Análisis de datos ómicos

PEC1

Sandra Vila Garcia

Innhold

Introduccion

Para la realización de esta práctica he decidido utilizar el dataset human_cachexia.csv, obtenido del repertorio de github, que puede visualizarse en el siguiente enlace: metaboData/Datasets/2024-Cachexia at main · nutrimetabolomics/metaboData · GitHub.

El propio repertorio nos proporciona la siguiente descripción de los datos incluidos en el dataset:

This is the famous cachexia dataset used in several MetaboAnalyst tutorials Available from: https://rest.xialab.ca/api/download/metaboanalyst/human_cachexia.csv

Successfully passed sanity check!"
[2] "Samples are not paired."
[3] "2 groups were detected in samples."
[6] "All data values are numeric."
[7] "A total of 0 (0%) missing values were detected."

El archivo human_cachexia.csv contiene información sobre pacientes con caquexia y casos control, incluyendo datos sobre las concentraciones de varios metabolitos. Algunos de los detalles clave del conjunto de datos son:

- Columnas principales:
  - Varias columnas correspondientes a concentraciones de metabolitos específicos, como 1,6-Anhydro-beta-D-glucose, 1-Methylnicotinamide, 2-Aminobutyrate, 3-Hydroxybutyrate, entre otros.
- Estructura:
  - El archivo incluye 65 columnas, de ellas 63 son medidas de metabolitos en diferentes pacientes. Y las dos restantes son:
    - Patient ID: Identificador único de cada paciente.
    - Muscle loss: Categoría que indica si el paciente es "cachexic" (afectado por la pérdida de masa muscular) o caso control.

  - Las concentraciones de metabolitos están representadas por valores numéricos que indican la cantidad presente en cada muestra. Mientras que Patient ID y Muscle loss son variables factoriales.

Este conjunto de datos es valioso para estudiar diferencias metabólicas entre pacientes con caquexia y otras condiciones, permitiendo un análisis profundo de los perfiles metabólicos asociados a la pérdida de masa muscular.

El archivo está estructurado en formato CSV, facilitando su importación y análisis en herramientas estadísticas y de bioinformática.

Objetivos

El objetivo principal de este trabajo es crear un contenedor del tipo SummarizedExperiment que contenga los datos y los metadatos (información acerca del dataset, las filas y las columnas) utilizando R y posteriormente realizar un análisis un análisis exploratorio del dataset.

Objetivos específicos podemos señalar los siguientes:

1. Elección del dataset dentro del repertorio de Github
2. Creacion del contenedor SummarizedExperiment.
3. Realización del análisis exploratorio de los datos para proporcionar una visión general de las variables y de los individuos.
4. Creación de un repositorio en Github que contenga el análisis realizado.

Materiales y metodos

1. Descripción del Dataset

El análisis se realizó utilizando el dataset "human_cachexia.csv", que contiene información sobre pacientes con caquexia y casos, y que está disponible en Github. Este conjunto de datos incluye un total de 77 muestras y 63 metabolitos medidos, incluyendo compuestos como glucosa, aminoácidos y otros metabolitos relevantes para el estudio de la caquexia.

2. Preparación de los Datos

Los datos fueron importados en R utilizando la función read.csv() para crear un objeto de tipo data.frame. Se realizó una limpieza inicial de los datos, eliminando filas con valores faltantes y asegurando que todas las variables relevantes estuvieran en el formato adecuado.

A continuación, se creó un objeto de tipo SummarizedExperiment utilizando la función SummarizedExperiment(). Se extrajeron los datos de los metabolitos (asayData) y los metadatos (colData) correspondientes, asegurando que las dimensiones coincidieran correctamente. El objeto final contenía 77 filas (muestras) y 63 columnas (metabolitos), además de una columna de metadatos que indicaba la pérdida de masa muscular (categorías: "cachexic" y "control").

