Universitat Oberta de Catalunya
Universitat de Barcelona
MÁSTER UNIVERSITARIO EN BIOINFORMÁTICA Y BIOESTADÍSTICA

Autor: Joan Cabanas Ballbé.
Fecha de entrega de la PEC: abril del 2025.

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

- Este análisis está respaldado por un estudio en profundidad mediante programación en R que encontraréis como apéndice entregado conjuntamente con el informe. En el informe se muestran las fracciones de código más representativas para su comprensión.

Tabla de contenidos.

1. Resumen.

Este informe presenta el análisis exploratorio de un conjunto de datos de metabolómica obtenidos del repositorio Metabolomics Workbench (ID: ST000291). Se descargaron los datos y se construyó un objeto de clase "*SummarizedExperiment"* para organizar los datos y metadatos de manera estructurada. Posteriormente, se realizó un análisis exploratorio de los datos, donde se incluyó estudios estadísticos univaritantes y multivariantes, mediante diagramas de cajas, análisis de componentes principales (PCA) y agrupamiento jerárquico. Los resultados sugieren que existen patrones en los datos asociados a los grupos de estudio, aunque también se observan solapamientos que pueden indicar la presencia de otras fuentes de variabilidad. Finalmente se recomienda realizar mejoras en el análisis para futuros estudios.

2. Objetivos.

El presente informe tiene como finalidad realizar un análisis exploratorio de un conjunto de datos metabolómicos, utilizando herramientas bioinformáticas disponibles en Bioconductor. Los objetivos específicos son:

- Construir un objeto de clase *"SummarixedExperiment"* con los datos y metadatos correctamente organizados.
- Realizar un análisis estadístico univariante para evaluar la distribución y características generales de los datos.
- Aplicar técnicas de análisis multivariante, como PCA y clustering jerárquico, para identificar patrones y relaciones entre las muestras.
- Interpretar los resultados obtenidos y evaluar la calidad del dataset.
- Proponer recomendaciones para mejorar futuros análisis en estudios de metabolómica.

3. Materiales y Métodos.

Los datos con los que se trabajará en este informe, consiste en un *dataset* extraído de *metabolómica* **[ref: 1]**. En este caso para poder obtener los datos se ha procedido a realizar una descarga independiente, utilizando el repositorio proporcionado de GitHub **[ref: 2]**. Desde este repositorio, accedemos a "Datasets" y en el seleccionamos los datos guardados en "2024-fobitools-UseCase_1". En concreto estos datos pertenecen al repositorio de **Metabolomics Workbench**, almacenados con el ID: **ST000291**, **[ref: 1]**.

La descarga de los datos ha consistido en tres ficheros: **features.csv**, con las características de los datos; **metadata.csv**, con todos los metadatos de las muestras; y **metaboliteNames.csv**, con los nombres de los metabolitos.

En este punto se ha procedido a construir un objeto de clase ***SummarizedExperiment***, que se ha denominado "***sumEx***", donde se ha contenido los datos y los metadatos adecuadamente. Una vez organizado el dataset, se ha procedido a realizar un estudio exploratorio siguiendo la plantilla de uno de los casos de estudio realizado anteriormente, "Analisis_de_datos_omicos-Ejemplo_0-Microarrays" **[ref: 3]**, teniendo en cuenta la naturaleza de nuestros datos.

Básicamente dicha exploración consistirá en:- *Análisis univariante de los datos, mediante boxplots y/o histogramas para estudiar la forma general de los mismos. - Análisis multivariante de los datos, mediante Análisis de Componentes Principales y Agrupamiento Jerárquico, para determinar*

si los grupos que aparezcan (en caso de hacerlo) parecen relacionarse con las fuentes de variabilidad del estudio o, si por el contrario, podrían haber otras fuentes de variabilidad como efectos batch.

3.1 Selección de los datos y descarga.

Como se ha comentado los datos seleccionado pertenecen al repositorio de Metabolomics Workbench, con el ID: **ST000291**. Este "dataset" se ha obtenidos del repositorio GitHub proporcionado **[ref: 1]** y se ha decidido trabajar con dichos datos por adaptarse a las condiciones del estudio.

