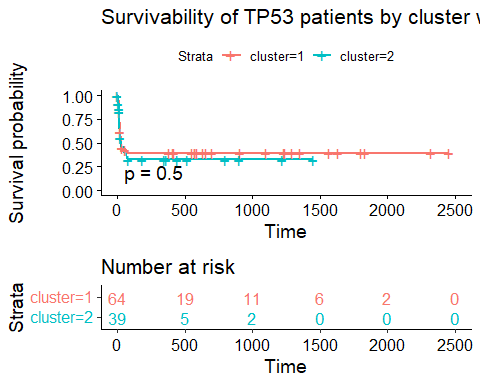
fit = survfit(Surv(overall\_survival, deceased) ~ cluster, data = data\_clinical\_common\_TP53)  
  
table(data\_clinical\_common\_TP53$cluster)

##   
## 1 2   
## 65 45

pval = surv\_pvalue(fit, data=data\_clinical\_common\_TP53)$pval  
print(pval)

## [1] 0.5026988

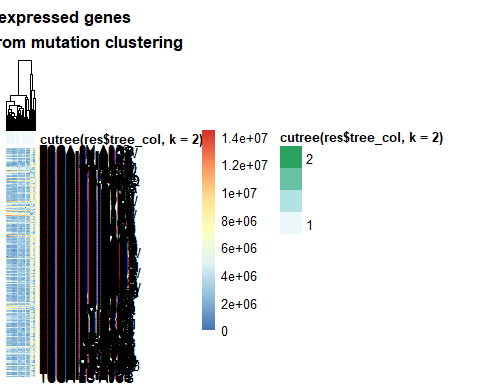
ggsurvplot(fit, data=data\_clinical\_common\_TP53, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35, title = "Survivability of TP53 patients by cluster with mutation data")

 # Differential Expression analysis

data\_rna\_common <- data\_rna\_common[rowSums(data\_rna\_common)>1,]  
# head(data\_rna\_common)

sampleDists = dist(t(data\_rna\_common),upper = TRUE)  
# sampleDists

# annot\_col = data.frame(colData$condition)  
# row.names(annot\_col) <- rownames(colData)  
  
sampleDistMatrix = as.matrix( sampleDists )  
rownames(sampleDistMatrix) = colnames(data\_rna\_common)  
colnames(sampleDistMatrix) = colnames(data\_rna\_common)  
  
res\_rna = pheatmap(sampleDistMatrix,  
 clustering\_distance\_rows = sampleDists,  
 clustering\_distance\_cols = sampleDists,  
 cluster\_rows=FALSE, show\_rownames=TRUE,  
 cluster\_cols=TRUE, shor\_colnames = FALSE, show\_colnames = FALSE,  
 annotation\_col=cluster, main = "Distance of expressed genes \n with annotation from mutation clustering",)



cluster\_rna = as.data.frame(cutree(res\_rna$tree\_col, k = 2))  
# cluster\_rna

Adding cluster\_rna cluster and survival analysis

data\_clinical\_common$deceased = data\_clinical\_common$PFS\_STATUS == "1:PROGRESSION"  
  
# create an "overall survival" variable that is equal to days\_to\_death  
# for dead patients, and to days\_to\_last\_follow\_up for patients who  
# are still alive  
data\_clinical\_common$overall\_survival = ifelse(data\_clinical\_common$deceased,  
 data\_clinical\_common$OS\_MONTHS,  
 data\_clinical\_common$DAYS\_LAST\_FOLLOWUP)  
  
data\_clinical\_common$cluster\_rna = cluster\_rna[data\_clinical\_common$PATIENT\_ID,]  
  
Surv(data\_clinical\_common$overall\_survival, data\_clinical\_common$deceased) ~ data\_clinical\_common$cluster\_rna

## Surv(data\_clinical\_common$overall\_survival, data\_clinical\_common$deceased) ~   
## data\_clinical\_common$cluster\_rna

