1. Heatmap

# 清空工作环境中的对象

rm(list = ls())

# 加载所需的R包

library("pacman") # 使用pacman包管理器加载多个包

p\_load(readr, data.table, tidyverse, limma, pheatmap, ggplot2)

# 列出当前工作目录下的文件

dir()

# 读取数据和分类信息

data <- read.csv("samples.csv", header = TRUE, row.names = 1) # 读取样本数据

Group <- read.csv("group.csv", header = TRUE, row.names = 1) # 读取分类信息

# 输出数据列名

colnames(data)

# 读取数据和分类信息

data1 <- read.csv("samples-MEAN.csv", header = TRUE, row.names = 1) # 读取样本均值数据

Group1 <- read.csv("group-MEAN.csv", header = TRUE, row.names = 1) # 读取分类信息

# 读取数据和分类信息

data2 <- read.csv("samples-MEAN.csv", header = TRUE, row.names = 1) # 读取样本中位数数据

Group2 <- read.csv("group-MEAN.csv", header = TRUE, row.names = 1) # 读取分类信息

# 绘制热图

# 绘制热图并保存为heatmap.pdf

pheatmap(data,

annotation = Group, # 分类信息

colorRampPalette(c("navy", "white", "firebrick3"))(100), # 色带

cluster\_cols = F,

cluster\_rows = F,

Rowv=FALSE,# 是否对列进行聚类

show\_colnames = T, # 是否显示列名

show\_rownames = T, # 是否显示行名

scale = "row", # 行缩放

fontsize = 10, # 字体大小

fontsize\_row = 8, # 行名字体大小

fontsize\_col = 10, # 列名字体大小

border = FALSE, # 是否显示边框

angle\_col = 90) # 列名文字角度

dev.off()

# 绘制热图

# 绘制热图并保存为heatmap1.pdf

pdf(file = "heatmap-MEAN.pdf", height = 12, width = 8)

pheatmap(data1,

annotation = Group1, # 分类信息

colorRampPalette(c("navy", "white", "firebrick3"))(100), # 色带

cluster\_cols = F, # 是否对列进行聚类

show\_colnames = F, # 是否显示列名

show\_rownames = F, # 是否显示行名

scale = "row", # 行缩放

fontsize = 10, # 字体大小

fontsize\_row = 8, # 行名字体大小

fontsize\_col = 10, # 列名字体大小

border = FALSE, # 是否显示边框

angle\_col = 90) # 列名文字角度

dev.off()

# 绘制热图

# 绘制热图并保存为heatmap2.pdf

pdf(file = "heatmap-MEDIAN.pdf", height = 12, width = 8)

pheatmap(data2,

annotation = Group2, # 分类信息

colorRampPalette(c("navy", "white", "firebrick3"))(100), # 色带

cluster\_cols = F, # 是否对列进行聚类

show\_colnames = T, # 是否显示列名

show\_rownames = T, # 是否显示行名

scale = "row", # 行缩放

fontsize = 10, # 字体大小

fontsize\_row = 8, # 行名字体大小

fontsize\_col = 10, # 列名字体大小

border = FALSE, # 是否显示边框

angle\_col = 90) # 列名文字角度

dev.off()

1. Network

rm(list=ls())

library(igraph)

library(Hmisc)

library(cooccurNet)

library(psych)

library(vegan)

library(igraph)

library(Hmisc)

co\_occurrence\_network<-function(matrix,cor.cutoff,p.cutoff){

matrix1<-matrix

matrix1[matrix1>0]<-1

#correlation analysis based on spearman's co-efficient

matrix.dist<-rcorr(t(matrix),type="spearman")

###matrix.dist<-rcorr(t(matrix),type="pearson")

matrix.cor<-matrix.dist$r

matrix.cor.p<-matrix.dist$P

#Multiple testing correction using Benjamini-Hochberg standard false discovery rate correction ("FDR-BH")

matrix.cor.p <- p.adjust(matrix.cor.p, method="BH")