Una vez descargados los ficheros, se ha procedido a cargar los datos como se muestra a continuación:

```
{r}
# Lectura de los tres ficheros.

# Cargar datos de caracteristicas.
features <- read.csv("Ficheros_PEC1/features.csv", row.names = 1, sep = ";")

# Cargar metadatos de las muestras.
metadata <- read.csv("Ficheros_PEC1/metadata.csv", row.names = 1, sep = ";")

# Cargar nombres de metabolitos.
metabolite_names <- read.csv("Ficheros_PEC1/metaboliteNames.csv", row.names = 1, sep = ";"
)
```

Figura 1: Cargamos los tres ficheros mediante la función read.csv.

Para proceder con el estudio hemos analizado los datos y comprobado su organización. Se ha empleado el siguiente código para verificar la concordancia entre los datos.

```
{r}
# Verificamos la concordancia entre los datos, que los nombres de las muestras coincidan.
all(rownames(metadata) %in% colnames(features))  # Debe devolver TRUE
```

```
[1] FALSE
```

Figura 2: Estudio de la concordancia de los datos.

En este caso podemos observar que los nombre de las filas y columnas de los ficheros no coinciden correctamente. A continuación se muestra una revisión más exhaustiva:

```
{r}
# Analisis mas detallado.
# Revisamos los nombres de las columnas de features, deben coincidir con las filas de
metadata.
head(colnames(features))
# Revisamos los nombres de las filas de metadata, deben coincidir con las columnas de
features.
head(rownames(metadata))
# Revisamos los nombres de las filas de features, deben coincidir con las filas de
metabolite_names.
head(rownames(features))
# Revisamos los nombres de las filas de metabolite_names, deben coincidir con las filas de
features.
head(rownames(metabolite_names))
```

```
[1] "b1"  "b10" "b11" "b12" "b13" "b14"
[1] "1" "2" "3" "4" "5" "6"
[1] "443489"   "107754"   "9543071"  "11011465" "5281160"  "440341"
[1] "1" "2" "3" "4" "5" "6"
```

Figura 3: Revisión de la concordancia.

Una vez detectado el error, procedemos a asegurar que los nombres coincidan, mediante la renombración de las filas en base a las columnas de “features”. A continuación se muestra el código empleado:

```r
{r}
# Renombramos el nombre de las filas para conseguir que coincidan con las columnas de
"features.csv".
rownames(metadata) <- colnames(features)
rownames(metabolite_names) <- rownames(features)
```

Figura 4: Corrigiendo la concordancia.

3.2 Estructura de los datos.

Los datos descargados constan de tres archivos separados: **features.csv**, **metadata.csv** y **metaboliteNames.csv**.

- El primer archivo, con las características (features) consta de 1541 variables y 45 muestras;
- El segundo archivo, los metadatos (metadata) con 45 filas y dos columnas (nombre de la muestra y nombre del grupo.
- El tercer archivo, los nombres de los metadatos (metaboliteNames), con 1541 filas y 3 columnas (nombre original, ID de PubChem e ID de KEGG).

3.3 Creamos el objeto SummarizedExperiment.

Una vez contamos con todos los datos y con los ajustes realizados se procede a crear el objeto “*SummarizedExperiment*”, en nuestro caso será “**sumEx**”. A continuación mostramos el código:

```r
{r}
sumEx<-SummarizedExperiment(assays = list(counts = as.matrix(features)),
                            colData = metadata,
                            rowData = metabolite_names)
```

Figura 5: Construcción del objeto SummarizedExperiment.

Realizamos un primer análisis inicial:

```r
{r}
# Mostramos el resumen del objeto.
sumEx
# Visualizamos los metadatos de muestras.
colData(sumEx)
# Visualizamos los metadatos de metabolitos.
rowData(sumEx)
# Mostramos la matriz de expresion.
assay(sumEx)
```

```
class: SummarizedExperiment
dim: 1541 45
metadata(0):
assays(1): counts
rownames(1541): 443489 107754 ... 53297445 11954209
rowData names(3): names PubChem KEGG
colnames(45): b1 b10 ... c8 c9
colData names(2): ID Treatment
DataFrame with 45 rows and 2 columns
             ID   Treatment
    <character> <character>
b1           b1    Baseline
b10         b10    Baseline
b11         b11    Baseline
b12         b12    Baseline
b13         b13    Baseline
...         ...         ...
```

Figura 6: Análisis inicial.