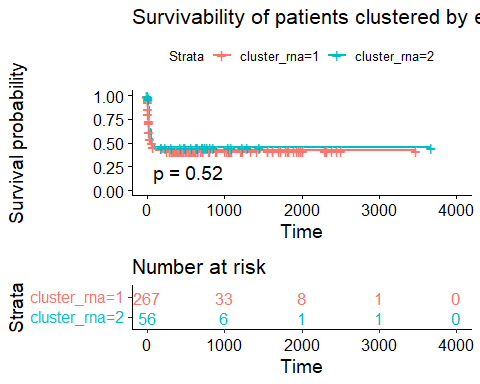
table(data\_clinical\_common$cluster\_rna)

##   
## 1 2   
## 291 62

fit = survfit(Surv(overall\_survival, deceased) ~ cluster\_rna, data = data\_clinical\_common)  
  
pval = surv\_pvalue(fit, data=data\_clinical\_common)$pval  
print(pval)

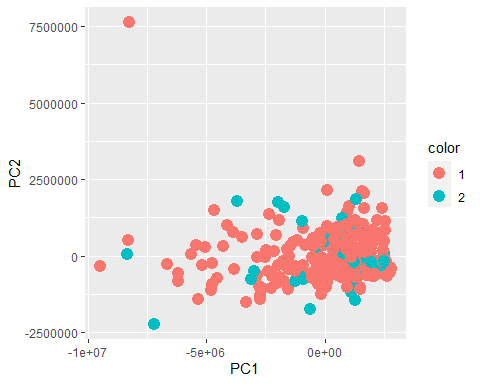
## [1] 0.518244

ggsurvplot(fit, data=data\_clinical\_common, pval=T, risk.table=T, risk.table.col="strata", risk.table.height=0.35, title = "Survivability of patients clustered by expression data")



PCA plot

rownames(cluster) = substr(rownames(cluster), start = 1, stop = 12)  
common\_cluster <- row.names(cluster)  
  
data\_rna\_common\_cluster <- data\_rna\_common[,common\_cluster]  
  
colnames(data\_rna\_common\_cluster) = rownames(cluster)  
  
pca\_res <- prcomp(t(data\_rna\_common\_cluster), scale. = FALSE)  
score <- pca\_res$x  
  
score = as.data.frame(score)  
score$color <- as.factor(cluster$`cutree(res$tree\_col, k = 2)`)  
  
ggplot(score, aes(x=PC1, y=PC2, color=color)) +  
 geom\_point(size = 4)



Actually running the DESeq pipeline

rownames(cluster) = substr(rownames(cluster), start = 1, stop = 12)  
common\_cluster <- row.names(cluster)  
  
data\_rna\_common\_cluster <- data\_rna\_common[,common\_cluster]  
  
colnames(data\_rna\_common\_cluster) = rownames(cluster)  
  
metadata <- data.frame(  
 patientID = row.names(cluster),  
 condition = cluster$`cutree(res$tree\_col, k = 2)`  
)  
  
### making metadata a factor so DESeq2 runs faster  
row.names(metadata) = metadata$patientID  
metadata$patientID = NULL  
metadata$condition = factor(metadata$condition)

dds <- DESeqDataSetFromMatrix(countData = data\_rna\_common\_cluster, colData = metadata, design = ~condition)

dds = DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 5788 genes  
## -- DESeq argument 'minReplicatesForReplace' = 7   
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

dds

## class: DESeqDataSet   
## dim: 54571 285   
## metadata(1): version  
## assays(6): counts mu ... replaceCounts replaceCooks  
## rownames(54571): ENSG00000000003.15 ENSG00000000005.6 ...  
## ENSG00000288674.1 ENSG00000288675.1  
## rowData names(23): baseMean baseVar ... maxCooks replace  
## colnames(285): TCGA-2V-A95S TCGA-2Y-A9GS ... TCGA-ZS-A9CF TCGA-ZS-A9CG  
## colData names(3): condition sizeFactor replaceable

print("Result table of DESeq")

## [1] "Result table of DESeq"

res\_de = results(dds, contrast=c(1, 2))  
mcols(res\_de, use.names = TRUE)