3. Análisis Exploratorio de Datos

Se utilizó el paquete ggplot2 para realizar visualizaciones del dataset. Se generaron gráficos de boxplot para evaluar la distribución de los metabolitos, con el objetivo de identificar posibles diferencias entre los grupos de pacientes.

4. Análisis Estadístico

Se realizó un Análisis de Componentes Principales (PCA) para explorar la variabilidad entre las muestras y observar patrones en la distribución de los metabolitos.

5. Software y Herramientas

Todo el análisis fue realizado en R (versión 4.3.2) utilizando los siguientes paquetes: SummarizedExperiment para la manipulación de datos, ggplot2 para la visualización, reshape2 para la transformación de datos, pheatmap para crear mapas de calor que visualicen patrones de expresión de los metabolitos y DESeq2 para análisis de expresión diferencial. El código utilizado y el objeto SummarizedExperiment se almacenaron en un repositorio de GitHub con el nombre "Vila-Garcia-Sandra-PEC1".

Resultados

Selección y preparación de los datos para el estudio

Seleccioné el dataset “human_cachexia.csv” del repertorio de Github. Tras importarlo en R procedí a crear el contenedor SummarizedExperiment:

```
# Vamos a crear el objeto assayData
# Primero excluimos las dos primeras columnas ('Patient.ID' y 'Muscle.loss') y lo convertimos a matriz
assayData <- as.matrix(data[, -c(1, 2)])

# Verificamos el número de columnas en assayData
num_samples <- ncol(assayData)
cat("Número de muestras (columnas) en assayData:", num_samples, "\n")
```

```
## Número de muestras (columnas) en assayData: 63
```

```
# Creamos colData solo con las muestras
colData <- data.frame(
  PatientID = as.character(data[["Patient.ID"]]),  # Convertimos a carácter
  MuscleLoss = as.factor(data[["Muscle.loss"]])   # Convertimos a factor
)

# Comprobamos que el número de filas en colData coincida con num_samples
if (nrow(colData) != nrow(data)) {
  stop("El número de filas en colData debe coincidir con el número de muestras en assayData.")
}

# Reducimos colData para que coincida con el número de muestras en assayData
colData <- colData[1:num_samples, ]  # Seleccionar solo las filas correspondientes a las muestras

# Verificamos las dimensiones
cat("Dimensiones de assayData:", dim(assayData), "\n")  # Debe ser [número de metabolitos] x [número de muestras]
```

```
## Dimensiones de assayData: 77 63
```

```
cat("Dimensiones de colData:", dim(colData), "\n")  # Debe ser [número de muestras] x 2
```

```
## Dimensiones de colData: 63 2
```

```
# Y ya podemos crear el objeto SummarizedExperiment
se <- SummarizedExperiment(assays = list(counts = assayData), colData = colData)

# Guardamos el objeto en formato binario
save(se, file = "human_cachexia.Rda")
```

Análisis de los datos del estudio

Una vez creado el contenedor pasamos a hacer una serie de gráficos y análisis para estudiar el dataset.

- Boxplot

  Los boxplot nos sirven para la comparación de grupos: En general, se pueden observar variaciones significativas en las distribuciones de varios metabolitos entre los dos grupos.

- PCA

1. Separación de Grupos: El gráfico de PCA muestra que hay una clara separación entre los dos grupos, "cachexic" y "control", en los dos primeros componentes principales (PC1 y PC2). Esto sugiere que los metabolitos tienen un perfil diferente entre los grupos, lo cual es consistente con los resultados del análisis diferencial, que veremos a continuación.
2. Varianza Explicada: Los ejes de PC1 y PC2 en el PCA generalmente representan la mayor parte de la variación en los datos. Debido a que los grupos se agrupan bien en estos componentes, se puede entender que los metabolitos medidos capturan diferencias biológicamente relevantes entre los estados de caquexia y control.
3. Outliers y Distribución: Observamos varios puntos que se desvían significativamente de los grupos, lo que podría representar casos atípicos. Estos individuos podrían tener características únicas que influyen en sus perfiles de metabolitos.

