Análisis de datos ómicos - PEC1

Jara Ballestin

2024-11-04

Descarga de los datos

El primer paso para comenzar la actividad es la descarga de un dataset de metabolómica del repositorio de github proporcionado (https://github.com/nutrimetabolomics/metaboData). Para ello, hacemos click en el botón verde "Code" y copiamos la URL. Para clonar el repositorio localmente, abrimos el símbolo del sistema y una vez seleccionada la carpeta de destino, escribimos: "git clone https://github.com/nutrimetabolomics/metaboData.git". Yo he elegido el primer dataset en esta carpeta: 2018-MetabotypingPaper. Los datos consisten de tres archivos:

- DataInfo_S013.csv: Metadata. Información de cada columna en el archivo "DataValues_S013.csv".
- DataValues_S013.csv: Valores clínicos y metabolómicos de 39 pacientes en 5 momentos diferentes.
- AAInformation_S006.htm*: Información adicional de los metabolitos en el archivo "DataValues_S013.csv".

*Este archivo debería ser un .csv pero en este repositorio está como .htm y no puedo abrirlo. Por lo tanto, he accedido al repositorio de github original (https://github.com/nutrimetabolomics/Metabotyping2018) y lo he clonado para tener el archivo original en mi ordenador. Al abrir el archivo, he visto que los nombres de los metabolitos no coinciden con los del archivo "DataValues_S013.csv". He buscado el paper original "Metabotypes of response to bariatric surgery independent of the magnitude of weight loss", y este archivo corresponde con la tabla suplementaria 1 (S1 Table): este archivo contiene información sobre varios metabolitos, incluyendo algunos que fueron excluidos del análisis. Además, algunos metabolitos están en este archivo pero no en "DataValues_S013.csv", o viceversa.

Crear un contenedor SummarizedExperiment

Cargamos los paquetes necesarios:

```
if (!requireNamespace("SummarizedExperiment", quietly = TRUE)) {
  install.packages("BiocManager")
  BiocManager::install("SummarizedExperiment")
}
library(SummarizedExperiment)
```

Cargamos los archivos de datos:

```
data_values <- read.csv("data/DataValues_S013.csv", row.names = 1)
data_info <- read.csv("data/DataInfo_S013.csv", stringsAsFactors = FALSE)
aa_info <- read.csv("data/AAInformation_S006.csv", sep = ';', row.names = 1)
```

Para poder guardar la información de aa_info en nuestro SummarizedExperiment, generamos un dataframe con la información de aa_info que corresponda con los metabolitos de data_values.

```
# 1. Crear un dataframe vacío aa_info_new
aa_info_new <- data.frame(matrix(ncol = ncol(aa_info) + 1, nrow = 0))
# 2. Iterar por cada nombre de las columnas en data_values y añadir a aa_info_new la fila
#correspondiente de aa_info
for (col_name in colnames(data_values)) {
  #Seleccionar sólo números y letras, excluir los dos caracteres referente al tiempo
  name_values <- gsub("[^[:alnum:]]", "", substr(col_name, 1, nchar(col_name) - 2))
  # 3. Buscar el nombre extraído en la columna "Metabolite.abbreviation" de aa_info y
  #añadir la fila a aa_info_new
  matched<- FALSE
  for (j in 1:nrow(aa_info)) {
    #Seleccionar sólo números y letras
    name_aa <- gsub("[^[:alnum:]]", "", aa_info$Metabolite.abbreviation[j])
    if (name_values == name_aa) {
      aa_info_new <- rbind(aa_info_new, c(col_name, as.character(unname(aa_info[1, ]))))
      matched <- TRUE
      break
    }
  }
  if (!matched) {
    aa_info_new <- rbind(aa_info_new, c(col_name, rep(NA, ncol(aa_info))))
  }
}
colnames(aa_info_new) <- c("name.DataValues", colnames(aa_info))

#Comprobamos dimensiones:
ncol(data_values)==nrow(data_info)
```

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

```
ncol(data_values)==nrow(aa_info_new)
```

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

```
#Comprobamos que los nombres coinciden
all(colnames(data_values) == data_info[, 1])
```