## DataFrame with 6 rows and 2 columns  
## type description  
## <character> <character>  
## baseMean intermediate mean of normalized c..  
## log2FoldChange results log2 fold change (ML..  
## lfcSE results standard error: +1,+2  
## stat results Wald statistic: +1,+2  
## pvalue results Wald test p-value: +..  
## padj results BH adjusted p-values

summary(res\_de)

##   
## out of 54493 with nonzero total read count  
## adjusted p-value < 0.1  
## LFC > 0 (up) : 25010, 46%  
## LFC < 0 (down) : 1934, 3.5%  
## outliers [1] : 0, 0%  
## low counts [2] : 11625, 21%  
## (mean count < 0)  
## [1] see 'cooksCutoff' argument of ?results  
## [2] see 'independentFiltering' argument of ?results

print("\n Number of genes with adjusted p-value that is less than 0.05")

## [1] "\n Number of genes with adjusted p-value that is less than 0.05"

res\_de.05 <- results(dds, alpha = 0.05)  
table(res\_de.05$padj < 0.05)

##   
## FALSE TRUE   
## 29379 1873

print("\n number of genes with log2 fold more than doubling (p-value < 0.1)")

## [1] "\n number of genes with log2 fold more than doubling (p-value < 0.1)"

resLFC1 <- results(dds, lfcThreshold=1)  
table(resLFC1$padj < 0.1)

##   
## FALSE TRUE   
## 27003 23

P-values

res\_de <- res\_de[order(res\_de$pvalue),]  
summary(res\_de)

##   
## out of 54493 with nonzero total read count  
## adjusted p-value < 0.1  
## LFC > 0 (up) : 25010, 46%  
## LFC < 0 (down) : 1934, 3.5%  
## outliers [1] : 0, 0%  
## low counts [2] : 11625, 21%  
## (mean count < 0)  
## [1] see 'cooksCutoff' argument of ?results  
## [2] see 'independentFiltering' argument of ?results

## number of adjusted p-values less than 0.1  
sum(res\_de$padj < 0.1, na.rm=TRUE)

## [1] 26944

multiple testing

# sum(res\_de$pvalue < 0.05, na.rm=TRUE)  
# sum(!is.na(res\_de$pvalue))  
# sum(res\_de$padj < 0.05, na.rm=TRUE)  
  
resSig <- subset(res\_de, padj < 0.06)  
resSig <- subset(res\_de, padj < 0.06)  
head(resSig[order( resSig$log2FoldChange ), ])

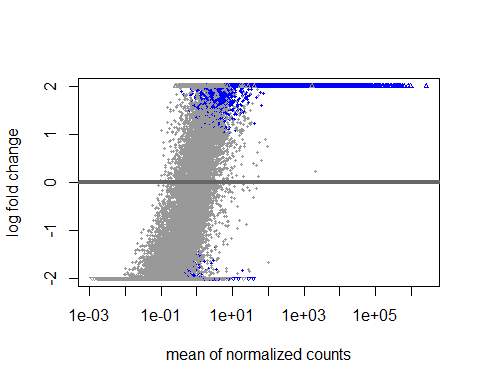
## log2 fold change (MLE): +1,+2   
## Wald test p-value: +1,+2   
## DataFrame with 6 rows and 6 columns  
## baseMean log2FoldChange lfcSE stat pvalue  
## <numeric> <numeric> <numeric> <numeric> <numeric>  
## ENSG00000250385.2 1.81257 -5.85375 2.05799 -2.84441 4.44944e-03  
## ENSG00000285933.1 5.59978 -5.55253 1.46623 -3.78694 1.52516e-04  
## ENSG00000142515.15 14.61962 -5.46463 2.17753 -2.50955 1.20884e-02  
## ENSG00000249641.2 1.12196 -5.15249 2.15537 -2.39054 1.68238e-02  
## ENSG00000284779.2 1.47193 -4.86638 1.62221 -2.99985 2.70116e-03  
## ENSG00000251630.1 1.77364 -4.76246 1.17470 -4.05420 5.03066e-05  
## padj  
## <numeric>  
## ENSG00000250385.2 8.07708e-03  
## ENSG00000285933.1 2.94388e-04  
## ENSG00000142515.15 2.12836e-02  
## ENSG00000249641.2 2.92012e-02  
## ENSG00000284779.2 4.96885e-03  
## ENSG00000251630.1 9.87475e-05

head(resSig[order(resSig$log2FoldChange, decreasing=TRUE),], n= 20)