- Analisis diferencial

El resumen de resultados muestra información clave sobre las diferencias en la expresión de los metabolitos entre los grupos:

- Número de metabolitos analizados:
    - Out of 77 with nonzero total read count: Esto significa que, de los 77 metabolitos analizados, se están considerando solo aquellos que tienen un conteo total mayor que cero en al menos una de las muestras.
- Ajuste de valores p:
    - Adjusted p-value < 0.1: Indica que se han encontrado metabolitos que presentan diferencias significativas con un valor p ajustado menor que 0.1. Aunque esto es algo elevado para que podamos decir que el valor es significativo.
- Diferencias de Metabolitos:
    - LFC > 0 (up): Se encontraron 1 metabolito que fue significativamente más alto en el grupo de "cachexic" comparado con el grupo de "control".
    - LFC < 0 (down): Se encontraron 4 metabolitos que fueron significativamente más bajos en el grupo de " cachexic".
- Outliers y low counts:
    - Outliers [1]: Indica que no hubo metabolitos considerados como outliers en los resultados.
    - Low counts [2]: Indica que no hubo metabolitos con conteos bajos que se hayan excluido del análisis.

Discusión

Conclusiones Generales

- Los resultados sugieren que la caquexia impacta el metabolismo, lo que se refleja en las diferencias en los metabolitos entre los dos grupos. Los metabolitos regulados al alza y a la baja pueden ser utilizados como biomarcadores para comprender mejor el estado de salud de los pacientes con caquexia.
- Estos hallazgos justifican investigaciones adicionales para explorar la funcionalidad de los metabolitos afectados y su posible rol en la fisiopatología de la caquexia. Esos análisis son podrían ayudar al desarrollo de nuevas estrategias terapéuticas o intervenciones dietéticas que podrían ayudar a gestionar los efectos de la caquexia en los pacientes.

Repertorio en Github

Tras la realización del análisis y el informe, he creado el repertorio público "Vila_Garcia_Sandra_PEC1" en Github. Puede accederse a en mediante el siguiente enlace:

https://github.com/Sandra-Vila/Vila_Garcia_Sandra_PEC1.git

Apendice: Código R

Vila_Garcia_Sandra_PEC1

Sandra Vila Garcia

2024-11-04

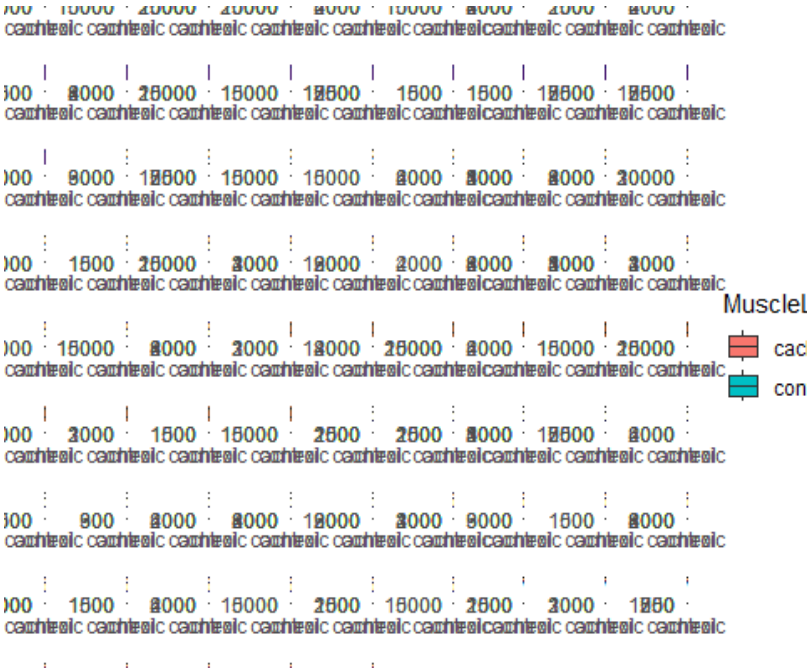
```
# Cargamos las librerías necesarias
library(SummarizedExperiment)

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

## Warning: package 'matrixStats' was built under R version 4.3.3

##
## 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 object is masked from 'package:utils':
##
##     findMatches

## 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

## Warning: package 'GenomeInfoDb' was built under R version 4.3.3

## 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

# Importamos los datos del data set elegido
data <- read.csv("C:/Users/sandr/Downloads/human_cachexia.csv")

# Verificamos los nombres de las columnas
print(colnames(data))
```

