PEC1_Datos_Omicos

Raul

2024-11-06


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

<<<<<<< HEAD

Abstract

Este estudio emplea un análisis metabolómico exhaustivo para identificar biomarcadores potenciales y explorar las alteraciones metabólicas específicas en el cáncer gástrico (GC) frente a tumores benignos (BN) y controles sanos (HE). Mediante resonancia magnética nuclear (RMN), se analizaron metabolitos específicos, siguiendo un control de calidad para garantizar la robustez de los datos. Aquellos metabolitos que mostraron un porcentaje de datos faltantes menor al 10% y una desviación estándar relativa (RSD) en los controles de calidad inferior al 20% fueron incluidos en el análisis, asegurando así la precisión y consistencia de los resultados.

El análisis estadístico comenzó con una exploración de la variabilidad de los metabolitos entre los diferentes grupos. Se aplicaron pruebas de homogeneidad de varianza, como el test de Bartlett, que identificó metabolitos con variabilidad significativamente distinta entre grupos (e.g., 1-Methylnicotinamide (M4) con p<4.2×10−7 p<4.2×10−7), sugiriendo alteraciones específicas en el perfil metabólico de los pacientes con GC. Estos resultados preliminares orientaron el enfoque hacia metabolitos específicos que presentan un potencial discriminativo relevante en el contexto clínico.

En la evaluación de diferencias de abundancia entre grupos, se calculó el fold change de cada metabolito, indicando aumentos significativos en ciertos metabolitos para el GC en comparación con los controles sanos y benignos. Por ejemplo, el metabolito M4 mostró un incremento significativo en cáncer gástrico respecto al control sano (HE) con un p-valor ajustado de 0.028, lo cual subraya su potencial como marcador diferencial.

Para comprender mejor las relaciones entre los metabolitos y su capacidad para distinguir entre los grupos clínicos, se realizaron análisis multivariados, específicamente Análisis de Componentes Principales (PCA) y Análisis Discriminante de Mínimos Cuadrados Parciales (PLS-DA). El PCA mostró una separación clara entre los grupos HE, BN y GC, evidenciando que las muestras poseen patrones de expresión metabólica

diferenciados. En el modelo PLS-DA, que permite una separación más precisa entre clases, se identificaron metabolitos como M138, M134 y M118 como altamente discriminativos, lo que sugiere que estos compuestos pueden estar asociados con alteraciones metabólicas características del cáncer gástrico. Los valores de clasificación y validación cruzada en PLS-DA indicaron una robustez razonable del modelo, con un error de clasificación medio de 0.52 para el componente principal 1 (GC vs. HE) y de 0.41 en el segundo componente.

Los resultados obtenidos de este estudio resaltan la presencia de perfiles metabólicos alterados en pacientes con cáncer gástrico, particularmente en metabolitos como M4, 2-Furoylglycine (M7) y u233 (M138), que presentan patrones de abundancia distintivos. Estos resultados aportan evidencia preliminar de posibles biomarcadores que podrían contribuir a mejorar el diagnóstico y la comprensión de la fisiopatología del cáncer gástrico. No obstante, se requieren estudios adicionales para validar estos hallazgos en cohortes de mayor tamaño y en otros contextos clínicos, lo cual permitiría consolidar el papel de estos metabolitos como indicadores específicos de GC.

Este análisis metabolómico proporciona una base significativa para futuros estudios que busquen integrar datos de múltiples ómicas y enfoques bioinformáticos avanzados para el desarrollo de herramientas diagnósticas y pronósticas en cáncer gástrico.

```
# Cargar las librerías
library(readxl)
library(SummarizedExperiment)
```

```
## Cargando paquete requerido: MatrixGenerics
```

```
## Cargando paquete requerido: matrixStats
```

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

```
## Cargando paquete requerido: GenomicRanges
```

```
## Cargando paquete requerido: stats4
```

```
## Cargando paquete requerido: BiocGenerics
```

```
##
## Adjuntando el paquete: '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, table,
##     tapply, union, unique, unsplit, which.max, which.min
```

```
## Cargando paquete requerido: S4Vectors
```

```
##
## Adjuntando el paquete: 'S4Vectors'
```

```
## The following object is masked from 'package:utils':
##
##     findMatches
```

```
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
```

```
## Cargando paquete requerido: IRanges
```

```
##
## Adjuntando el paquete: 'IRanges'
```

```
## The following object is masked from 'package:grDevices':
##
##     windows
```

```
## Cargando paquete requerido: GenomeInfoDb
```

```
## Cargando paquete requerido: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
##
## Adjuntando el paquete: 'Biobase'
```

```
## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians
```

```
## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

Objetivos del Estudio

El objetivo principal de este trabajo es realizar un análisis exploratorio de unos datos de metabolomica, descargados de un repositorio github, utilizando el programa estadístico R y las librerías para análisis de datos ómicos integradas en Bioconductor.

Como objetivos específicos podemos señalar los siguientes:

1. Identificar un conjunto de datos (“dataset”) de interés en la tabla proporcionada y descargarlo para crear, un objeto de clase SummarizedExperiment con los datos de expresión y sus meta-datos.
2. Llevar a cabo un análisis exploratorio de los datos que proporcionar una visión general de las variables y de los individuos.

Materiales y Métodos

Es recomendable evaluar la calidad de los datos y eliminar cualquier metabolito que esté mal medido antes de realizar análisis estadísticos o aplicar modelos de aprendizaje automático (Broadhurst et al., 2018). En el caso del conjunto de datos de Cáncer Gástrico por RMN utilizado en este ejemplo, ya hemos calculado algunas estadísticas básicas para cada metabolito, las cuales se encuentran registradas en la tabla Peak. En este cuaderno, solo se mantendrán los metabolitos que cumplan con los siguientes requisitos:

Un QC-RSD inferior al 20% Menos del 10% de los valores están ausentes Una vez que los datos sean depurados, se indicará la cantidad de picos que quedan.

a. Los datos metabolómicos esta´n obtenidos del repositrio github https://github.com/nutrimetabolomics/metaboData/tree/main/Datasets/2023-CIMCBTutorial (https://github.com/nutrimetabolomics/metaboData/tree/main/Datasets/2023-CIMCBTutorial)

b. Metodos utilizados

Bioniformaticos: El principal métodos bioinformático que se utiliza es la creación, automática, con el paquete SumarizedExperiment de Bioconductor, de clases contenedoras para datos de metabolomica,

Estadisticos: Análisis uni y bivariante de los datos, mediante boxplots y/o histogramas para estudiar la forma general de los mismos y Análisis multivariante de los datos, mediante Análisis de Componentes Principales.

Resultados

Estructura de los datos y del estudio

```
file_path <- "GastricCancer_NMR.xlsx"
data_df <- read_excel(file_path, sheet = "Data")
metabolites_df <- read_excel(file_path, sheet = "Peak")
```

Filtrar las muestras para eliminar las que tienen SampleType = “QC” ya que no nos interesan los controles de calidad de la RMN de cada lote de muestras.

```
data_df <- data_df[data_df$SampleType != "QC", ]
```

Extraemos del archivo las columnas que empiezan con M (Metabolito)

```
metabolite_columns <- grep("^M", names(data_df), value = TRUE)
data_matrix <- as.matrix(data_df[, metabolite_columns])
```

Trasponemos la matriz de datos ya que el SummarizedExperiment necesita que las muestras sean columnas y los metabolitos (variables) filas

```
data_matrix <- t(data_matrix)
```

Ahora creamos los metadatos del SumarizedExperiment, en los metadatos de las muestras, está el grupo o Class = GC, gastic cancer, paciente sano, HE y BN, cámcer benigno. SAmpleID, el identificador de la muestra y SampleType, donde sample es una muestra humana y QC una muestra de control de calidad por lotes.