## log2 fold change (MLE): +1,+2   
## Wald test p-value: +1,+2   
## DataFrame with 20 rows and 6 columns  
## baseMean log2FoldChange lfcSE stat pvalue  
## <numeric> <numeric> <numeric> <numeric> <numeric>  
## ENSG00000163631.17 2610593 21.1939 0.446255 47.4927 0  
## ENSG00000087086.15 545819 20.2335 0.327027 61.8712 0  
## ENSG00000197249.14 963226 19.6484 0.347743 56.5027 0  
## ENSG00000198804.2 809312 19.4548 0.279884 69.5104 0  
## ENSG00000158874.11 558465 19.4243 0.458847 42.3328 0  
## ... ... ... ... ... ...  
## ENSG00000198888.2 232394 17.9615 0.283928 63.2608 0.00000e+00  
## ENSG00000171560.16 506679 17.8472 0.427564 41.7415 0.00000e+00  
## ENSG00000257017.9 406341 17.8178 0.565548 31.5054 7.33251e-218  
## ENSG00000106927.12 284987 17.7434 0.375081 47.3056 0.00000e+00  
## ENSG00000198786.2 151522 17.7019 0.360167 49.1491 0.00000e+00  
## padj  
## <numeric>  
## ENSG00000163631.17 0  
## ENSG00000087086.15 0  
## ENSG00000197249.14 0  
## ENSG00000198804.2 0  
## ENSG00000158874.11 0  
## ... ...  
## ENSG00000198888.2 0.00000e+00  
## ENSG00000171560.16 0.00000e+00  
## ENSG00000257017.9 3.64601e-217  
## ENSG00000106927.12 0.00000e+00  
## ENSG00000198786.2 0.00000e+00

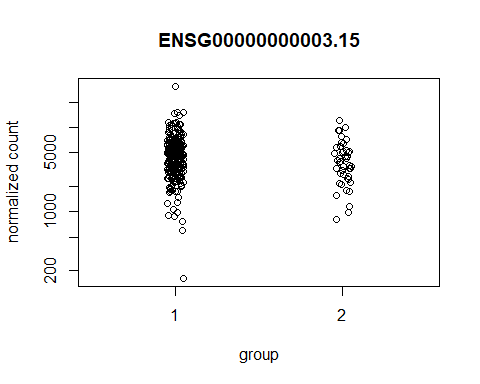
MA-plot

plotMA(res\_de, ylim=c(-2,2))



Plot counts

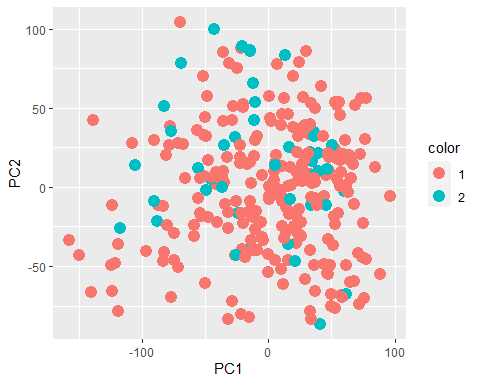
plotCounts(dds, gene=which.min(res\_de$padj), intgroup="condition")

 Effect of transformations on the variance

# this gives log2(n + 1)  
ntd <- normTransform(dds)  
# Variance stabilizing transformation  
vsd <- vst(dds)  
  
# Regularized log transformation  
# The blind=TRUE argument results in a transformation unbiased to sample condition information.  
# rld <- vst(dds, blind=FALSE)