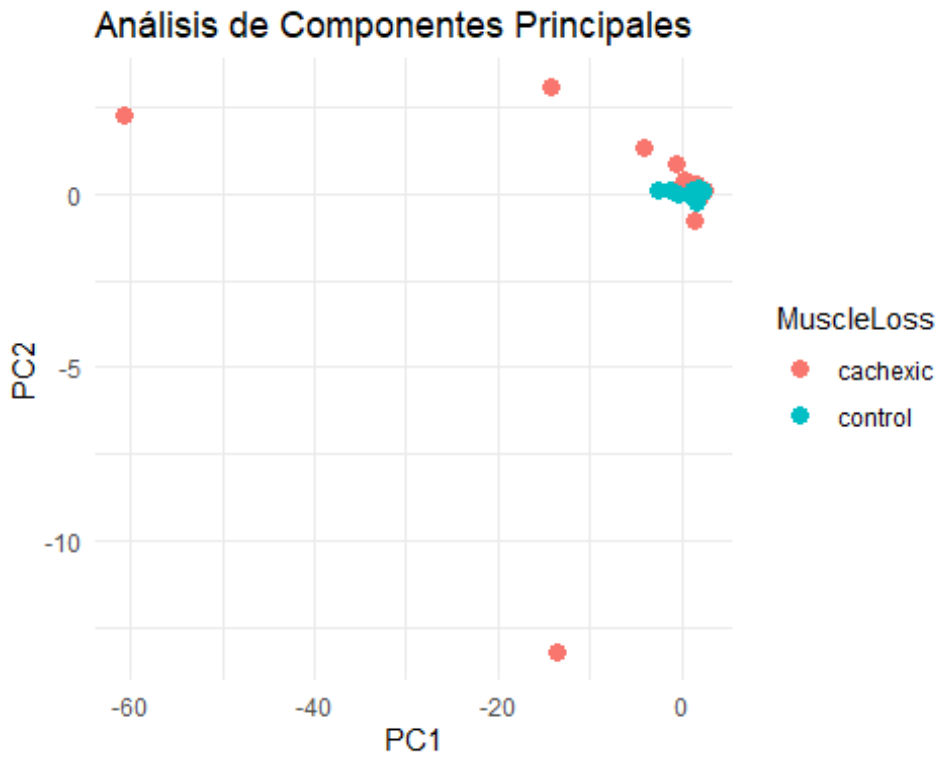
```
##  [1] "Patient.ID"                  "Muscle.loss"
##  [3] "X1.6.Anhydro.beta.D.glucose" "X1.Methylnicotinamide"
##  [5] "X2.Aminobutyrate"            "X2.Hydroxyisobutyrate"
##  [7] "X2.Oxoglutarate"             "X3.Aminoisobutyrate"
##  [9] "X3.Hydroxybutyrate"          "X3.Hydroxyisovalerate"
## [11] "X3.Indoxylsulfate"           "X4.Hydroxyphenylacetate"
## [13] "Acetate"                     "Acetone"
## [15] "Adipate"                     "Alanine"
## [17] "Asparagine"                  "Betaine"
## [19] "Carnitine"                   "Citrate"
## [21] "Creatine"                    "Creatinine"
## [23] "Dimethylamine"               "Ethanolamine"
## [25] "Formate"                     "Fucose"
## [27] "Fumarate"                    "Glucose"
## [29] "Glutamine"                   "Glycine"
## [31] "Glycolate"                   "Guanidoacetate"
## [33] "Hippurate"                   "Histidine"
## [35] "Hypoxanthine"                "Isoleucine"
## [37] "Lactate"                     "Leucine"
## [39] "Lysine"                      "Methylamine"
## [41] "Methylguanidine"             "N.N.Dimethylglycine"
## [43] "O.Acetylcarnitine"           "Pantothenate"
## [45] "Pyroglutamate"               "Pyruvate"
## [47] "Quinolinate"                 "Serine"
## [49] "Succinate"                   "Sucrose"
## [51] "Tartrate"                    "Taurine"
## [53] "Threonine"                   "Trigonelline"
## [55] "Trimethylamine.N.oxide"      "Tryptophan"
## [57] "Tyrosine"                    "Uracil"
## [59] "Valine"                      "Xylose"
## [61] "cis.Aconitate"               "myo.Inositol"
## [63] "trans.Aconitate"             "pi.Methylhistidine"
## [65] "tau.Methylhistidine"

# Vamos a crear el objeto assayData
# Primero excluimos las dos primeras columnas ('Patient.ID' y 'Muscle.loss'
) y lo convertimos a matriz
assayData <- as.matrix(data[, -c(1, 2)])

# Verificamos el número de columnas en assayData
num_samples <- ncol(assayData)
cat("Número de muestras (columnas) en assayData:", num_samples, "\n")

## Número de muestras (columnas) en assayData: 63

# Creamos colData solo con las muestras
colData <- data.frame(
  PatientID = as.character(data[["Patient.ID"]]),  # Convertimos a carácter
  MuscleLoss = as.factor(data[["Muscle.loss"]])   # Convertimos a factor
)

# Comprobamos que el número de filas en colData coincida con num_samples
if (nrow(colData) != nrow(data)) {
  stop("El número de filas en colData debe coincidir con el número de muest
ras en assayData.")
```

```
}