Las principales diferencias entre "***SummarizedExperiment***" y "***ExpressionSet***" radican en su diseño y flexibilidad para el análisis de datos ómicos, a continuación se explican las diferencias claves:

Sin nos fijamos "*SummarizedExperiment*", permite manejar múltiples matrices de datos (diferentes niveles de expresión o normalización mediante la lista "assays", usa "DataFrame" de Bioconductor para "rowData" y "colData" lo que ofrece mayor flexibilidad en la anotación de genes y muestras. En resumen es más fácil su integración con análisis avanzados en Bioconductor.

Por otro lado, "*ExpressionSet*", solo admite una única matriz de expresión (exprs), y maneja los metadatos con "AnnotatedDataFrame", que tiene una estructura más rígida en comparación con DataFrame de Bioconductor.

Es importante añadir que "*SummarizedExperiment*" permite almacenar más de una matriz de datos de expresión en "assays", como datos crudos, normalizados o transformados, ampliando las posibilidades de comparar distintos niveles de procesamiento en el mismo objeto.

A continuación se muestra una tabla resumen de las principales diferencias:

Característica	**SummarizedExperiment**	**ExpresionSet**
Tipo de datos	RNA-seq, ChIP-seq, proteómica.	Microarrays
Múltiples matrices de datos	Assays	Unicamente "exprs"
Metadatos de genes y muestras	RowData, colData (es más flexible)	FeatureData, phenoData (más rígido)
Soporte para coordenadas genómicas	RowRanges con Granges	No los soporta
Uso recomendado en Bioconductor	Actualizado	Obsoleto para nuevos análisis

En la siguiente referencia se ha extraído información sobre el objeto de datos "*ExpressionSet*" y el objeto "*SummarizedExperiment*", **[ref: 4]**.

3.4 Análisis exploratorio de los datos.

3.4.1 Análisis estadístico univariado.

Una vez extraídos los datos y la información procedemos a realizar una exploración básica, explorando la distribución de los valores de expresión en el "SummarizedExperiment". Podemos analizar la dimensión del conjunto de datos y realizar un resumen de las estadísticas descriptivas de las muestras.

En este caso, los datos ómicos son de alto rendimiento, por lo que resulta complejo tener una buena visión general simplemente "inspeccionado los datos", por lo que una mejor opción, es generar un diagrama de cajas que permita ver todas las muestras a la vez y proporcionar pistas sobre la conveniencia de realizar algún tipo de preprocesamiento.

Primero se procede a realizar un Boxplot (diagrama de cajas) para observar todas las muestras a la vez y poder tener una visión global de los valores de expresión. A continuación se muestra el código empleado:

```r
{r}
# Boxplot de los valores de expresion.
boxplot(assay(sumEx), las=2, col="lightblue",
        main="Distribución de los valores de expresión",
        xlab="Muestras", ylab="Expresión")
```

Figura 7: Código para genera el primer Boxplot.