# sampleDists = dist(t(assay(rld)),upper = TRUE)  
#   
# # annot\_col = data.frame(cluster$`cutree(res$tree\_col, k = 2)`)  
# # row.names(annot\_col) <- rownames(clusters)  
#   
# sampleDistMatrix = as.matrix( sampleDists )  
# rownames(sampleDistMatrix) = colnames(data\_rna\_common)  
# colnames(sampleDistMatrix) = colnames(data\_rna\_common)  
#   
# pheatmap(sampleDistMatrix,  
# clustering\_distance\_rows = sampleDists,  
# clustering\_distance\_cols = sampleDists,  
# cluster\_rows=FALSE, show\_rownames=TRUE,  
# cluster\_cols=TRUE,  
# annotation\_col=cluster)

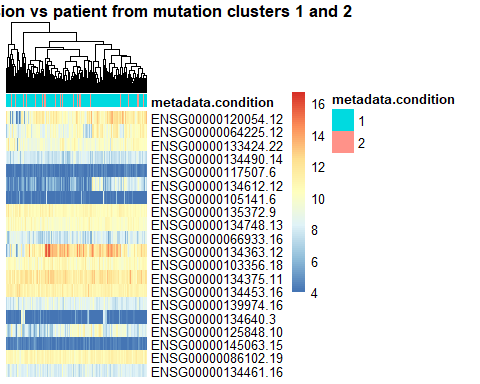
pca\_res <- prcomp(t(assay(vsd)), scale. = FALSE)  
score <- pca\_res$x  
  
score = as.data.frame(score)  
score$color <- as.factor(cluster$`cutree(res$tree\_col, k = 2)`)  
  
  
ggplot(score, aes(x=PC1, y=PC2, color=color)) +  
 geom\_point(size = 4)



Selecting the top 20 upregulated (or top 20 downregulated)

# Top20genes = head(resSig[order(resSig$log2FoldChange, decreasing=TRUE),], n= 20)  
  
# genes <- order(res$padj, decreasing = TRUE) [1:20]  
  
# we can select a subset of genes to plot.let’s choose the 20 genes with the largest positive log2fold change.  
genes <- order(res\_de$log2FoldChange,decreasing = TRUE)[1:20]  
  
# or largest negative log2fold change  
# genes <- order(res$log2FoldChange, decreasing = FALSE)[1:20]  
  
# or we can select the top 20 significant genes

annot\_col = data.frame(metadata$condition)  
row.names(annot\_col) <- rownames(metadata)  
  
sampleMatrix <- assay(vsd)[genes,]  
  
rownames(sampleMatrix) = rownames(data\_rna\_common\_cluster[genes,])  
colnames(sampleMatrix) = colnames(data\_rna\_common\_cluster)  
  
pheatmap(sampleMatrix , cluster\_rows=FALSE, show\_rownames=TRUE, show\_colnames = FALSE,  
 cluster\_cols=TRUE, annotation\_col=annot\_col, main = "Heatmap of Gene expression vs patient from mutation clusters 1 and 2")



# Pathway analysis

Adding gene notation

columns(org.Hs.eg.db)

## [1] "ACCNUM" "ALIAS" "ENSEMBL" "ENSEMBLPROT" "ENSEMBLTRANS"  
## [6] "ENTREZID" "ENZYME" "EVIDENCE" "EVIDENCEALL" "GENENAME"   
## [11] "GENETYPE" "GO" "GOALL" "IPI" "MAP"   
## [16] "OMIM" "ONTOLOGY" "ONTOLOGYALL" "PATH" "PFAM"   
## [21] "PMID" "PROSITE" "REFSEQ" "SYMBOL" "UCSCKG"   
## [26] "UNIPROT"

row.names(res\_de)= substr(row.names(res\_de), start = 1, stop = 15)  
res\_de$symbol = mapIds(org.Hs.eg.db,  
 keys=row.names(res\_de),   
 column="SYMBOL",  
 keytype="ENSEMBL",  
 multiVals="first")