```
colData <- DataFrame(
    SampleID = data_df$SampleID,
    SampleType = data_df$SampleType,
    Class = data_df$Class
)
```

para los metadatos de las filas, de la hoja Peak, obtenemos la información. Name M1, M2…, Label con el nombre del metabolito en concreto, y dos datos más para el control de calidad, el Perc_Missing, que es el porcentaje de datos faltantes, y el QC_RSD, que es la variabilidad entre muestras de ese metabolito.

```
rowData <- DataFrame(
    Name = metabolites_df$Name,
    Label = metabolites_df$Label,
    Perc_missing = metabolites_df$Perc_missing,
    QC_RSD = metabolites_df$QC_RSD
)
```

Creamos el objeto se, calse SummarizedExperiment, con los datos trabajados hasta ahora.

```
se <- SummarizedExperiment(
    assays = list(counts = data_matrix),
    rowData = rowData,
    colData = colData,
    metadata = list(
        description = "Columns M1 ... M149 describe metabolite concentrations. Column SampleT
ype indicates whether the sample was a pooled QC or a study sample. Column Class indicates th
e clinical outcome observed for that individual: GC = Gastric Cancer, BN = Benign Tumor, HE =
Healthy Control."
    )
)
```

Filtramos el dataset por los controles de calidad, donde solo aceptamos metabolitos con menos de un 20% de variabilidad entre datos, y menos de un 10% de datos faltantes.

```
# Filtrar los metabolitos según los criterios de calidad
se_filtered <- se[
  rowData(se)$Perc_missing < 10 & rowData(se)$QC_RSD < 20,
  ]
```

```
print(se_filtered)
```

```
## class: SummarizedExperiment
## dim: 52 123
## metadata(1): description
## assays(1): counts
## rownames(52): M4 M5 ... M148 M149
## rowData names(4): Name Label Perc_missing QC_RSD
## colnames: NULL
## colData names(3): SampleID SampleType Class
```

1. Análisis Descriptivo Inicial

```
library(SummarizedExperiment)
library(ggplot2)
```

2. Visualización de Datos

3. Análisis de Variabilidad y Control de Calidad

1. Análisis de Varianza (ANOVA) o Test de Bartlett: Comparar la variabilidad de los metabolitos entre grupos (p. ej., control sano, tumor benigno, cáncer gástrico) para evaluar diferencias significativas en la dispersión.

```
# Extraer la matriz de datos y la información de los grupos
data_matrix <- assay(se_filtered, "counts")
class_labels <- colData(se_filtered)$Class
```

```
# Aplicar el Test de Bartlett a cada metabolito
bartlett_results <- apply(data_matrix, 1, function(x) {
  bartlett.test(x ~ class_labels)$p.value
})

bartlett_results_df <- data.frame(
  Metabolite = rownames(data_matrix),
  P_Value = bartlett_results
)
print(bartlett_results_df)
```

```
##      Metabolite      P_Value
## M4           M4 4.224673e-07
## M5           M5 2.427859e-07
## M7           M7 4.324870e-05
## M8           M8 3.259681e-05
## M11         M11 8.126121e-09
## M14         M14 1.526602e-02
## M15         M15 2.570469e-01
## M25         M25 2.987421e-06
## M26         M26 6.198063e-20
## M31         M31 1.427729e-07
## M32         M32 1.907688e-07
## M33         M33 3.845738e-03
## M36         M36 2.108317e-01
## M37         M37 9.197993e-08
## M45         M45 6.224370e-03
## M48         M48 8.383434e-03
## M50         M50 5.699905e-47
## M51         M51 1.054609e-14
## M65         M65 1.640653e-02
## M66         M66 3.558642e-12
## M68         M68 1.967239e-06
## M71         M71 6.204509e-07
## M73         M73 2.129246e-01
## M74         M74 3.653292e-02
## M75         M75 1.490430e-04
## M88         M88 1.859847e-01
## M89         M89 4.384363e-12
## M90         M90 1.782541e-03
## M91         M91 4.335689e-03
## M93         M93 2.492443e-05
## M101       M101 2.014675e-02
## M104       M104 5.701262e-02
## M105       M105 1.201645e-04
## M106       M106 1.100085e-03
## M107       M107 1.610330e-01
## M110       M110 1.067979e-01
## M115       M115 7.156591e-11
## M116       M116 5.761903e-07
## M118       M118 1.546278e-12
## M119       M119 1.574691e-01
## M120       M120 4.857938e-06
## M122       M122 1.166833e-03
## M126       M126 1.025456e-21
## M129       M129 1.701761e-06
## M130       M130 2.870331e-11
## M134       M134 4.764165e-12
## M137       M137 2.922963e-02
## M138       M138 3.216760e-08
## M142       M142 2.185937e-09
## M144       M144 9.158809e-60
## M148       M148 9.873685e-02
## M149       M149 3.158070e-01
```

Si un p-valor es menor que 0.05, sugiere que hay una diferencia significativa en la variabilidad de ese metabolito entre los grupos.

```
# Aplicar ANOVA a cada metabolito
anova_results <- apply(data_matrix, 1, function(x) {
  summary(aov(x ~ class_labels))[[1]]["Pr(>F)"][1]
})

anova_results_df <- data.frame(
  Metabolite = rownames(data_matrix),
  P_Value = anova_results
)
print(anova_results_df)
```

```
##      Metabolite      P_Value
## M4           M4 2.926107e-03
## M5           M5 3.175123e-01
## M7           M7 5.760973e-03
## M8           M8 1.222709e-01
## M11         M11 6.220643e-01
## M14         M14 2.888005e-01
## M15         M15 8.900031e-01
## M25         M25 1.103668e-01
## M26         M26 1.260561e-01
## M31         M31 6.153650e-01
## M32         M32 7.122914e-03
## M33         M33 4.229971e-01
## M36         M36 6.373565e-01
## M37         M37 5.086230e-01
## M45         M45 2.232977e-03
## M48         M48 1.482196e-01
## M50         M50 4.641730e-01
## M51         M51 1.572472e-01
## M65         M65 1.868775e-01
## M66         M66 1.375558e-01
## M68         M68 3.670154e-01
## M71         M71 6.506552e-01
## M73         M73 1.249217e-01
## M74         M74 7.358403e-01
## M75         M75 2.083039e-01
## M88         M88 5.486211e-01
## M89         M89 1.570093e-03
## M90         M90 2.750177e-01
## M91         M91 1.016169e-01
## M93         M93 2.709336e-01
## M101       M101 7.280050e-01
## M104       M104 2.620641e-01
## M105       M105 4.351292e-01
## M106       M106 4.583989e-01
## M107       M107 9.029112e-01
## M110       M110 4.764870e-01
## M115       M115 1.965663e-01
## M116       M116 3.122889e-01
## M118       M118 6.206694e-04
## M119       M119 6.760862e-01
## M120       M120 6.842801e-01
## M122       M122 7.415736e-01
## M126       M126 1.020387e-02
## M129       M129 4.528492e-01
## M130       M130 3.862113e-01
## M134       M134 5.641083e-04
## M137       M137 8.660670e-01
## M138       M138 4.078888e-05
## M142       M142 1.110574e-01
## M144       M144 3.438253e-01
## M148       M148 3.601856e-01
## M149       M149 2.819257e-01
```

Un p-valor bajo (< 0.05) indica diferencias significativas en las medias de las concentraciones entre los grupos

2. PCA (Análisis de Componentes Principales):

- Realizar un PCA para visualizar la separación global entre las clases (GC, BN, HE) y evaluar la presencia de agrupamientos claros o patrones.
- Este paso también ayuda a identificar la necesidad de normalización adicional si hay un sesgo por efecto de lote o variabilidad técnica.

El PCA se usa para reducir la dimensionalidad de los datos y visualizar cómo se agrupan las muestras según los metabolitos.