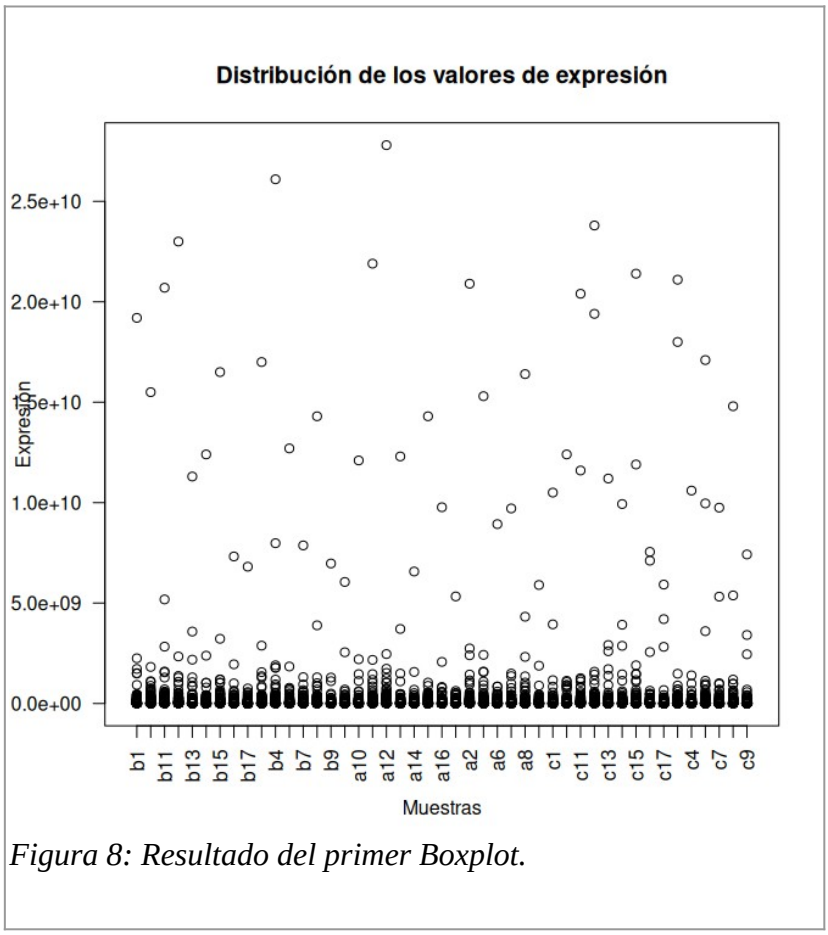

Figura 8: Resultado del primer Boxplot.

Se observa que los datos son claramente asimétricos, lo que sugiere que puede tener sentido trabajar con los mismos datos en escala logarítmica. A continuación se realiza la transformación logarítmica y el segundo Boxplot:

```r
{r}
# Realizamos una transformacion logaritmica para mejorar la simetria.
logExpr <- log2(assay(sumEx) + 1) # Sumamos 1 para evitar log(0).

# Boxplot despues de la transformacion logaritmica.
boxplot(logExpr, las=2, col="khaki1",
        main="Distribución de los valores log2 de expresión",
        xlab="Muestras", ylab="log2(Expresión)")
```

Figura 9: Código para generar el segundo Boxplot.

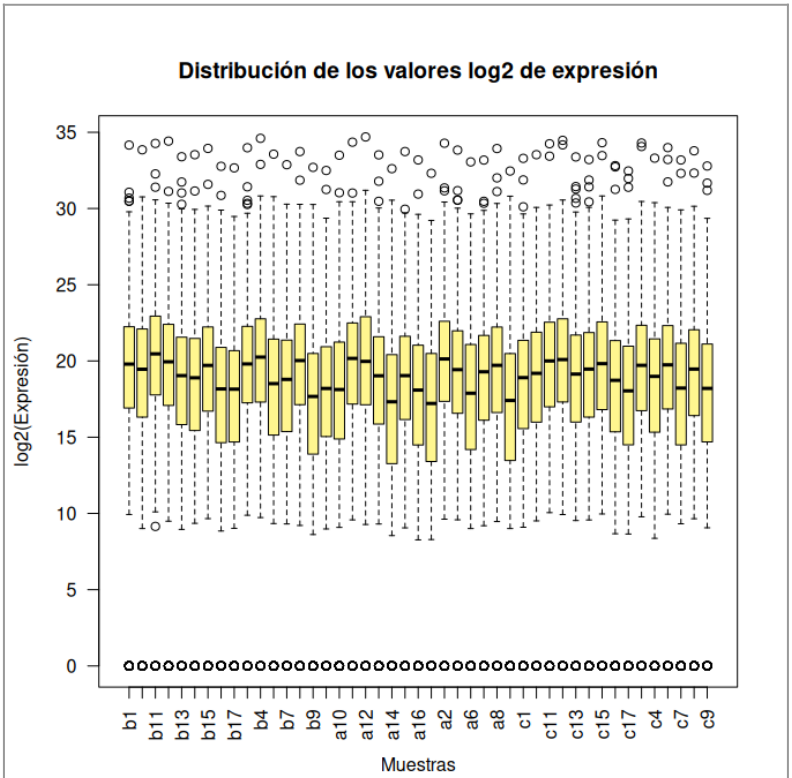

Figura 10: Resultado del Boxplot con los datos transformados.

Claramente, podemos concluir, a la vista del segundo gráfico, que es mejor trabajar con los datos transformados logarítmicamente, ya que la transformación ayuda a estabilizar la varianza.

3.4.2 Análisis Multivariante.

Para identificar patrones en los datos, realizamos un **Análisis de Componentes Principales (PCA)** y una **Agrupación Jerárquica**.