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

res\_de$entrez = mapIds(org.Hs.eg.db,  
 keys=row.names(res\_de),   
 column="ENTREZID",  
 keytype="ENSEMBL",  
 multiVals="first")

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

res\_de$name = mapIds(org.Hs.eg.db,  
 keys=row.names(res\_de),   
 column="GENENAME",  
 keytype="ENSEMBL",  
 multiVals="first")

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

head(res\_de, 10)

## log2 fold change (MLE): +1,+2   
## Wald test p-value: +1,+2   
## DataFrame with 10 rows and 9 columns  
## baseMean log2FoldChange lfcSE stat pvalue  
## <numeric> <numeric> <numeric> <numeric> <numeric>  
## ENSG00000000003 4974.658 11.84570 0.246020 48.1494 0  
## ENSG00000000419 1245.483 10.23732 0.139225 73.5307 0  
## ENSG00000000457 542.066 8.78990 0.170892 51.4354 0  
## ENSG00000000971 58558.734 15.32732 0.352084 43.5331 0  
## ENSG00000001036 2891.948 11.94004 0.224317 53.2283 0  
## ENSG00000001084 4340.614 12.74366 0.270735 47.0707 0  
## ENSG00000001167 926.293 10.05109 0.205238 48.9728 0  
## ENSG00000001461 444.561 8.95119 0.188333 47.5284 0  
## ENSG00000001497 2030.143 11.09935 0.160704 69.0672 0  
## ENSG00000001617 923.542 10.10906 0.204536 49.4244 0  
## padj symbol entrez name  
## <numeric> <character> <character> <character>  
## ENSG00000000003 0 TSPAN6 7105 tetraspanin 6  
## ENSG00000000419 0 DPM1 8813 dolichyl-phosphate m..  
## ENSG00000000457 0 SCYL3 57147 SCY1 like pseudokina..  
## ENSG00000000971 0 CFH 3075 complement factor H  
## ENSG00000001036 0 FUCA2 2519 alpha-L-fucosidase 2  
## ENSG00000001084 0 GCLC 2729 glutamate-cysteine l..  
## ENSG00000001167 0 NFYA 4800 nuclear transcriptio..  
## ENSG00000001461 0 NIPAL3 57185 NIPA like domain con..  
## ENSG00000001497 0 LAS1L 81887 LAS1 like ribosome b..  
## ENSG00000001617 0 SEMA3F 6405 semaphorin 3F

library("org.Hs.eg.db")  
library(pathview)  
library(gage)  
library(gageData)  
kegg.sets.hs = data(kegg.sets.hs)  
sigmet.idx.hs = data(sigmet.idx.hs)  
  
# Focus on signaling and metabolic pathways only  
kegg.sets.hs = kegg.sets.hs[sigmet.idx.hs]  
kegg.sets.hs = list(kegg.sets.hs)  
# Examine the first 3 pathways  
# head(kegg.sets.hs, 3)

foldchanges = res\_de$log2FoldChange  
names(foldchanges) = res\_de$entrez  
head(foldchanges)

## 7105 8813 57147 3075 2519 2729   
## 11.845697 10.237323 8.789901 15.327318 11.940039 12.743663

# Get the results  
keggres = gage(foldchanges, gsets=kegg.sets.hs)

attributes(keggres)

