Sanchez-Sierra-David-PEC1

David Sánchez Sierra

2025-04-01

Análisis de datos ómicos (M0-157). Prueba de evaluación continua.

Índice

1. Resumen

2. Objetivos

3. Enfoque y método seguido

3.1. Selección y descarga del dataset

Para esta actividad, se seleccionó el dataset "caquexia" obtenido del repositorio de github adjunto al encunciado. Este conjunto de datos ha sido ampliamente utilizado en diversos tutoriales de MetaboAnalyst, lo que permitió expandir el análisis realizado al disponer de información sobre numerosos enfoques y metodologías establecidas.

Las técnicas metabolómicas se han utilizado para estudiar los cambios metabólicos, incluyendo las variaciones en las concentraciones de metabolitos y las vías metabólicas alteradas en la progresión de la caquexia relacionada con el cáncer (CC), así como para ampliar la comprensión fundamental de la pérdida muscular(Cui et al. 2022). En este dataset, se incluyen datos sobre 77 pacientes oncológicos, clasificados en dos grupos según su estado de pérdida muscular: controles y caquéxicos.

La descarga de los datos se realizó utilizando R, empleando la función read.csv() para cargar el archivo en formato tabular:

```
dataset <- read.csv("human_cachexia.csv", header = TRUE, sep = ",", stringsAsFactors = FALSE)
str(dataset, max.level = 0)
```

```
## 'data.frame':    77 obs. of  65 variables:
```

Como puede observarse, el dataset contiene información sobre 65 variables, incluyendo las concentraciones de diversos metabolitos en las muestras de orina asociadas con los dos grupos de pacientes. Además, el conjunto de datos incluye el identificador de cada paciente y los metadatos relacionados con las características de estos, en este caso el tipo categórico (control o caquéxicos).

3.2. Creación del objeto SummarizedExperiment

Tras cargar los datos, se creó un objeto de clase SummarizedExperiment proporcionada por el paquete Bioconductor, el cual permite estructurar los datos y metadatos asociados a cada observación, facilitando su manipulación y análisis. Primero se asignaron los valores de Patient.ID como nombres de fila para el dataset, asegurando que cada muestra tuviera una identificación única en la matriz. Tras ello, se eliminó la columna Patient.ID del conjunto de datos, ya que solo se necesitaba para asignar los nombres de las filas, y no debía formar parte de los datos de expresión.

A continuación, se extrajeron las mediciones de los metabolitos del dataset, excluyendo la columna Muscle.loss, que corresponde a la información de los metadatos y no a las concentraciones de los metabolitos. Las concentraciones de metabolitos se almacenaron en una matriz de expresión (expr_matrix), la cual fue transpuesta para que las muestras fueran representadas en las columnas, lo que facilita la asociación con los metadatos. En el siguiente paso, se creó el DataFrame para los metadatos que debía contener la columna Muscle.loss (información sobre el estado de caquexia de las muestras).

El código que implementa este procedimiento es el siguiente:

```
# Cargar paquetes necesarios
library(S4Vectors)
library(SummarizedExperiment)
```

```
# Asignar ID como nombre de fila
rownames(dataset) <- dataset$Patient.ID
dataset <- dataset[, -1] # Eliminar la columna 'Patient ID'

# Asegurarse de que 'Muscle.loss' es excluida correctamente
expr_matrix <- as.matrix(dataset[, -which(names(dataset) == "Muscle.loss")])

# Transponer la matriz para que las muestras sean columnas
expr_matrix <- t(expr_matrix)

# Crear el dataframe de metadatos (solo 'Muscle.loss')
metadata_df <- DataFrame(Muscle.loss = dataset$Muscle.loss)
rownames(metadata_df) <- rownames(dataset)  # Asegurar nombres de fila en metadatos

# Crear el objeto SummarizedExperiment
se_object <- SummarizedExperiment(
  assays = list(counts = expr_matrix),
  colData = metadata_df
```

```
)

# Verificar el objeto creado
se_object
```

```
## class: SummarizedExperiment
## dim: 63 77
## metadata(0):
## assays(1): counts
## rownames(63): X1.6.Anhydro.beta.D.glucose X1.Methylnicotinamide ...
##   pi.Methylhistidine tau.Methylhistidine
## rowData names(0):
## colnames(77): PIF_178 PIF_087 ... NETL_003_V1 NETL_003_V2
## colData names(1): Muscle.loss
```

```
# Guardar el objeto SummarizedExperiment en un archivo .Rda
save(se_object, file = "SE_object.Rda")
```