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

```
all(colnames(data_values) == aa_info_new[, 1])
```

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

```
#Unimos ambas dataframes _info:
data_info_new <- cbind(data_info, aa_info_new[,-1])
head(data_info_new)
```

```
##          X  VarName    varTpe Description Class Metabolite.abbreviation
## 1 SUBJECTS SUBJECTS   integer    dataDesc  <NA>                     <NA>
## 2  SURGERY  SURGERY character    dataDesc  <NA>                     <NA>
## 3      AGE      AGE   integer    dataDesc  <NA>                     <NA>
## 4   GENDER   GENDER character    dataDesc  <NA>                     <NA>
## 5    Group    Group   integer    dataDesc  <NA>                     <NA>
## 6 MEDDM_T0 MEDDM_T0   integer    dataDesc  <NA>                     <NA>
##   Metabolite Platform Data.type    X
## 1       <NA>     <NA>      <NA> <NA>
## 2       <NA>     <NA>      <NA> <NA>
## 3       <NA>     <NA>      <NA> <NA>
## 4       <NA>     <NA>      <NA> <NA>
## 5       <NA>     <NA>      <NA> <NA>
## 6       <NA>     <NA>      <NA> <NA>
```

Creamos el objeto SummarizedExperiment con los datos:

```
se <- SummarizedExperiment(
    assays = list(counts = data_values),
    colData = data_info_new
)
#Observamos un resumen del objeto SummarizedExperiment:
se
```

```
## class: SummarizedExperiment
## dim: 39 695
## metadata(0):
## assays(1): counts
## rownames(39): 1 2 ... 38 39
## rowData names(0):
## colnames(695): SUBJECTS SURGERY ... SM.C24.0_T5 SM.C24.1_T5
## colData names(10): X VarName ... Data.type X
```

```
#Guardamos el objeto contenedor en formato binario (.Rda)
save(se, file = "SummarizedExperiment.Rda")
#Guardamos los datos en formato texto
write.table(assay(se), file = "data/datos.txt", sep = "\t")
```

Exploración del dataset

Este conjunto de datos recopila información clínica y metabolómica de 39 pacientes en cuatro momentos diferentes. Incluye datos demográficos y de salud, así como niveles de metabolitos específicos.

Comenzamos observando el número de variables y muestras:

```
#Número de variables
num_variables <- ncol(assay(se))
cat("Número de variables (variables demográficas y metabolitos):", num_variables, "\n")
```

```
## Número de variables (variables demográficas y metabolitos): 695
```

```r
#Número de muestras
num_muestras <- nrow(assay(se))
cat("Número de muestras:", num_muestras, "\n")
```

```
## Número de muestras: 39
```

Observamos que hay 695 variables (clínicas y metabolómicas), y 39 muestras (39 pacientes, como indica la descripción).

Comprobamos si faltan valores (NA):

```r
na <- colSums(is.na(assay(se)))
cat("Datos faltantes:", count(na))
```

```
## Datos faltantes: 157
```

Observamos que faltan bastantes valores en el dataset, lo que puede ser problemático más adelante en el análisis de los datos.

A continuación, realizamos una descripción estadística de las variables demográficas:

```r
#Resumen cirugía:
print(table(data_values$SURGERY))
```

```
##
## by pass tubular
##      26      13
```

```r
#Resumen edad:
print(summary(data_values$AGE))
```