Un análisis en componentes principales puede facilitar la visualización de los datos en dimensión reducida y, sobretodo, detectar posibles patrones que no se detecten a simple vista.

El PCA transforma las variables originales de forma que las nuevas componentes (las variables transformadas) resultan tener dos propiedades muy interesantes: Son independientes entre ellas, y cada componente explica un porcentaje de variabilidad mayor que la anterior, con lo que suele bastar con las dos o tres primeras componentes para obtener una visualización de los datos en dimensión reducida.

En primer lugar, realizar el cálculo de las componentes principales. Para realizar este paso previamente ha sido necesario eliminar los valores problemáticos (en el apéndice se muestra en detalle).

A continuación se podrá proceder a gráficar los primeros componentes principales, en este caso 3.

```
{r}
# Calculamos PCA con datos transformados.
pcaRes <- prcomp(t(logExprClean), scale=TRUE)
# Mostramos la proporcion de varianza explicada.
summary(pcaRes)

# Realizamos la grafica del PCA.
autoplot(pcaRes, data = as.data.frame(colData(sumEx)),
         colour = "Treatment", label = TRUE, label.size = 3) +
  ggtitle("Análisis de Componentes Principales (PCA)") +
  theme_minimal()
```

Figura 11: Código para calcular los componentes principales y realizar su gráfica.

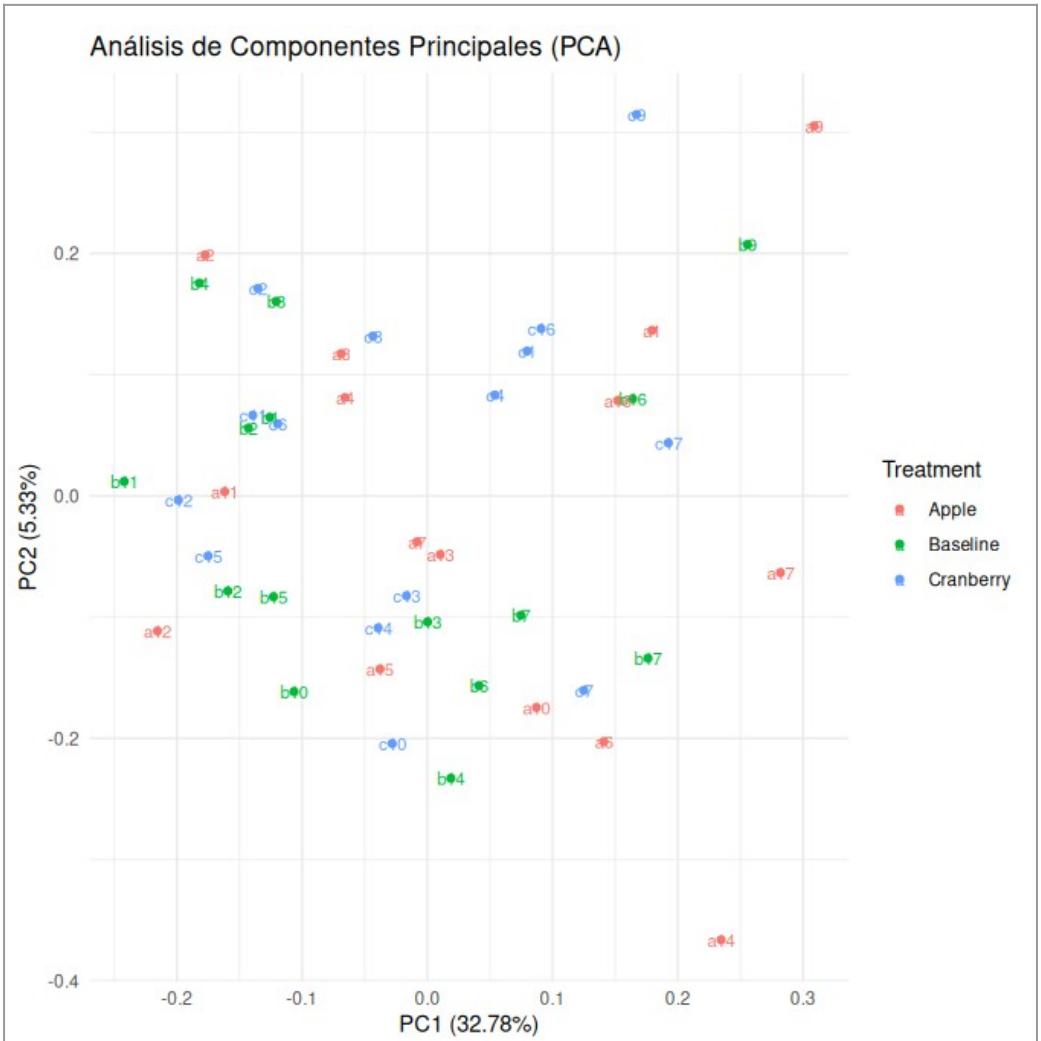

Figura 12: Gráfica PCA.