4. Resultados

4.1. Diferencias entre SummarizedExperiment y ExpressionSet

La clase SummarizedExperiment es una extensión moderna de ExpressionSet, con varias mejoras en cuanto a la estructura de almacenamiento y la interoperabilidad con otras herramientas de Bioconductor. Una de las diferencias más notables es que SummarizedExperiment utiliza el componente assays para manejar las matrices de datos en lugar de la función exprs() que se usaba en ExpressionSet. Además, SummarizedExperiment también incorpora colData y rowData para el almacenamiento de los metadatos relativos a las muestras y las características, respectivamente, sustituyendo a las funciones pData() y fData() que se encontraban en ExpressionSet. Esta modificación simplifica y vuelve más adaptable la gestión de dichos metadatos. Un beneficio adicional significativo de SummarizedExperiment reside en su mejorada interoperabilidad con análisis de datos multi-ómicos, lo que la establece como una herramienta más idónea para la realización de estudios que combinan distintos tipos de datos ómicos, tales como datos genómicos, transcriptómicos y proteómicos ("Convert Seurat Object to Summarised Experiment or ExpressionSet" n.d.).

Estas diferencias hacen que SummarizedExperiment resulte una opción más adecuada para manejar y analizar datos provenientes de diversas fuentes dentro de un mismo entorno de trabajo, otorgándole una mayor flexibilidad en comparación con ExpressionSet.

4.2. Análisis exploratorio

En esta sección se muestra el análisis inicial del objeto SummarizedExperiment (SE), que contiene los datos de expresión de metabolitos y los metadatos asociados a las muestras:

```
dim(se_object)
```

```
## [1] 63 77
```

```
head(assay(se_object)[ , 1:10])
```

```
##                             PIF_178 PIF_087 PIF_090 NETL_005_V1 PIF_115 PIF_110
## X1.6.Anhydro.beta.D.glucose   40.85   62.18  270.43      154.47   22.20  212.72
## X1.Methylnicotinamide         65.37  340.36   64.72       52.98   73.70   31.82
## X2.Aminobutyrate              18.73   24.29   12.18      172.43   15.64   18.36
## X2.Hydroxyisobutyrate         26.05   41.68   65.37       74.44   83.93   80.64
## X2.Oxoglutarate               71.52   67.36   23.81     1199.91   33.12   47.94
## X3.Aminoisobutyrate         1480.30  116.75   14.30      555.57   29.67   17.46
##                             NETL_019_V1 NETCR_014_V1 NETCR_014_V2 PIF_154
## X1.6.Anhydro.beta.D.glucose      151.41        31.50        51.42  117.92
## X1.Methylnicotinamide             36.60         6.82        30.27   52.46
## X2.Aminobutyrate                   8.67         4.18         7.54   19.49
## X2.Hydroxyisobutyrate             42.52        12.94        34.81   72.24
## X2.Oxoglutarate                  223.63        25.03        80.64   73.70
## X3.Aminoisobutyrate               56.26         8.67        17.99   57.97
```

El SE tiene una dimensión de 63 filas y 77 columnas, lo que verifica que contiene 63 metabolitos (características) y 77 pacientes (muestras), confirmando una estructura típica de esta clase de objetos. Los datos de expresión de metabolitos son accesibles mediante la función assay(), tal y como muestra el bloque de código, en el que se visualizan los datos de los primeras 10 pacientes para los 6 primeros metabolitos.

A continuación, se presenta el contenido del objeto colData, que almacena los metadatos de las muestras. En este caso, contiene una sola columna denominada Muscle.loss, que refleja la condición de cada paciente (ya sea cachexic o control):

```
colData(se_object)
```

```
## DataFrame with 77 rows and 1 column
##              Muscle.loss
##              <character>
## PIF_178         cachexic
## PIF_087         cachexic
## PIF_090         cachexic
## NETL_005_V1     cachexic
## PIF_115         cachexic
## ...                  ...
## NETCR_019_V2     control
## NETL_012_V1      control
## NETL_012_V2      control
## NETL_003_V1      control
## NETL_003_V2      control
```

Finalmente, al inspeccionar los primeros nombres de las columnas con colnames(se_object), se observan los identificadores de los pacientes, como PIF_178, PIF_087, PIF_090, entre otros.

```
head(colnames(se_object))
```

```
## [1] "PIF_178"     "PIF_087"     "PIF_090"     "NETL_005_V1" "PIF_115"
## [6] "PIF_110"
```

Este análisis preliminar del SE nos proporciona una visión general de la estructura de los datos y nos permite verificar la correcta organización tanto de las características (metabolitos) como de las muestras (pacientes).