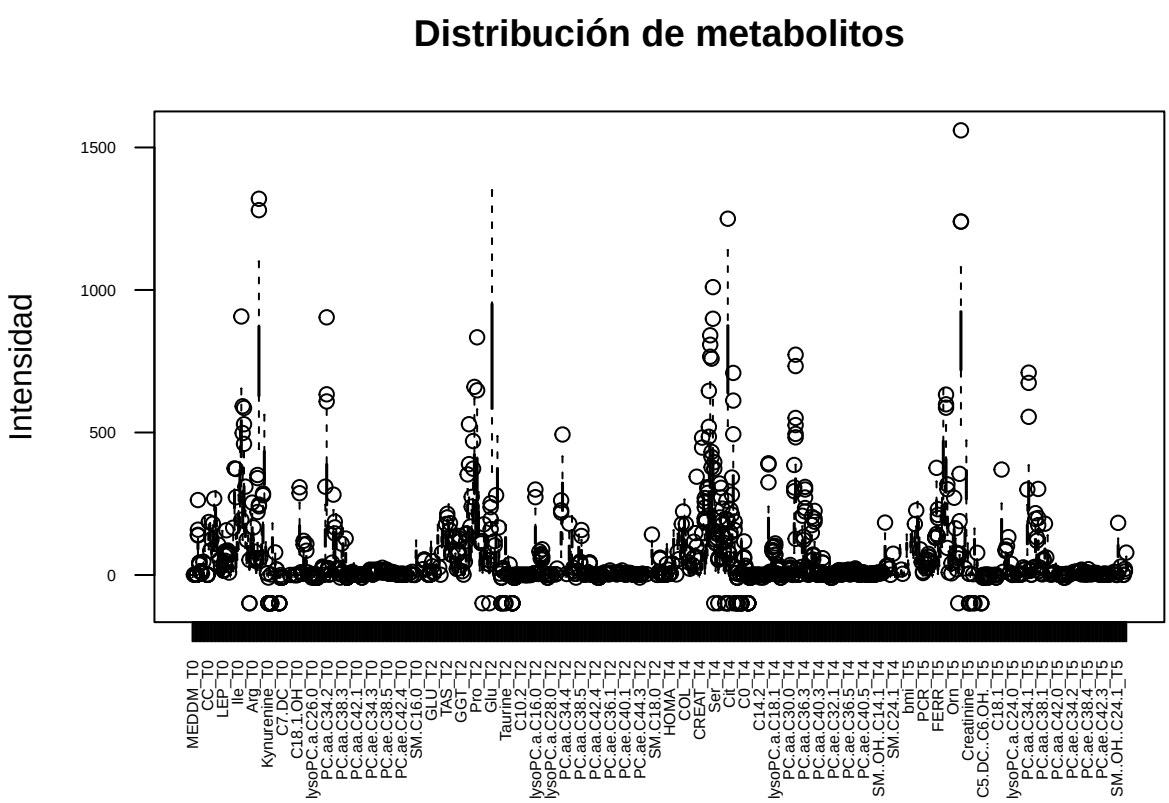
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.00   35.00   41.00   40.79   46.00   59.00
```