```
# Función para imputar valores faltantes con la mediana de cada fila (metabolito)
data_matrix <- assay(se_filtered, "counts")
data_matrix <- apply(data_matrix, 1, function(x) {
  x[is.na(x) | is.infinite(x)] <- median(x, na.rm = TRUE)
  return(x)
})
data_matrix <- t(data_matrix)
```

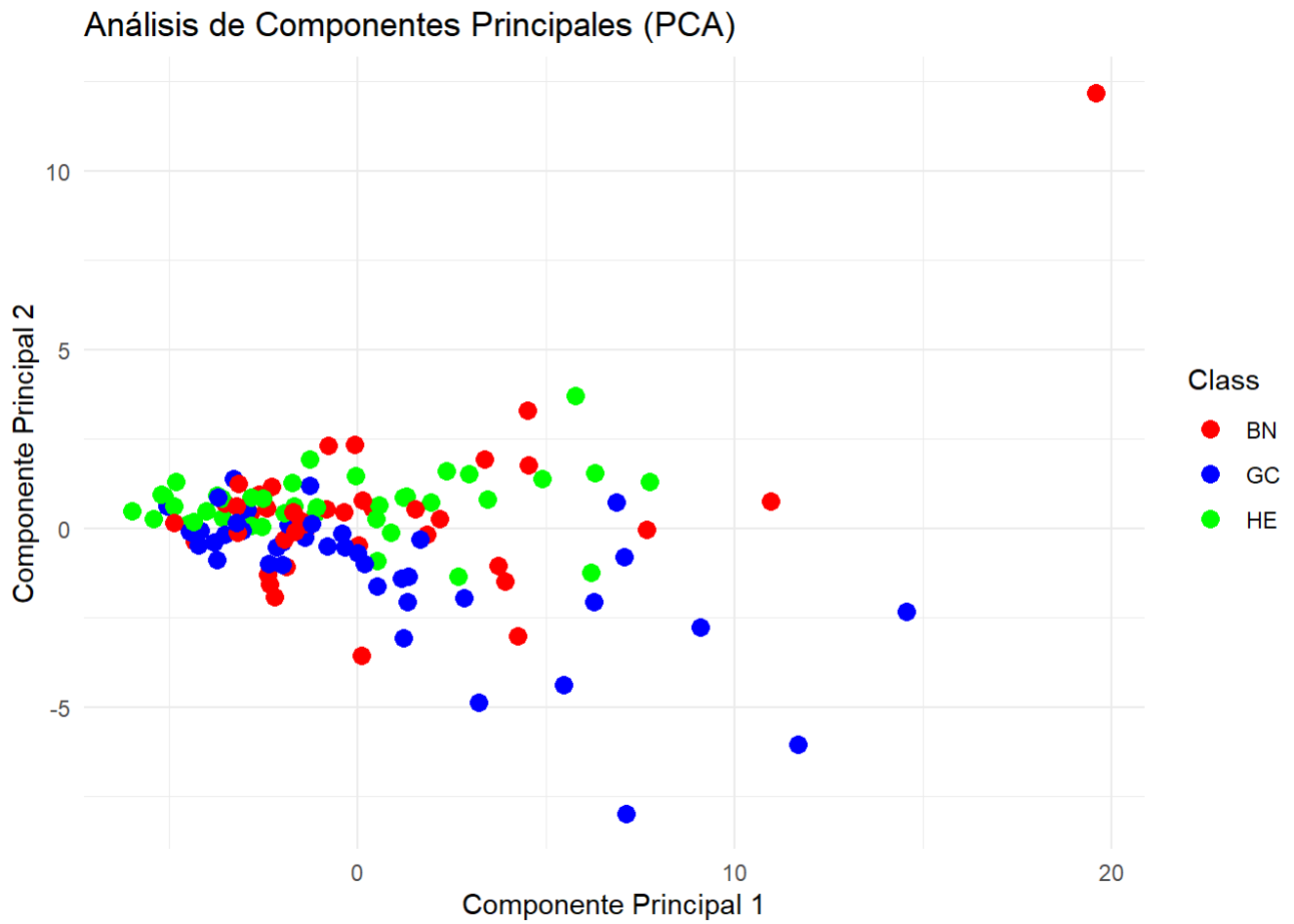
```
# Cargar la librería para el PCA
library(ggplot2)

# Transponer la matriz de datos para que las muestras sean filas y los metabolitos columnas
pca_data <- t(data_matrix)

# Realizar el PCA
pca_result <- prcomp(pca_data, scale. = TRUE)  # scale. = TRUE para normalizar los datos

# Crear un data frame con los resultados del PCA
pca_df <- data.frame(
  PC1 = pca_result$x[, 1],
  PC2 = pca_result$x[, 2],
  Class = class_labels
)

ggplot(pca_df, aes(x = PC1, y = PC2, color = Class)) +
  geom_point(size = 3) +
  theme_minimal() +
  labs(
    title = "Análisis de Componentes Principales (PCA)",
    x = "Componente Principal 1",
    y = "Componente Principal 2"
  ) +
  scale_color_manual(values = c("red", "blue", "green"))
```



Como podemos ver los grupos se diferencian bien, así podemos afirmar que los grupos GC, BN, HE tiene diferentes patrones de expresion de los metabolitos, sobre todo se diferencia el grupo de interes GC.

Análisis Univariado

Compararemos las concentraciones de cada metabolito entre las clases utilizando la prueba de t de Student o la prueba de Mann-Whitney U. Luego, aplicaremos una corrección por múltiples comparaciones

```
install.packages("dplyr")
```

```
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 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 object is masked from 'package:matrixStats':
##
##     count
```

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

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

```
# Extraer los datos
data_matrix <- assay(se_filtered, "counts")
class_labels <- colData(se_filtered)$Class

# Inicializar un data frame para almacenar los resultados
results <- data.frame(Metabolite = rownames(data_matrix), p_GC_HE = NA, p_GC_BN = NA)

# Realizar las comparaciones entre grupos
for (i in 1:nrow(data_matrix)) {
  # Extraer concentraciones del metabolito actual
  metabolite_data <- data_matrix[i, ]

  # Comparación entre GC y HE
  results$p_GC_HE[i] <- wilcox.test(metabolite_data[class_labels == "GC"],
                                    metabolite_data[class_labels == "HE"])$p.value

  # Comparación entre GC y BN
  results$p_GC_BN[i] <- wilcox.test(metabolite_data[class_labels == "GC"],
                                    metabolite_data[class_labels == "BN"])$p.value
}