# Reducimos colData para que coincida con el número de muestras en assayDat
a
colData <- colData[1:num_samples, ]  # Seleccionar solo las filas correspon
dientes a las muestras

# Verificamos las dimensiones
cat("Dimensiones de assayData:", dim(assayData), "\n")  # Debe ser [número
de metabolitos] x [número de muestras]

## Dimensiones de assayData: 77 63

cat("Dimensiones de colData:", dim(colData), "\n")  # Debe ser [número de m
uestras] x 2

## Dimensiones de colData: 63 2

# Y ya podemos crear el objeto SummarizedExperiment
se <- SummarizedExperiment(assays = list(counts = assayData), colData = col
Data)

# Guardamos el objeto en formato binario
save(se, file = "human_cachexia.Rda")

# Resumen del objeto SummarizedExperiment
summary(se)

## [1] "SummarizedExperiment object of length 77 with 0 metadata columns"

# Obtenemos también un resumen estadístico de los metabolitos
rowMeans(assay(se), na.rm = TRUE)  # Promedio de cada metabolito

##  [1]  699.85571  708.30238  771.79444 1021.28206  441.21968  537.47508
##  [7]  400.84921   82.76952  207.80190  478.06825  367.51810  650.75175
## [13]  484.69984  355.16476   53.48016   56.66952  318.71127  424.13524
## [19]  356.54508  461.15635  460.74619  645.12460  546.22698  153.92429
## [25]  183.79032  350.55016 1237.54032  516.60905   62.81317  738.88921
## [31]  199.60651  376.69540  227.96540  327.87587  191.82460  148.50810
## [37]  496.28603  581.78587  270.28206  198.65333  502.98032  697.46762
## [43]  279.24349  579.71683  745.91127  525.01794  143.27984   72.36413
## [49]  639.13111   76.81286   71.89698  170.47540  396.23556  343.36556
## [55]   64.50270   54.02413  289.17048  347.33032  361.04302  137.42016
## [61]  357.11778   42.79619  316.90905  159.56984   63.52333  240.74381
## [67]  467.35238   97.23984  511.55175  110.01683  118.80492   56.98556
## [73]  342.26270  142.84159  147.54762  159.46317  121.69937

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.3.3

# Convertimos a formato largo para ggplot
assay_long <- as.data.frame(t(assay(se)))
assay_long$MuscleLoss <- colData(se)$MuscleLoss

# Convertimos a formato largo utilizando reshape2
```

```r
library(reshape2)
assay_long <- melt(assay_long, id.vars = "MuscleLoss", variable.name = "Met
abolite", value.name = "Value")

# Creamos boxplots para los metabolitos
ggplot(assay_long, aes(x = MuscleLoss, y = Value, fill = MuscleLoss)) +
  geom_boxplot() +
  facet_wrap(~Metabolite, scales = "free") +
  labs(title = "Boxplots de Metabolitos por Grupo",
       x = "Grupo",
       y = "Valor") +
  theme_minimal()
```