```r
#Resumen sexo:
print(table(data_values$GENDER))
```

```
##
##  F  M
## 27 12
```

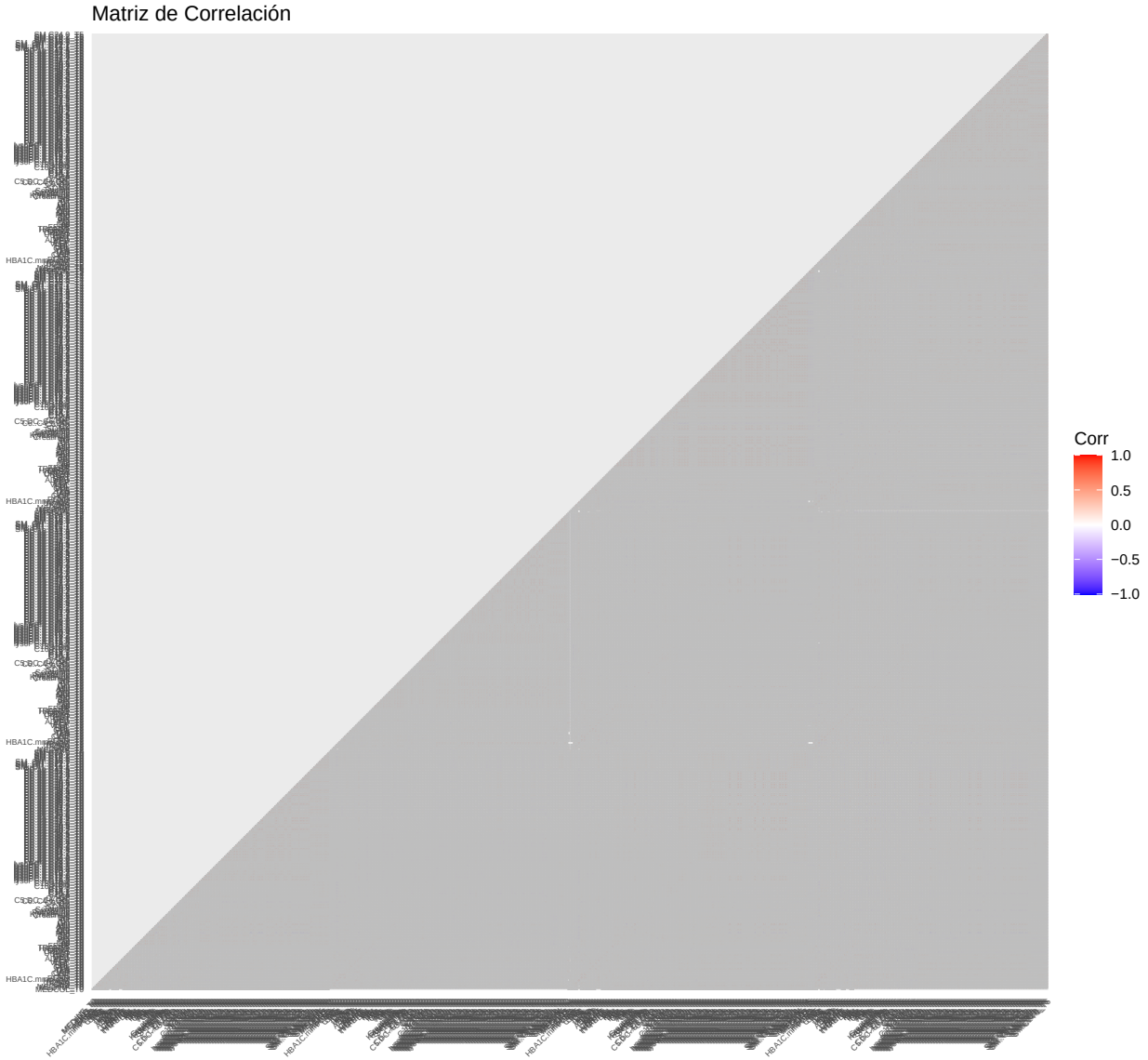
A continuación, realizaremos un gráfico simple para visualizar la distribución de los metabolitos:

```r
# Seleccionar sólo los datos de metabolitos
metabolitos <- assay(se)[, !(colnames(assay(se)) %in% c("SUBJECTS", "SURGERY", "AGE",
      "GENDER", "Group"))]

# Crear un boxplot para todos los metabolitos
boxplot(metabolitos, main = "Distribución de metabolitos",
        ylab = "Intensidad", las = 2, cex.axis = 0.5)
```



Probamos a generar una matriz de correlación para comprobar si se pueden sacar conclusiones:

```
#Calculamos la matriz de correlación entre las variables
correlation_matrix <- cor(metabolitos, use = "pairwise.complete.obs")

#Visualizamos la matriz de correlación
library(ggcorrplot)
ggcorrplot(correlation_matrix,
           type = "lower",
           title = "Matriz de Correlación") +
  theme(axis.text.x = element_text(size = 5),
        axis.text.y = element_text(size = 5))
```



En este caso, visualizar la matriz de correlación así no nos aporta nada, ya que es una gran cantidad de datos.

Ya que el numero de metabolitos analizados es muy grande, sería útil hacer un PCA para reducir la dimensionalidad y entender mejor los datos. También se podrían ver patrones o tendencias en los datos, o agrupamientos entre muestras. Así, podríamos identificar metabolitos relevantes, así como outliers. Yo he tenido muchos problemas con esto, especialmente con los valores NA. Hay demasiados como para eliminar esos datos, por lo que sería conveniente imputarlos, pero me da errores cuando intento hacerlo de varias maneras diferentes.

Repositorio Github

https://github.com/jarabb/Ballestin-Ballestin-Jara-PEC1.git