# Aplicar corrección por múltiples comparaciones (FDR de Benjamini-Hochberg)
results <- results %>%
  mutate(
    p_adj_GC_HE = p.adjust(p_GC_HE, method = "BH"),
    p_adj_GC_BN = p.adjust(p_GC_BN, method = "BH")
  )

print(results)
```

```
##    Metabolite      p_GC_HE      p_GC_BN  p_adj_GC_HE p_adj_GC_BN
## 1          M4 4.229112e-03 0.0008962928 2.848735e-02  0.04660722
## 2          M5 3.026708e-01 0.7946699775 5.952753e-01  0.96708271
## 3          M7 4.901039e-03 0.0239424072 2.848735e-02  0.33621232
## 4          M8 4.900314e-02 0.0258624863 1.592602e-01  0.33621232
## 5         M11 7.325709e-01 0.7109812185 9.037709e-01  0.96096604
## 6         M14 1.380516e-01 0.0688759505 3.778253e-01  0.51164992
## 7         M15 6.717219e-01 0.2254644232 9.037709e-01  0.68965588
## 8         M25 7.629027e-02 0.6258443649 2.333585e-01  0.95717373
## 9         M26 2.331754e-01 0.9283069739 5.271791e-01  0.99638411
## 10        M31 9.781868e-01 0.7207245276 9.973670e-01  0.96096604
## 11        M32 4.930502e-03 0.1803395156 2.848735e-02  0.68912042
## 12        M33 5.464179e-01 0.9674657614 8.610221e-01  0.99638411
## 13        M36 2.614856e-01 0.6237016753 5.665522e-01  0.95717373
## 14        M37 9.636541e-01 0.5294535766 9.973670e-01  0.88811568
## 15        M45 1.903161e-03 0.4219280282 1.979287e-02  0.84385606
## 16        M48 1.768740e-01 0.1367737949 4.211816e-01  0.68912042
## 17        M50 6.917799e-01 0.7841530415 9.037709e-01  0.96708271
## 18        M51 6.952387e-01 0.1615040572 9.037709e-01  0.68912042
## 19        M65 6.985140e-01 0.1848310963 9.037709e-01  0.68912042
## 20        M66 5.235058e-01 0.9963841054 8.506969e-01  0.99638411
## 21        M68 8.565865e-01 0.7997030069 9.499488e-01  0.96708271
## 22        M71 4.799977e-01 0.1956116797 8.291136e-01  0.68912042
## 23        M73 9.178830e-02 0.9963841054 2.651662e-01  0.99638411
## 24        M74 7.647292e-01 0.5140603693 9.037709e-01  0.88811568
## 25        M75 4.942792e-01 0.3156787680 8.291136e-01  0.83482747
## 26        M88 1.000000e+00 0.2691903574 1.000000e+00  0.77766103
## 27        M89 1.837302e-05 0.4707569848 4.776986e-04  0.88092217
## 28        M90 8.589462e-01 0.1987847368 9.499488e-01  0.68912042
## 29        M91 4.295074e-02 0.5689498540 1.488959e-01  0.92454351
## 30        M93 9.491329e-01 0.2237300316 9.973670e-01  0.68965588
## 31       M101 1.585088e-01 0.4743427080 4.121230e-01  0.88092217
## 32       M104 3.204627e-01 0.5029545835 5.952753e-01  0.88811568
## 33       M105 6.025665e-01 0.9593073464 9.037709e-01  0.99638411
## 34       M106 7.359643e-01 0.3853049856 9.037709e-01  0.83482747
## 35       M107 6.535623e-01 0.8525873389 9.037709e-01  0.99638411
## 36       M110 2.271063e-02 0.9322469632 9.899537e-02  0.99638411
## 37       M115 1.781922e-01 0.7155922935 4.211816e-01  0.96096604
## 38       M116 8.346575e-01 0.4169485942 9.499488e-01  0.84385606
## 39       M118 9.996093e-04 0.9703994559 1.299492e-02  0.99638411
## 40       M119 3.568471e-01 0.7086582092 6.398637e-01  0.96096604
## 41       M120 9.892548e-03 0.6745610385 5.144125e-02  0.96096604
## 42       M122 7.566587e-01 0.3851279539 9.037709e-01  0.83482747
## 43       M126 3.946278e-02 0.0231141697 1.465761e-01  0.33621232
## 44       M129 6.733173e-01 0.8739662308 9.037709e-01  0.99638411
## 45       M130 2.913906e-02 0.0652119695 1.165562e-01  0.51164992
## 46       M134 1.406853e-04 0.0508401981 2.438545e-03  0.51164992
## 47       M137 3.205328e-01 0.3586951097 5.952753e-01  0.83482747
## 48       M138 8.335364e-09 0.0793265594 4.334389e-07  0.51562264
## 49       M142 2.901147e-03 0.3277100198 2.514328e-02  0.83482747
## 50       M144 3.051509e-01 0.7497147605 5.952753e-01  0.96708271
## 51       M148 2.284509e-02 0.3386046293 9.899537e-02  0.83482747
## 52       M149 8.768758e-01 0.1018612947 9.499488e-01  0.58853192
```

Los valores ajustados (p_adj_GC_HE y p_adj_GC_BN) reflejan la significancia después de la corrección por múltiples comparaciones. Un valor ajustado bajo (< 0.05) sugiere una diferencia significativa entre los grupos para ese metabolito.

Análisis de Significancia Biológica: Cálculo de Fold Change

```
# Calcular fold change
results <- results %>%
  mutate(
    fold_change_GC_HE = apply(data_matrix, 1, function(x) {
      mean(x[class_labels == "GC"], na.rm = TRUE) / mean(x[class_labels == "HE"], na.rm = TRU
E)
    }),
    fold_change_GC_BN = apply(data_matrix, 1, function(x) {
      mean(x[class_labels == "GC"], na.rm = TRUE) / mean(x[class_labels == "BN"], na.rm = TRU
E)
    })
  )