4.3. Anova

```r
# Cargar las librerías necesarias
library(ggplot2)
library(dplyr)
```

```
##
## Adjuntando el paquete: 'dplyr'

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

## The following objects are masked from 'package:GenomicRanges':
##
##     intersect, setdiff, union

## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect

## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union

## The following object is masked from 'package:matrixStats':
##
##     count

## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union

## The following objects are masked from 'package:BiocGenerics':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
dim(assay(se_object))
```

```
## [1] 63 77
```

```r
anova_results <- apply(assay(se_object), 1, function(x) {
  aov_result <- aov(x ~ colData(se_object)$Muscle.loss)
  # Extraemos el valor p sin la estructura de lista
  p_value <- summary(aov_result)[[1]]["Pr(>F)"][1, 1]
  return(p_value)
})

# Crear el DataFrame correctamente
results_df <- data.frame(
  metabolite = rownames(assay(se_object)),
  p_value = anova_results
)

# Verificamos que el DataFrame se crea correctamente
results_df
```

```
##                                           metabolite      p_value
## X1.6.Anhydro.beta.D.glucose X1.6.Anhydro.beta.D.glucose 0.0507913658
## X1.Methylnicotinamide             X1.Methylnicotinamide 0.9343185339
## X2.Aminobutyrate                       X2.Aminobutyrate 0.0274454761
## X2.Hydroxyisobutyrate             X2.Hydroxyisobutyrate 0.0052880207
## X2.Oxoglutarate                         X2.Oxoglutarate 0.2250725245
## X3.Aminoisobutyrate                 X3.Aminoisobutyrate 0.1779554046
## X3.Hydroxybutyrate                   X3.Hydroxybutyrate 0.0011853671
## X3.Hydroxyisovalerate             X3.Hydroxyisovalerate 0.0078274621
## X3.Indoxylsulfate                     X3.Indoxylsulfate 0.0089161341
## X4.Hydroxyphenylacetate         X4.Hydroxyphenylacetate 0.4818107935
## Acetate                                         Acetate 0.0060731953
## Acetone                                         Acetone 0.3723624590
## Adipate                                         Adipate 0.0274331625
## Alanine                                         Alanine 0.0011788609
## Asparagine                                   Asparagine 0.0067852184
## Betaine                                         Betaine 0.0029929659
## Carnitine                                     Carnitine 0.0621334639
## Citrate                                         Citrate 0.0128569793
## Creatine                                       Creatine 0.0526324364
## Creatinine                                   Creatinine 0.0005129808
## Dimethylamine                             Dimethylamine 0.0004460069
## Ethanolamine                              Ethanolamine 0.0265705687
## Formate                                         Formate 0.0174219763
## Fucose                                           Fucose 0.0058534857
## Fumarate                                       Fumarate 0.0590430806
## Glucose                                         Glucose 0.0333063584
## Glutamine                                     Glutamine 0.0010607678
## Glycine                                         Glycine 0.0281395160
## Glycolate                                     Glycolate 0.0563341550
## Guanidoacetate                           Guanidoacetate 0.1378998638
## Hippurate                                     Hippurate 0.0232194018
## Histidine                                     Histidine 0.0109704361
## Hypoxanthine                               Hypoxanthine 0.2555158507
## Isoleucine                                   Isoleucine 0.1326513615
## Lactate                                         Lactate 0.1228434803
## Leucine                                         Leucine 0.0002695046
```

```
## Lysine                                     Lysine 0.2817634974
## Methylamine                           Methylamine 0.0019466201
## Methylguanidine                   Methylguanidine 0.2616448016
## N.N.Dimethylglycine           N.N.Dimethylglycine 0.0001554346
## O.Acetylcarnitine               O.Acetylcarnitine 0.0616844278
## Pantothenate                         Pantothenate 0.5353413018
## Pyroglutamate                       Pyroglutamate 0.0004845363
## Pyruvate                                 Pyruvate 0.0174471086
## Quinolinate                           Quinolinate 0.0001185108
## Serine                                     Serine 0.0037584213
## Succinate                             Succinate 0.0113771704
## Sucrose                                 Sucrose 0.1194324540
## Tartrate                               Tartrate 0.4464672671
## Taurine                                 Taurine 0.0323497353
## Threonine                             Threonine 0.0032881849
## Trigonelline                       Trigonelline 0.0128629342
## Trimethylamine.N.oxide   Trimethylamine.N.oxide 0.0415941735
## Tryptophan                           Tryptophan 0.0018983944
## Tyrosine                               Tyrosine 0.0112887118
## Uracil                                   Uracil 0.5429079298
## Valine                                   Valine 0.0001394238
## Xylose                                   Xylose 0.2154519963
## cis.Aconitate                     cis.Aconitate 0.0038978573
## myo.Inositol                       myo.Inositol 0.0022150347
## trans.Aconitate                 trans.Aconitate 0.0220856691
## pi.Methylhistidine           pi.Methylhistidine 0.1413081041
## tau.Methylhistidine         tau.Methylhistidine 0.0220399755
```

```r
# Filtrar los metabolitos con valor p < 0.05
significant_metabolites <- results_df[results_df$p_value < 0.05, ]

# Ver los metabolitos significativos
head(significant_metabolites)
```

```
##                                  metabolite     p_value
## X2.Aminobutyrate           X2.Aminobutyrate 0.027445476
## X2.Hydroxyisobutyrate X2.Hydroxyisobutyrate 0.005288021
## X3.Hydroxybutyrate       X3.Hydroxybutyrate 0.001185367
## X3.Hydroxyisovalerate X3.Hydroxyisovalerate 0.007827462
## X3.Indoxylsulfate         X3.Indoxylsulfate 0.008916134
## Acetate                             Acetate 0.006073195
```