#1.Consider positive cooccurence at given coefficient (cor.cutoff) and p-value cutoffs

matrix.cor1<-matrix.cor

matrix.cor1.p<-matrix.cor.p

matrix.cor1[which(matrix.cor1 <= cor.cutoff)]=0

matrix.cor1[which(matrix.cor1.p>p.cutoff)]=0

# delete those rows and columns with sum = 0

matrix.cor1<-matrix.cor1[which(rowSums(matrix.cor1)!=1),]

matrix.cor1<-matrix.cor1[,which(colSums(matrix.cor1)!=0)]

#2.Consider netagive cooccurence at given coefficient (-cor.cutoff) and p-value cutoffs

matrix.cor2<-matrix.cor

matrix.cor2.p<-matrix.cor.p

matrix.cor2[which(matrix.cor2 > (-cor.cutoff))]=0

matrix.cor2[which(matrix.cor2.p>p.cutoff)]=0

# delete those rows and columns with sum = 0

matrix.cor2<-matrix.cor2[which(rowSums(matrix.cor2)!=0),]

matrix.cor2<-matrix.cor2[,which(colSums(matrix.cor2)!=0)]

#3.Consider both positive and netagive cooccurence at given coefficient (cor.cutoff) and p-value cutoffs

matrix.cor3<-matrix.cor

matrix.cor3.p<-matrix.cor.p

matrix.cor3[which(matrix.cor3>=(-cor.cutoff) & matrix.cor3 <= cor.cutoff)]=0

matrix.cor3[which(matrix.cor3.p>p.cutoff)]=0

# delete those rows and columns with sum = 0

matrix.cor3<-matrix.cor3[which(rowSums(matrix.cor3)!=1),]

matrix.cor3<-matrix.cor3[,which(colSums(matrix.cor3)!=0)]

# generate graph using igraph

g1<-graph.adjacency(matrix.cor1,weight=T,mode="undirected")

g1<-simplify(g1)

V(g1)$label <- V(g1)$name

V(g1)$degree <- degree(g1)

g2<-graph.adjacency(matrix.cor2,weight=T,mode="undirected")

g2<-simplify(g2)

V(g2)$label <- V(g2)$name

V(g2)$degree <- degree(g2)

g3<-graph.adjacency(matrix.cor3,weight=T,mode="undirected")

g3<-simplify(g3)

V(g3)$label <- V(g3)$name

V(g3)$degree <- degree(g3)

# append the output into results

result<-list()

result$matrix.cor<-matrix.cor

result$matrix.cor.p<-matrix.cor.p

result$matrix.cor1<-matrix.cor1

result$graph1<-g1

###result$matrix.cor2<-matrix.cor2

###result$graph2<-g2

result$matrix.cor3<-matrix.cor3

result$graph3<-g3

return(result)

}

##############################################################

Abu=read.csv('ins.csv',header=T,row.names = 1)

Abu<-as.matrix(Abu)

###1. Filtering OTUs by occurrence frequency (i.e.,number of samples an OTU is Present)

table<-Abu

table[table>0]<-1

table.generalist<-Abu[which(rowSums(table)>=8),]

Abu<-table.generalist

###2. Creating gml files of network (to be visulized in Gephi or Cytoscape)

pattern<-co\_occurrence\_network(Abu,0.6, 0.01) ## cutoffs for correlation coefficient and P-value

#write.graph(pattern$graph1,'Zhuhai-average-Pos0.6.gml',format='gml') #network file for positive association

#write.graph(pattern$graph2,'Neg0.6-NW.gml',format='gml') #network file for negative association (if any)

write.graph(pattern$graph3,'C-average-PosNeg0.6.gml',format='gml') #network file for all association

###3. Calculating network topological properties

g<-pattern$graph1 ###positive network

#g<-pattern$graph1 ###negative network

c <- cluster\_walktrap(g)