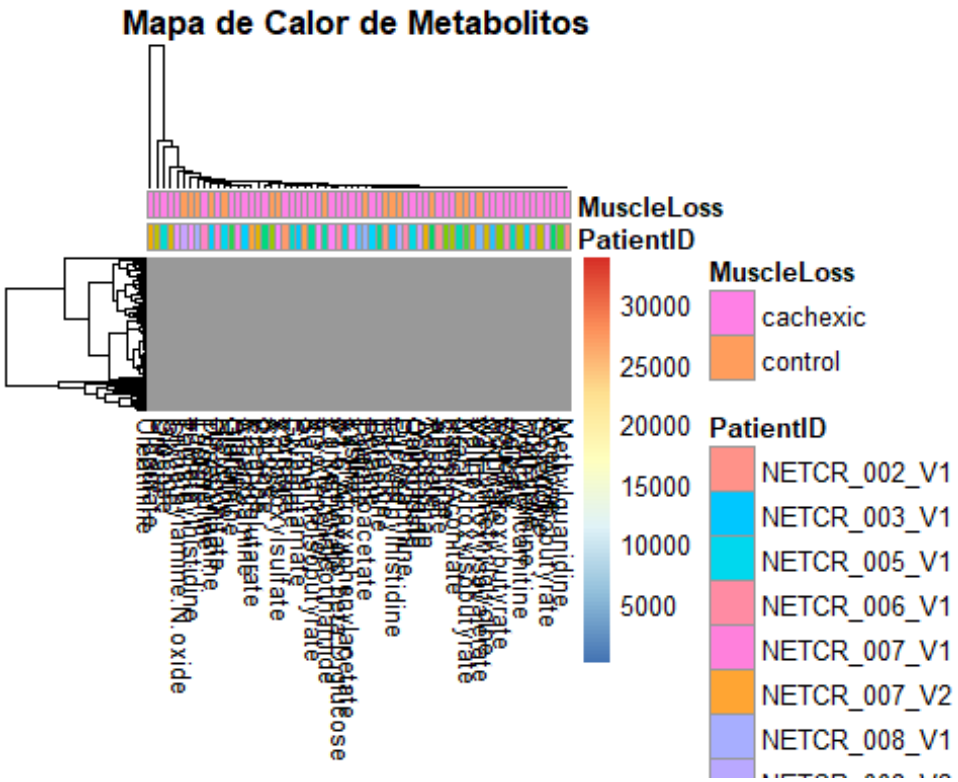

```r
# Realizamos un PCA
pca <- prcomp(t(assay(se)), scale. = TRUE)  # Transponemos para que las mue
stras estén en filas
pca_df <- data.frame(pca$x)
pca_df$MuscleLoss <- colData(se)$MuscleLoss

# Gráfica de PCA
ggplot(pca_df, aes(x = PC1, y = PC2, color = MuscleLoss)) +
  geom_point(size = 3) +
  labs(title = "Análisis de Componentes Principales",
       x = "PC1",
       y = "PC2") +
  theme_minimal()
```




```
library(pheatmap)
```

```
## Warning: package 'pheatmap' was built under R version 4.3.3
```

```
# Creamos un mapa de calor
pheatmap(assay(se),
         cluster_rows = TRUE,
         cluster_cols = TRUE,
         show_rownames = TRUE,
         show_colnames = TRUE,
         annotation_col = as.data.frame(colData(se)),
         main = "Mapa de Calor de Metabolitos")
```




```r
# Convertimos los datos de assay a enteros
assay(se) <- round(assay(se))

# Verificamos que los datos son ahora enteros
is_integer <- all(assay(se) == floor(assay(se)))
if (!is_integer) {
  stop("Algunos datos no son enteros, revise la conversión.")
}

# Cargamos la librería
library(DESeq2)

## Warning: package 'DESeq2' was built under R version 4.3.3

# Ajustamos el nivel de referencia de MuscleLoss
se$MuscleLoss <- relevel(se$MuscleLoss, ref = "control")  # Cambia "control
" por el nivel que desees usar como referencia

# Creamos el objeto DESeqDataSet
dds <- DESeqDataSet(se, design = ~ MuscleLoss)

## converting counts to integer mode

# Continuamos con el análisis diferencial
dds <- DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship
```

```
## -- note: fitType='parametric', but the dispersion trend was not well cap
tured by the
##    function: y = a/x + b, and a local regression fit was automatically s
ubstituted.
##    specify fitType='local' or 'mean' to avoid this message next time.

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 12 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```