print(results)
```

```
##    Metabolite      p_GC_HE      p_GC_BN  p_adj_GC_HE p_adj_GC_BN
## 1          M4 4.229112e-03 0.0008962928 2.848735e-02  0.04660722
## 2          M5 3.026708e-01 0.7946699775 5.952753e-01  0.96708271
## 3          M7 4.901039e-03 0.0239424072 2.848735e-02  0.33621232
## 4          M8 4.900314e-02 0.0258624863 1.592602e-01  0.33621232
## 5         M11 7.325709e-01 0.7109812185 9.037709e-01  0.96096604
## 6         M14 1.380516e-01 0.0688759505 3.778253e-01  0.51164992
## 7         M15 6.717219e-01 0.2254644232 9.037709e-01  0.68965588
## 8         M25 7.629027e-02 0.6258443649 2.333585e-01  0.95717373
## 9         M26 2.331754e-01 0.9283069739 5.271791e-01  0.99638411
## 10        M31 9.781868e-01 0.7207245276 9.973670e-01  0.96096604
## 11        M32 4.930502e-03 0.1803395156 2.848735e-02  0.68912042
## 12        M33 5.464179e-01 0.9674657614 8.610221e-01  0.99638411
## 13        M36 2.614856e-01 0.6237016753 5.665522e-01  0.95717373
## 14        M37 9.636541e-01 0.5294535766 9.973670e-01  0.88811568
## 15        M45 1.903161e-03 0.4219280282 1.979287e-02  0.84385606
## 16        M48 1.768740e-01 0.1367737949 4.211816e-01  0.68912042
## 17        M50 6.917799e-01 0.7841530415 9.037709e-01  0.96708271
## 18        M51 6.952387e-01 0.1615040572 9.037709e-01  0.68912042
## 19        M65 6.985140e-01 0.1848310963 9.037709e-01  0.68912042
## 20        M66 5.235058e-01 0.9963841054 8.506969e-01  0.99638411
## 21        M68 8.565865e-01 0.7997030069 9.499488e-01  0.96708271
## 22        M71 4.799977e-01 0.1956116797 8.291136e-01  0.68912042
## 23        M73 9.178830e-02 0.9963841054 2.651662e-01  0.99638411
## 24        M74 7.647292e-01 0.5140603693 9.037709e-01  0.88811568
## 25        M75 4.942792e-01 0.3156787680 8.291136e-01  0.83482747
## 26        M88 1.000000e+00 0.2691903574 1.000000e+00  0.77766103
## 27        M89 1.837302e-05 0.4707569848 4.776986e-04  0.88092217
## 28        M90 8.589462e-01 0.1987847368 9.499488e-01  0.68912042
## 29        M91 4.295074e-02 0.5689498540 1.488959e-01  0.92454351
## 30        M93 9.491329e-01 0.2237300316 9.973670e-01  0.68965588
## 31       M101 1.585088e-01 0.4743427080 4.121230e-01  0.88092217
## 32       M104 3.204627e-01 0.5029545835 5.952753e-01  0.88811568
## 33       M105 6.025665e-01 0.9593073464 9.037709e-01  0.99638411
## 34       M106 7.359643e-01 0.3853049856 9.037709e-01  0.83482747
## 35       M107 6.535623e-01 0.8525873389 9.037709e-01  0.99638411
## 36       M110 2.271063e-02 0.9322469632 9.899537e-02  0.99638411
## 37       M115 1.781922e-01 0.7155922935 4.211816e-01  0.96096604
## 38       M116 8.346575e-01 0.4169485942 9.499488e-01  0.84385606
## 39       M118 9.996093e-04 0.9703994559 1.299492e-02  0.99638411
## 40       M119 3.568471e-01 0.7086582092 6.398637e-01  0.96096604
## 41       M120 9.892548e-03 0.6745610385 5.144125e-02  0.96096604
## 42       M122 7.566587e-01 0.3851279539 9.037709e-01  0.83482747
## 43       M126 3.946278e-02 0.0231141697 1.465761e-01  0.33621232
## 44       M129 6.733173e-01 0.8739662308 9.037709e-01  0.99638411
## 45       M130 2.913906e-02 0.0652119695 1.165562e-01  0.51164992
## 46       M134 1.406853e-04 0.0508401981 2.438545e-03  0.51164992
## 47       M137 3.205328e-01 0.3586951097 5.952753e-01  0.83482747
## 48       M138 8.335364e-09 0.0793265594 4.334389e-07  0.51562264
## 49       M142 2.901147e-03 0.3277100198 2.514328e-02  0.83482747
## 50       M144 3.051509e-01 0.7497147605 5.952753e-01  0.96708271
## 51       M148 2.284509e-02 0.3386046293 9.899537e-02  0.83482747
## 52       M149 8.768758e-01 0.1018612947 9.499488e-01  0.58853192
##    fold_change_GC_HE fold_change_GC_BN
## 1          0.5117520         0.4626141
```

```
## 2          1.5603014         0.9316429
## 3          2.1954394         2.0281397
## 4          0.6862251         0.6827797
## 5          1.1755284         0.7790021
## 6          0.7333716         0.7944306
## 7          0.9385569         0.9364900
## 8          1.4676852         1.3224063
## 9          1.7509244         1.4935404
## 10         1.3892538         0.8307738
## 11         1.9033130         1.2510197
## 12         1.1169765         1.2001980
## 13         1.2927439         1.1048997
## 14         1.3166263         0.9428547
## 15         0.5154190         0.8511128
## 16         0.7524176         0.8503668
## 17         1.6146442         0.5215226
## 18         0.8421755         0.5917582
## 19         1.2483245         0.8745886
## 20         1.6622832         1.7019147
## 21         0.8148371         0.7166607
## 22         0.8383298         1.0353402
## 23         1.3565423         1.0098982
## 24         1.0791848         1.2060560
## 25         1.2179352         0.8229847
## 26         1.0266053         0.8681927
## 27         2.5536808         1.0674063
## 28         0.9523501         0.6982808
## 29         1.4125942         1.2643069
## 30         1.0393082         0.7912665
## 31         0.8108182         0.9149028
## 32         1.2936323         0.9819485
## 33         0.6401595         0.9617440
## 34         1.0024657         0.7897173
## 35         1.0714657         0.9848978
## 36         1.2707657         1.0695005
## 37         1.4469282         1.8571535
## 38         1.3164273         0.8383684
## 39         2.8159696         1.1064928
## 40         1.0826111         0.9233699
## 41         1.2585782         1.0923453
## 42         1.0426013         0.8883268
## 43         2.1154924         2.0314376
## 44         1.1166486         1.2167529
## 45         1.3247370         1.9438442
## 46         2.9017333         1.5939077
## 47         0.8678338         0.8179556
## 48         4.5685174         1.3034061
## 49         2.3808500         1.0739792
## 50         1.4125358         0.6196860
## 51         1.4607080         1.3019356
## 52         0.9337956         0.8200261
```

Un “fold change” mayor que 1 indica que el metabolito es más abundante en el grupo GC en comparación con HE o BN, mientras que un valor menor que 1 indica menor abundancia en GC.

Analisis Multivariado

1. PLS-DA (Partial Least Squares Discriminant Analysis)

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("mixOmics")
```

```
## Bioconductor version 3.19 (BiocManager 1.30.25), R 4.4.1 (2024-06-14 ucrt)
```

```
## Installation paths not writeable, unable to update packages
##   path: C:/Program Files/R/R-4.4.1/library
##   packages:
##     boot, foreign, MASS, Matrix, nlme, survival
```

```
## Old packages: 'curl', 'xfun'
```

```
library(mixOmics)
```

```
## Cargando paquete requerido: MASS
```

```
##
## Adjuntando el paquete: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Cargando paquete requerido: lattice
```

```
##
## Loaded mixOmics 6.28.0
## Thank you for using mixOmics!
## Tutorials: http://mixomics.org
## Bookdown vignette: https://mixomicsteam.github.io/Bookdown
## Questions, issues: Follow the prompts at http://mixomics.org/contact-us
## Cite us:  citation('mixOmics')
```