# Global toplogical features

modularity(c)

md <- modularity(g, membership(c), weights = NULL)

cc <- transitivity(g, vids = NULL,

weights = NULL)

spl <- average.path.length(g, directed=FALSE, unconnected=TRUE)

gd <- graph.density(g, loops=FALSE)

nd <- diameter(g, directed = FALSE, unconnected = TRUE, weights = NA)

node.degree <- degree(g, v = V(g), mode="all")

ad <- mean(node.degree)

e <- ecount(g)

v <- vcount(g)

global.topology <- data.frame(e,v,cc,spl,md,gd,nd,ad)

write.csv(global.topology, file="C2-average-Pos0.7-global.topology.csv")

# Node toplogical features

betweenness.centrality <- betweenness(g, v=V(g),

directed = FALSE, weights = NA,

nobigint = TRUE, normalized = FALSE)

closeness.centrality <- closeness(g, vids = V(g),

weights = NA, normalized = FALSE)

node.transitivity <- transitivity(g, type = c("local"), vids = NULL,

weights = NA)

node.topology <- data.frame(node.degree, betweenness.centrality, closeness.centrality, node.transitivity)

write.csv(node.topology, file="FS-average-Pos0.8-node.topology.csv")

# Ploting node degreee distribution in a log-log plot

degree.df <- data.frame(table(degree=factor(node.degree, levels=seq\_len(max(node.degree)))))

degree.df$degree <- as.numeric(as.character(degree.df$degree))

#4. Creating an abundance table for OTUs present in the positive and negative network

my.list1 <- row.names(pattern$matrix.cor1)

my.list2 <- row.names(pattern$matrix.cor2)

logical1 <- row.names(Abu) %in% my.list1

logical2 <- row.names(Abu) %in% my.list2

tab.subset1 <- subset(Abu,logical1)tab.subset2 <- subset(Abu,logical2)

write.table(tab.subset1,'S-average-Pos0.7.txt',sep="\t")

write.table(tab.subset2,'Neg0.7-S.txt',sep="\t")

###############################################################

otu\_pro = read.table("C-Phylum.txt",head=T,row.names=1)

head(otu\_pro)

# set vertices size

g.size = otu\_pro[V(g)$name,] # 筛选对应OTU属性

g.size1 = log((g.size$abundance)\*100) # 原始数据是什么，为什么\*100再取e对数

V(g)$size = g.size1

# set vertices color

g.col = otu\_pro[V(g)$name,]

levels(g.size)

levels(g.size) = c("green","deeppink","deepskyblue","yellow","brown","pink","gray","cyan","peachpuff","green","deeppink","deepskyblue","yellow","brown","pink","gray","cyan","peachpuff","green","deeppink","deepskyblue","yellow","brown","pink","gray","cyan","peachpuff","peachpuff") # 直接修改levles可以连值全部对应替换

V(g)$color = as.character(g.col)

set.seed(123)

plot(g,vertex.frame.color=NA,vertex.label=NA,edge.width=1,

vertex.size=3,edge.lty=1,edge.curved=TRUE,margin=c(0,0,0,0))

plot(g)

1. volcano plot

setwd('')

rm(list = ls())

#加载R包

library(ggplot2)

library(tidyverse)

library(ggrepel)

#读取数据

df <- read.csv("data.csv",header = T)

df$label <- ifelse(df$p\_adj<0.05,"adjust Pvalue<0.05","adjust Pvalue>=0.05")

#依次获取每个时期最显著的十个基因(这里根据你需要，复制粘贴改个名就行，废物)

top10sig1 <- filter(df,stage=="1") %>% distinct(geneID,.keep\_all = T) %>% top\_n(10,abs(log2FC))

top10sig2 <- filter(df,stage=="2") %>% distinct(geneID,.keep\_all = T) %>% top\_n(10,abs(log2FC))

top10sig3 <- filter(df,stage=="3") %>% distinct(geneID,.keep\_all = T) %>% top\_n(10,abs(log2FC))

top10sig4 <- filter(df,stage=="4") %>% distinct(geneID,.keep\_all = T) %>% top\_n(10,abs(log2FC))

top10sig5 <- filter(df,stage=="5") %>% distinct(geneID,.keep\_all = T) %>% top\_n(10,abs(log2FC))

top10sig6 <- filter(df,stage=="6") %>% distinct(geneID,.keep\_all = T) %>% top\_n(10,abs(log2FC))