results <- results(dds)

summary (results)
```

```
##
## out of 77 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 1, 1.3%
## LFC < 0 (down)     : 4, 5.2%
## outliers [1]       : 0, 0%
## low counts [2]     : 0, 0%
## (mean count < 14)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
# Filtramos los metabolitos que tienen un LFC positivo (más altos en cachex
ic que en control)
upregulated <- results[results$log2FoldChange > 0 & results$padj < 0.1, ]
# Filtramos los metabolitos que tienen un LFC negativo (más bajos en cachex
ic que en control)
downregulated <- results[results$log2FoldChange < 0 & results$padj < 0.1, ]

# Mostrar los metabolitos que están regulados al alza
print("Metabolitos regulados al alza:")
```

```
## [1] "Metabolitos regulados al alza:"
```

```
print(upregulated)
```

```
## log2 fold change (MLE): MuscleLoss cachexic vs control
## Wald test p-value: MuscleLoss cachexic vs control
## DataFrame with 1 row and 6 columns
##    baseMean log2FoldChange     lfcSE      stat     pvalue      padj
##   <numeric>      <numeric> <numeric> <numeric>  <numeric> <numeric>
## 1   31.8921       0.957848  0.325623   2.94159 0.00326537 0.0628583
```

```
# Mostrar los metabolitos que están regulados a la baja
print("Metabolitos regulados a la baja:")
```

```
## [1] "Metabolitos regulados a la baja:"
```

```
print(downregulated)
```

```
## log2 fold change (MLE): MuscleLoss cachexic vs control
## Wald test p-value: MuscleLoss cachexic vs control
## DataFrame with 4 rows and 6 columns
##    baseMean log2FoldChange     lfcSE      stat      pvalue       padj
##   <numeric>      <numeric> <numeric> <numeric>   <numeric>  <numeric>
## 1  135.2014      -0.626501  0.211814  -2.95779 3.09850e-03 0.06285829
## 2   84.5435      -0.938628  0.233301  -4.02324 5.74018e-05 0.00441994
## 3  109.1034      -0.781994  0.248635  -3.14515 1.66000e-03 0.06285829
## 4   23.0317      -0.833933  0.291503  -2.86080 4.22573e-03 0.06507621
```