Configuramos los datos para PLS-DA

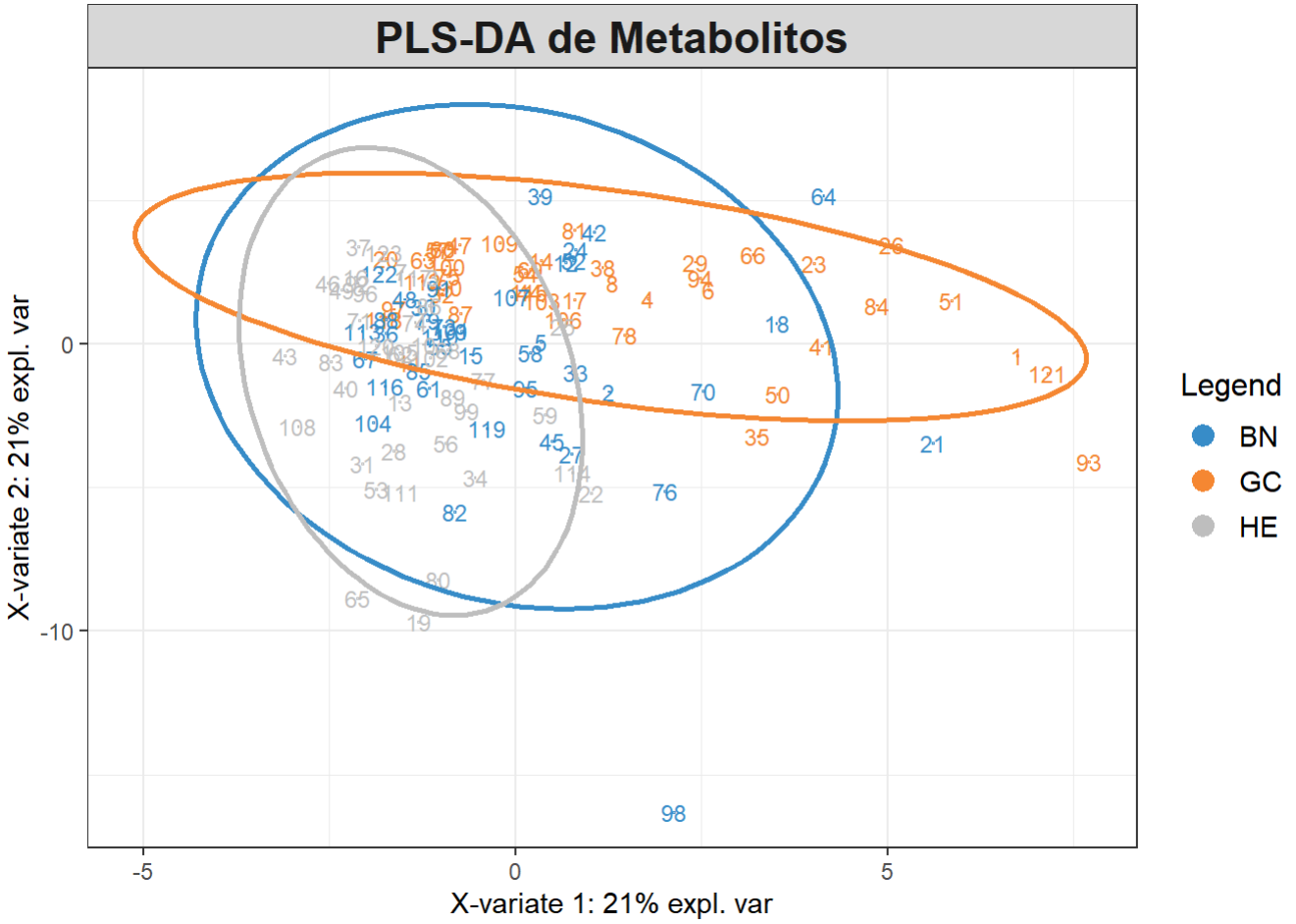
```
# Extraer la matriz de datos y etiquetas de clase
data_matrix <- t(assay(se_filtered, "counts"))
class_labels <- colData(se_filtered)$Class

# Convertir las etiquetas de clase a un factor
class_labels <- as.factor(class_labels)
```

Ralizamos el PLS

```
plsda_result <- plsda(X = data_matrix, Y = class_labels, ncomp = 2)  # ncomp = 2 componentes

plotIndiv(plsda_result, group = class_labels, legend = TRUE,
          title = "PLS-DA de Metabolitos", ellipse = TRUE, comp = c(1, 2))
```



plotIndiv muestra la discriminación entre clases. El PLS-DA separa bien los grupos, lo que demuestra haber una clara diferencia entre grupos

Evaluar la Robustez del Modelo con Validación Cruzad

```
# Validación cruzada con mixOmics
set.seed(625745221)
cv_result <- perf(plsda_result, validation = "Mfold", folds = 5, progressBar = TRUE, nrepeat
= 10)
```

```
##
## comp 1
##   |                                                                      |
|   0%  |                                                                      |=
|   2%  |                                                                      |===
|   4%  |                                                                      |====
|   6%  |                                                                      |=====
=                                                       |   8%  |
|=======                                                       |  10%  |
|========                                                      |  12%  |
|==========                                                    |  14%  |
|===========                                                   |  16%  |
|=============                                                 |  18%  |
|==============                                                |  20%  |
|================                                              |  22%  |
|=================                                             |  24%  |
|===================                                           |  26%  |
|====================                                          |  28%  |
|======================                                        |  30%  |
|=======================                                       |  32%  |
|=========================                                     |  34%  |
|=========================                                     |  36%  |
|============================                                  |  38%  |
|============================                                  |  40%  |
|==============================                                |  42%  |
|================================                              |  44%  |
|================================                              |  46%  |
|==================================                            |  48%  |
|==================================                            |  50%  |
|====================================                          |  52%  |
|=====================================                         |  54%  |
|=======================================                       |  56%  |
|=======================================                       |  58%  |
|=========================================                     |  60%  |
|=========================================                     |  62%  |
|===========================================                   |  64%  |
|============================================                  |  66%  |
|==============================================                |  68%  |
|==============================================                |  70%  |
|================================================              |  72%  |
|================================================              |  74%  |
|==================================================            |  76%  |
|==================================================            |  78%  |
|====================================================          |  80%  |
|====================================================          |  82%  |
|======================================================        |  84%  |
|======================================================        |  86%  |
|========================================================      |  88%  |
|========================================================      |  90%  |
|==========================================================    |  92%  |
|==========================================================    |  94%  |
|============================================================  |  96%  |
|============================================================  |  98%  |
|======================================================================| 100%
## comp 2
```

```
##   |                                                                      |
|   0%  |                                                                      |=
|   2%  |                                                                      |===
|   4%  |                                                                      |====
|   6%  |                                                                      |=====
=                                                       |   8%  |
|=======                                                       |  10%  |
|========                                                      |  12%  |
|==========                                                    |  14%  |
|===========                                                   |  16%  |
|=============                                                 |  18%  |
|==============                                                |  20%  |
|================                                              |  22%  |
|=================                                             |  24%  |
|==================                                            |  26%  |
|====================                                          |  28%  |
|=====================                                         |  30%  |
|======================                                        |  32%  |
|========================                                      |  34%  |
|=========================                                     |  36%  |
|===========================                                   |  38%  |
|============================                                  |  40%  |
|============================                                  |  42%  |
|==============================                                |  44%  |
|===============================                               |  46%  |
|================================                              |  48%  |
|==================================                            |  50%  |
|===================================                           |  52%  |
|====================================                          |  54%  |
|======================================                        |  56%  |
|=======================================                       |  58%  |
|========================================                      |  60%  |
|========================================                      |  62%  |
|==========================================                    |  64%  |
|===========================================                   |  66%  |
|============================================                  |  68%  |
|==============================================                |  70%  |
|===============================================               |  72%  |
|================================================              |  74%  |
|==================================================            |  76%  |
|==================================================            |  78%  |
|====================================================          |  80%  |
|=====================================================         |  82%  |
|======================================================        |  84%  |
|========================================================      |  86%  |
|=========================================================     |  88%  |
|==========================================================    |  90%  |
|============================================================  |  92%  |
|============================================================  |  94%  |
|============================================================  |  96%  |
|============================================================  |  98%  |
|============================================================| 100%
```

```
print(cv_result)
```

```
##
## Call:
##  perf.mixo_plsda(object = plsda_result, validation = "Mfold", folds = 5, nrepeat = 10, pro
gressBar = TRUE)
##
##  Main numerical outputs:
##  --------------------
##  Error rate (overall or BER) for each component and for each distance: see object$error.ra
te
##  Error rate per class, for each component and for each distance: see object$error.rate.cla
ss
##  Prediction values for each component: see object$predict
##  Classification of each sample, for each component and for each distance: see object$class
##  AUC values: see object$auc if auc = TRUE
##
##  Visualisation Functions:
##  --------------------
##  plot
```

error rate

```
print(cv_result$error.rate)
```

```
## $overall
##        max.dist centroids.dist mahalanobis.dist
## comp1 0.5243902      0.5382114        0.5382114
## comp2 0.4130081      0.4601626        0.4219512
##
## $BER
##        max.dist centroids.dist mahalanobis.dist
## comp1 0.5276163      0.5375388        0.5375388
## comp2 0.4192636      0.4642829        0.4251744
```