#将提取的基因合并

top10sig <- rbind(top10sig1,top10sig2,top10sig3,top10sig4,top10sig5,top10sig6)

#新增一列，将Top10的差异基因标记为2，其他的标记为1

df$size <- case\_when(!(df$geneID %in% top10sig$geneID)~ 1,

df$geneID %in% top10sig$geneID ~ 2)

#提取非Top10的基因表格；

dt <- filter(df,size==1)

#绘制散点火山图

dt <- filter(df,size==1)

head(dt)

p <- ggplot()+

geom\_jitter(data = dt,

aes(x = stage, y = log2FC, color = label),

size = 0.85,

width =0.4)

p

#叠加每个Cluster Top10基因散点

p <- ggplot()+

geom\_jitter(data = dt,

aes(x = stage, y = log2FC, color = label),

size = 0.85,

width =0.4)+

geom\_jitter(data = top10sig,

aes(x = stage, y = log2FC, color = label),

size = 1,

width =0.4)

p

#根据图p中log2FC区间确定背景柱长度

dfbar<-data.frame(x=c(1,2,3,4),

y=c(14,9,6,14))

dfbar1<-data.frame(x=c(1,2,3,4),

y=c(-3.5,-8,-7.9,-4.3))

p1 <- ggplot()+

geom\_col(data = dfbar,

mapping = aes(x = x,y = y),

fill = "#dcdcdc",alpha = 0.6)+

geom\_col(data = dfbar1,

mapping = aes(x = x,y = y),

fill = "#dcdcdc",alpha = 0.6)

p1

#把散点火山图叠加到背景柱上

p2 <- ggplot()+

geom\_col(data = dfbar,

mapping = aes(x = x,y = y),

fill = "#dcdcdc",alpha = 0.6)+

geom\_col(data = dfbar1,

mapping = aes(x = x,y = y),

fill = "#dcdcdc",alpha = 0.6)+

geom\_jitter(data = dt,

aes(x = stage, y = log2FC, color = label),

size = 0.85,

width =0.4)+

geom\_jitter(data = top10sig,

aes(x = stage, y = log2FC, color = label),

size = 1,

width =0.4)

p2

#添加X轴的stage色块标签（这里记得添加颜色，还有x和label数量也要改，垃圾东西）

dfcol<-data.frame(x=c(1:6),

y=0,

label=c(1:6))

mycol <- c("#4DBBD57F","#00A0877F","#3C54887F","#F39B7F7F","yellow","red")

p3 <- p2 + geom\_tile(data = dfcol,

aes(x=x,y=y),

height=1.7,

color = "black",

fill = mycol,

alpha = 1.7,

show.legend = F)

p3

#给每个stage差异表达前Top10基因加上标签

p4 <- p3+

geom\_text\_repel(

data=top10sig,

aes(x=stage,y=log2FC,label=geneID),

force = 1.2,

arrow = arrow(length = unit(0.008, "npc"),

type = "open", ends = "last")

)

p4

#散点颜色调整

p5 <- p4 +

scale\_color\_manual(name=NULL,

values = c("red","black"))

p5

#修改X/Y轴标题

p6 <- p5+

labs(x="stage",y="average logFC")+

geom\_text(data=dfcol,

aes(x=x,y=y,label=label),

size =6,

color ="white")

p6

#自定义主题美化：

p7 <- p6+

theme\_minimal()+

theme(

axis.title = element\_text(size = 13,

color = "black",

face = "bold"),

axis.line.y = element\_line(color = "black",

size = 1.2),

axis.line.x = element\_blank(),

axis.text.x = element\_blank(),

panel.grid = element\_blank(),

legend.position = "top",

legend.direction = "vertical",

legend.justification = c(1,0),

legend.text = element\_text(size = 15)

)

p7