Alternativamente, se puede utilizar una agrupación jerárquica para visualizar cualquier agrupación esperada (o descubrir una inesperada) de las muestras.

A continuación se muestra el código utilizado:

```r
{r}
# Realizamos una matriz de distancias y clustering jerarquico.
distMat <- dist(t(logExpr))
hc <- hclust(distMat, method="average")

# Generamos el dendrograma.
plot(hc, main="Clustering jerárquico de muestras", hang=-1)
```

Figura 13: Código para realizar la agrupación jerárquica.

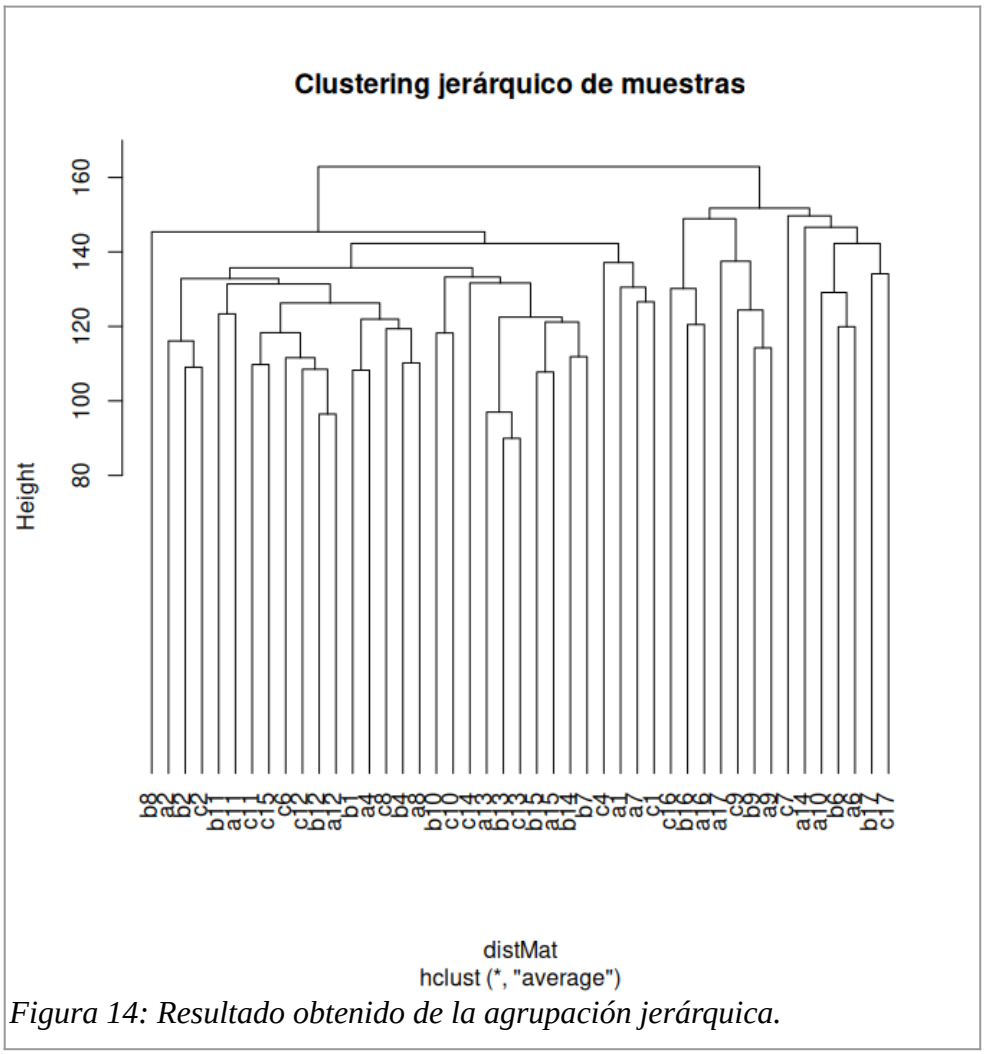

Figura 14: Resultado obtenido de la agrupación jerárquica.