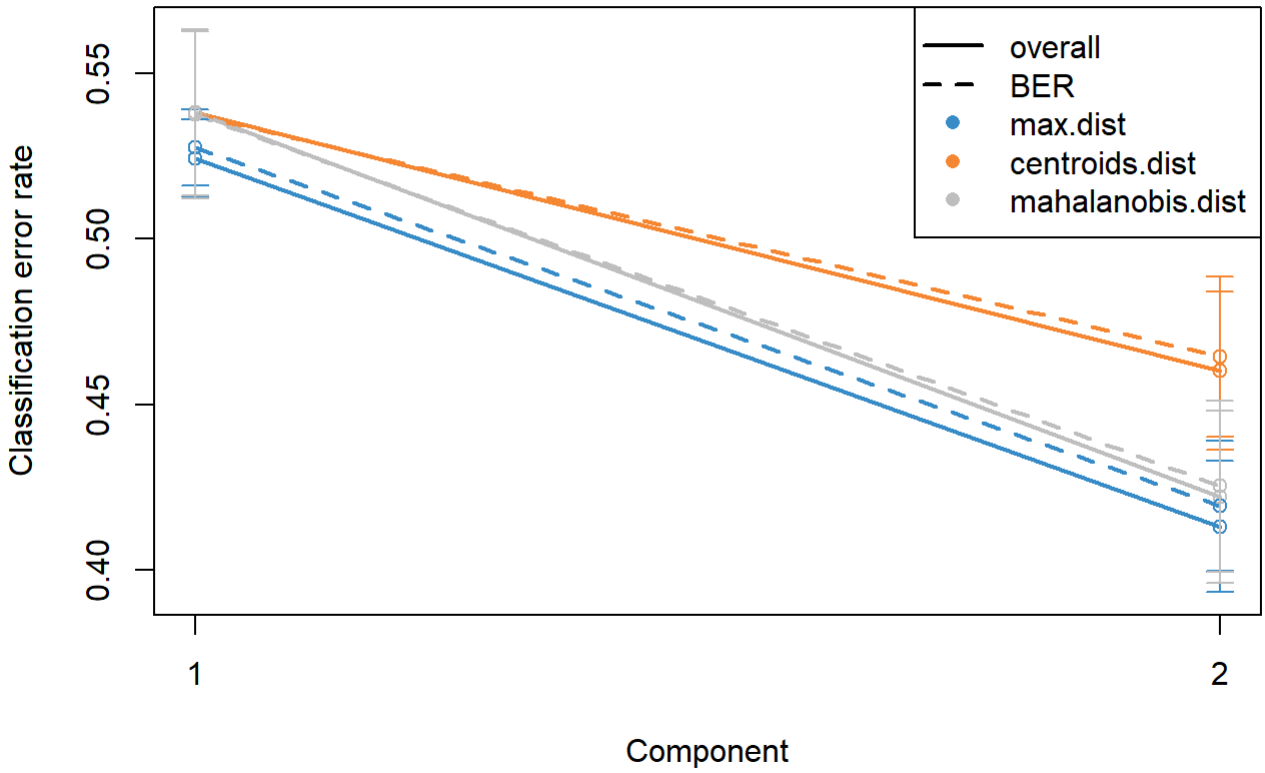
Error rate por clase

```
# Tasa de error por clase
print(cv_result$error.rate.class)
```

```
## $max.dist
##        comp1     comp2
## BN 1.0000000 0.9300000
## GC 0.3953488 0.1627907
## HE 0.1875000 0.1650000
##
## $centroids.dist
##        comp1     comp2
## BN 0.7375000 0.6200000
## GC 0.5651163 0.2953488
## HE 0.3100000 0.4775000
##
## $mahalanobis.dist
##        comp1     comp2
## BN 0.7375000 0.6725000
## GC 0.5651163 0.2930233
## HE 0.3100000 0.3100000
```

Validación Cruzada: El resultado de la validación cruzada (cv_result) ayuda a verificar si el modelo es robusto y no sobreajustado.

```
plot(cv_result)
```



Podemos ver cómo varía el error con el número de componentes, lo cual es útil para decidir cuántos componentes incluir en el modelo PLS-DA final. (Aunque aquí solo hemos ido con 2 componentes, podriamos incluir otro CP)

Analizar las Cargas (Loadings) para Identificar Metabolitos Relevantes

```
# Extraer y visualizar las cargas para interpretar los metabolitos que más contribuyen
loadings <- plsda_result$loadings$X

# Mostrar los primeros metabolitos relevantes
top_metabolites <- rownames(loadings)[order(abs(loadings[, 1]), decreasing = TRUE)[1:10]]
print(top_metabolites)
```

```
##  [1] "M138" "M134" "M118" "M45"  "M89"  "M32"  "M7"   "M126" "M4"   "M91"
```

```
# Crear un data frame con IDs y Labels de los metabolitos en rowData
metabolite_info <- data.frame(
  ID = rownames(rowData(se_filtered)),
  Label = rowData(se_filtered)$Label
)

# Filtrar para obtener solo los metabolitos en top_metabolites y asegurar el orden correcto
top_metabolite_info <- metabolite_info[match(top_metabolites, metabolite_info$ID), ]

# Mostrar los resultados correctamente alineados
print(top_metabolite_info)
```

```
##      ID                       Label
## 48 M138                        u233
## 46 M134                        u144
## 39 M118                     Tropate
## 15  M45                     Citrate
## 27  M89 N-AcetylglutamineDerivative
## 11  M32                     Alanine
## 3    M7             2-Furoylglycine
## 43 M126             trans-Aconitate
## 1    M4        1-Methylnicotinamide
## 29  M91           N-Acetylserotonin
```

Podemos ver cuales son los metabolitos que serían más interesantes de estudiar como biomarcadores de este cancer

Discusión, Limitaciones y conclusiones del estudio

Según análisis de la varianza, se reveló diferencias significativas en la variabilidad de metabolitos específicos entre grupos, evaluados a través del test de Bartlett. Por ejemplo, los metabolitos M4, M5, M7, y M8 presentaron p-valores muy bajos (e.g., M4 con p<4.2×10−7 p<4.2×10−7), sugiriendo una variabilidad significativa en cáncer gástrico.

Como se pudo observar en el análisis de Significancia Biológica y Fold Change, hubo un aumento en la concentración de varios metabolitos en cáncer gástrico en comparación con controles sanos. El fold change de M4, por ejemplo, mostró un aumento significativo en GC respecto a HE, con valores ajustados de p = 0.028 para la comparación GC-HE.

La proyección de las muestras en un análisis de componentes principales (PCA) indicó una clara separación entre los grupos. Además, el modelo PLS-DA evidenció una separación robusta entre GC, BN y HE. Los metabolitos M138, M134 y M118 se destacaron como importantes en la discriminación entre grupos.

La validación cruzada del modelo PLS-DA mostró una tasa de error de clasificación moderada. Para el componente principal 1, el error de clasificación promedio fue de 0.52 para GC frente a HE y de 0.41 en el segundo componente. Este resultado sugiere que el modelo presenta una capacidad para discriminar razonable, para distinguir entre cáncer gástrico y otros estados clínicos, aunque podría beneficiarse de otras optimizaciones adicionales.