## $names  
## [1] "greater" "less" "stats"

# Look at the first few up (greater) pathways  
head(keggres$greater)

## p.geomean stat.mean p.val q.val set.size exp1  
## [1,] NA NaN NA NA 1 NA

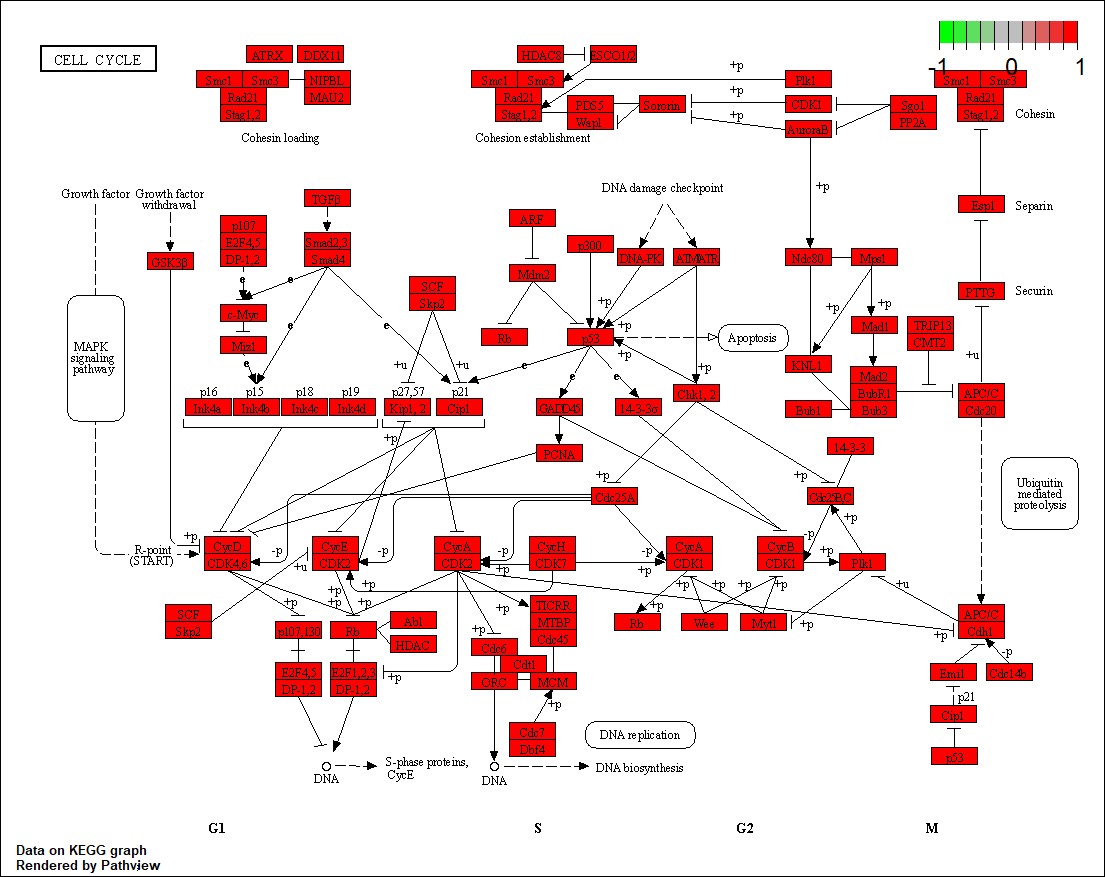
pathview(gene.data=foldchanges, pathway.id="hsa04110")

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

## Info: Working in directory D:/Classes/BMEG 310/BMEG310\_FinalProject

## Info: Writing image file hsa04110.pathview.png

knitr::include\_graphics("hsa04110.pathview.png")



## Focus on top 5 upregulated pathways here for demo purposes only  
keggrespathways <- rownames(keggres$greater)[1:1]  
  
# Extract the 8 character long IDs part of each string  
keggresids = substr(keggrespathways, start=1, stop=8)  
keggresids

## character(0)

pathview(gene.data=foldchanges, pathway.id=keggresids, species="hsa")

## Info: Downloading xml files for hsaNA, 1/1 pathways..

## Warning in download.file(xml.url, xml.target, quiet = T): cannot open URL  
## 'https://rest.kegg.jp/get/hsaNA/kgml': HTTP status was '400 Bad Request'

## Warning: Download of hsaNA xml file failed!  
## This pathway may not exist!

## Info: Downloading png files for hsaNA, 1/1 pathways..

## Warning: Download of hsaNA png file failed!  
## This pathway may not exist!

## Warning: Failed to download KEGG xml/png files, hsa skipped!

knitr::include\_graphics("hsa04141.pathview.png")