4. Interpretación de los resultados.

Al observar los resultados obtenidos en el **Análisis de Componentes Principales (PCA)**, **figura 12**, nos permite visualizar la variabilidad en los datos de expresión. Podemos observar tres grupos de muestras, coloreadas según su tratamiento: **Rojo** (apple), **Verde** (baseline), y en **Azul** (Cranberry).

Las dos primeras componentes principales (PC1 y PC2) explican conjuntamente 38.11% de la variabilidad total de los datos (PC1: 32.78% y PC2: 5.33%).

Se puede observar una cierta diferencia entre los grupos aunque con bastante solapamiento. Este solapamiento entre grupos puede indicar que existen otras fuentes de variabilidad (por ejemplo, efectos batch o ruido técnico).

En el **Agrupamiento Jerárquico (Cluster Dendrogram)**, **figura 14**, representa la similitud entre muestras según su perfil de expresión.

Se observan agrupaciones dentro del dendrograma que pueden estar relacionadas con los tratamientos, pero no hay una separación completamente definida entre ellos.

En este caso, la mezcla de muestras en algunos clusters sugiere que pueden existir otras fuentes de variabilidad, como efectos batch o diferencias biológicas individuales.

En el apéndice se encuentra el código completo con los resultados de cada paso de forma detallada.

5. Discusión.

El análisis exploratorio realizado permitió identificar ciertas diferencias en los perfiles metabolómicos entre los grupos de estudio, aunque con un notable solapamiento. Esto puede deberse a múltiples factores, como la heterogeneidad biológica de las muestras, la presencia de efectos batch o la necesidad de aplicar métodos de normalización más avanzados.

El PCA mostró que las dos primeras componentes principales explican el 38.11% de la variabilidad total, lo cual sugiere que existen otros factores influyentes en los datos. La falta de una clara separación entre los grupos en el clustering jerárquico refuerza la idea de que podría ser necesario explorar otras fuentes de variabilidad, como posibles sesgos técnicos en la adquisición de datos o la necesidad de transformación adicional de los datos.

Para mejorar la interpretación de los resultados, se podrían aplicar técnicas de corrección de efectos batch, evaluar la calidad de los datos antes de la normalización y emplear otros métodos de reducción de dimensionalidad. Además, sería útil realizar un análisis funcional para correlacionar las variaciones observadas con rutas metabólicas relevantes.

6. Conclusiones.

El análisis realizado muestra que los datos metabolómicos presentan una estructura compleja con cierta diferenciación entre los grupos de estudio, aunque con solapamientos significativos.

El PCA y el clustering sugieren que el tratamiento influye en la variabilidad de los datos, pero no es la única fuente de variabilidad (ya que la variabilidad explicada por las dos primeras componentes principales es relativamente baja). Sería recomendable evaluar si existen efectos batch o realizar una normalización adicional para mejorar la separación de los grupos y mejorar la interpretación de los datos.

7. Referencias.

[1] *metaboData/Datasets/2024-fobitools-UseCase_1 at main · nutrimetabolomics/metaboData*. (n.d.). Retrieved April 2, 2025, from https://github.com/nutrimetabolomics/metaboData/tree/main/Datasets/2024-fobitools-UseCase_1

[2] *Banco de trabajo de metabolómica: repositorio de datos del NIH*. (n.d.). Retrieved April 2, 2025, from https://www.metabolomicsworkbench.org/data/DRCCMetadata.php?Mode=Study&DataMode=AllData&StudyID=ST000291&StudyType=MS&ResultType=5#DataTabs

[3] *Introduction to microarray data exploration and analysis with basic R functions*. (n.d.). Retrieved April 2, 2025, from https://aspteaching.github.io/Analisis_de_datos_omicos-Ejemplo_0-Microarrays/ExploreArrays.html

[4] *The ExpressionSet Data Object • BS831*. (n.d.). Retrieved April 2, 2025, from https://montilab.github.io/BS831/articles/docs/ExpressionSet.html