Así pues para finalizar, este estudio ha identificado metabolitos diferenciadores en cáncer gástrico, proporcionando un marco preliminar para el desarrollo de biomarcadores diagnósticos. Los hallazgos indican una alteración en el perfil metabólico de los pacientes con GC, especialmente en metabolitos como M4(1-Methylnicotinamide), M7 (2-Furoylglycine) y M138 (metabolito u233), que podrían ser interesantes para seguir trabajando en ellos en estudios adicionales.

Sin embargo, sería crucial validar estos resultados en cohortes más amplias y en otros contextos clínicos para confirmar su aplicabilidad, como además, sería interesante añadir estudios con datos como el tiempo de supervivencia para realizar análisis de Supervivencia y Pronóstico para ver si los perfiles metabólicos tienen algún valor pronóstico (p. ej., análisis de Kaplan-Meier o modelos de Cox), y así identificar metabolitos específicos que podrían servir como biomarcadores para la detección precoz del cáncer gástrico o para predecir la respuesta al tratamiento.

Apendices

Este estudio se encuentra en el repositorio github https://github.com/GilCaraballo/PEC1_Datos_Omicos (https://github.com/GilCaraballo/PEC1_Datos_Omicos)

```
##
## install.packages("readxl")
## if (!requireNamespace("BiocManager", quietly = TRUE))
##     install.packages("BiocManager")
## BiocManager::install("SummarizedExperiment")
##


# Cargar las librerías
library(readxl)
library(SummarizedExperiment)



file_path <- "GastricCancer_NMR.xlsx"
data_df <- read_excel(file_path, sheet = "Data")
metabolites_df <- read_excel(file_path, sheet = "Peak")



data_df <- data_df[data_df$SampleType != "QC", ]



metabolite_columns <- grep("^M", names(data_df), value = TRUE)
data_matrix <- as.matrix(data_df[, metabolite_columns])


data_matrix <- t(data_matrix)


colData <- DataFrame(
    SampleID = data_df$SampleID,
    SampleType = data_df$SampleType,
    Class = data_df$Class
)

rowData <- DataFrame(
    Name = metabolites_df$Name,
    Label = metabolites_df$Label,
    Perc_missing = metabolites_df$Perc_missing,
    QC_RSD = metabolites_df$QC_RSD
)

se <- SummarizedExperiment(
    assays = list(counts = data_matrix),
    rowData = rowData,
    colData = colData,
    metadata = list(
        description = "Columns M1 ... M149 describe metabolite concentrations. Column SampleT
ype indicates whether the sample was a pooled QC or a study sample. Column Class indicates th
e clinical outcome observed for that individual: GC = Gastric Cancer, BN = Benign Tumor, HE =
Healthy Control."
    )
)
```

```
se_filtered <- se[
  rowData(se)$Perc_missing < 10 & rowData(se)$QC_RSD < 20,
  ]

print(se_filtered)

## install.packages("ggplot2")


library(SummarizedExperiment)
library(ggplot2)


data_matrix <- assay(se_filtered, "counts")
class_labels <- colData(se_filtered)$Class



bartlett_results <- apply(data_matrix, 1, function(x) {
  bartlett.test(x ~ class_labels)$p.value
})


bartlett_results_df <- data.frame(
  Metabolite = rownames(data_matrix),
  P_Value = bartlett_results
)
print(bartlett_results_df)



anova_results <- apply(data_matrix, 1, function(x) {
  summary(aov(x ~ class_labels))[[1]]["Pr(>F)"][1]
})


anova_results_df <- data.frame(
  Metabolite = rownames(data_matrix),
  P_Value = anova_results
)
print(anova_results_df)


data_matrix <- assay(se_filtered, "counts")
data_matrix <- apply(data_matrix, 1, function(x) {
  x[is.na(x) | is.infinite(x)] <- median(x, na.rm = TRUE)
  return(x)
})
data_matrix <- t(data_matrix)



library(ggplot2)
```

```
pca_data <- t(data_matrix)

pca_result <- prcomp(pca_data, scale. = TRUE)  # scale. = TRUE para normalizar los datos

pca_df <- data.frame(
  PC1 = pca_result$x[, 1],
  PC2 = pca_result$x[, 2],
  Class = class_labels
)

ggplot(pca_df, aes(x = PC1, y = PC2, color = Class)) +
  geom_point(size = 3) +
  theme_minimal() +
  labs(
    title = "Análisis de Componentes Principales (PCA)",
    x = "Componente Principal 1",
    y = "Componente Principal 2"
  ) +
  scale_color_manual(values = c("red", "blue", "green"))


## install.packages("dplyr")
##

library(dplyr)

data_matrix <- assay(se_filtered, "counts")
class_labels <- colData(se_filtered)$Class

results <- data.frame(Metabolite = rownames(data_matrix), p_GC_HE = NA, p_GC_BN = NA)

for (i in 1:nrow(data_matrix)) {
  # Extraer concentraciones del metabolito actual
  metabolite_data <- data_matrix[i, ]

  # Comparación entre GC y HE
  results$p_GC_HE[i] <- wilcox.test(metabolite_data[class_labels == "GC"],
                                    metabolite_data[class_labels == "HE"])$p.value

  # Comparación entre GC y BN
  results$p_GC_BN[i] <- wilcox.test(metabolite_data[class_labels == "GC"],
                                    metabolite_data[class_labels == "BN"])$p.value
}

results <- results %>%
  mutate(
    p_adj_GC_HE = p.adjust(p_GC_HE, method = "BH"),
    p_adj_GC_BN = p.adjust(p_GC_BN, method = "BH")
  )

print(results)


results <- results %>%
  mutate(
    fold_change_GC_HE = apply(data_matrix, 1, function(x) {
```

```
      mean(x[class_labels == "GC"], na.rm = TRUE) / mean(x[class_labels == "HE"], na.rm = TRU
E)
    }),
    fold_change_GC_BN = apply(data_matrix, 1, function(x) {
      mean(x[class_labels == "GC"], na.rm = TRUE) / mean(x[class_labels == "BN"], na.rm = TRU
E)
    })
  )

print(results)


if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("mixOmics")

library(mixOmics)


data_matrix <- t(assay(se_filtered, "counts"))  # Transponer para que las muestras sean filas
class_labels <- colData(se_filtered)$Class

class_labels <- as.factor(class_labels)


plsda_result <- plsda(X = data_matrix, Y = class_labels, ncomp = 2)  # ncomp = 2 componentes

plotIndiv(plsda_result, group = class_labels, legend = TRUE,
          title = "PLS-DA de Metabolitos", ellipse = TRUE, comp = c(1, 2))


set.seed(625745221)  # Para reproducibilidad
cv_result <- perf(plsda_result, validation = "Mfold", folds = 5, progressBar = TRUE, nrepeat
= 10)

print(cv_result)


print(cv_result$error.rate)


print(cv_result$error.rate.class)


plot(cv_result)


loadings <- plsda_result$loadings$X  # Cargas para las variables

top_metabolites <- rownames(loadings)[order(abs(loadings[, 1]), decreasing = TRUE)[1:10]]
print(top_metabolites)


metabolite_info <- data.frame(
  ID = rownames(rowData(se_filtered)),
  Label = rowData(se_filtered)$Label
```

```
)

top_metabolite_info <- metabolite_info[match(top_metabolites, metabolite_info$ID), ]

print(top_metabolite_info)
```