En el análisis realizado, se identificaron varios metabolitos cuya concentración se asocia de manera significativa con el estado de pérdida muscular (Muscle.loss). Los metabolitos que mostraron una relación estadísticamente significativa ($p < 0.05$) incluyen:

- X2.Aminobutyrate ($p = 0.027$)
- X2.Hydroxyisobutyrate ($p = 0.005$)
- X3.Hydroxybutyrate ($p = 0.001$)
- X3.Hydroxyisovalerate ($p = 0.007$)
- X3.Indoxylsulfate ($p = 0.009$)
- Acetate ($p = 0.006$)

5. Discusión

La caquexia es una enfermedad compleja y multifactorial, caracterizada por la pérdida de peso a través de la pérdida de masa muscular esquelética y tejido adiposo, un desequilibrio en la regulación metabólica y una reducción en la ingesta de alimentos. Las principales causas de este trastorno son factores catabólicos producidos por tumores en la circulación sistémica, así como factores fisiológicos como la activación inflamatoria desequilibrada, la proteólisis, la autofagia y la lipólisis que pueden ocurrir en cánceres gástrico, pancreático, esofágico, pulmonar, hepático y colorrectal ("Caquexia y cáncer - Efectos secundarios" 2024).

Conclusiones

En el contexto oncológico, la caquexia es una de las principales causas de morbilidad y mortalidad, contribuyendo directamente a aproximadamente el 25% de los fallecimientos relacionados con el cáncer. Esta condición no solo deteriora la calidad de vida del paciente, sino que también puede limitar la eficacia de los tratamientos antineoplásicos y aumentar la toxicidad asociada a los mismos.

Las causas de la caquexia en pacientes con cáncer son complejas y multifactoriales. Se cree que los tumores malignos pueden liberar sustancias como el factor de necrosis tumoral alfa (TNF-a), la interleucina-6 (IL-6) y otras citoquinas inflamatorias que inducen la degradación de proteínas y lípidos, llevando a la pérdida de masa muscular y tejido adiposo. Además, la inflamación sistémica y los cambios metabólicos asociados al cáncer pueden provocar un desequilibrio entre la ingesta y el gasto energético, exacerbando la pérdida de peso y la atrofia muscular.

Bibliografía

"Caquexia y cáncer - Efectos secundarios." 2024. {cgvArticle}. https://www.cancer.gov/espanol/cancer/tratamiento/efectos-secundarios/caquexia-cancer.

"Convert Seurat Object to Summarised Experiment or ExpressionSet." n.d. https://support.bioconductor.org/p/9157595/. Accessed April 1, 2025.

Cui, Pengfei, Xiaoyi Li, Caihua Huang, Qinxi Li, and Donghai Lin. 2022. "Metabolomics and Its Applications in Cancer Cachexia." *Frontiers in Molecular Biosciences* 9 (February): 789889. https://doi.org/10.3389/fmolb.2022.789889.