Análisis de Fosfopeptidos - PEC1

Edna Rey Agudelo

3 de noviembre de 2024

Contenido

Introducción

El presente estudio se centra en un conjunto de datos de fosfoproteómica derivado de un experimento diseñado específicamente para analizar las diferencias en la fosforilación de proteínas entre dos subtipos tumorales: MSS y PD, utilizando modelos de ratón PDX (xenoinjertos derivados de pacientes). Este conjunto de datos incluye abundancias normalizadas de señales de fosfopeptidos, obtenidas mediante espectrometría de masas. El objetivo fundamental de esta investigación es identificar fosfopeptidos que permitan distinguir entre estos dos subtipos tumorales. Esta diferenciación no solo contribuiría a la identificación de posibles biomarcadores, sino que también ofrecería una mayor comprensión de los mecanismos de señalización celular involucrados en cada subtipo, proporcionando así una base para futuras estrategias de intervención y tratamiento.

#2.Acerca de la base de datos

El conjunto de datos, proporcionado en formato Excel (TIO2+PTYR-human-MSS+MSIvsPD.XLSX), fue descargado y cargado en R a través del repositorio de GitHub nutrimetabolomics/metaboData para su análisis. Las muestras se agrupan en dos categorías:

Grupo MSS: Incluye las muestras M1, M5 y T49, cada una con dos réplicas. Grupo PD: Compuesto por las muestras M42, M43 y M64, también con dos réplicas cada una. Cada fosfopeptido en el conjunto de datos está representado por una serie de

columnas que contienen las abundancias normalizadas de cada réplica en cada muestra.

#3. Información acerca del dataset

```
library(BiocManager)

## Warning: package 'BiocManager' was built under R version 4.4.1

library(SummarizedExperiment)

## Warning: package 'SummarizedExperiment' was built under R version
4.4.1

## Cargando paquete requerido: MatrixGenerics

## Warning: package 'MatrixGenerics' was built under R version 4.4.1

## Cargando paquete requerido: matrixStats

## Warning: package 'matrixStats' was built under R version 4.4.1

##
## 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

## Warning: package 'GenomicRanges' was built under R version 4.4.1

## Cargando paquete requerido: stats4

## Cargando paquete requerido: BiocGenerics

## Warning: package 'BiocGenerics' was built under R version 4.4.1
```

```
##
## 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, saveRDS, setdiff,
##     table, tapply, union, unique, unsplit, which.max, which.min

## Cargando paquete requerido: S4Vectors

## Warning: package 'S4Vectors' was built under R version 4.4.1

##
## 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

## Warning: package 'IRanges' was built under R version 4.4.1

##
## Adjuntando el paquete: 'IRanges'

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

## Cargando paquete requerido: GenomeInfoDb

## Warning: package 'GenomeInfoDb' was built under R version 4.4.1

## Cargando paquete requerido: Biobase

## Warning: package 'Biobase' was built under R version 4.4.1

## 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

library(readxl)

## Warning: package 'readxl' was built under R version 4.4.1

library(readxl)

file_path <- "D:\\Descargas\\TIO2+PTYR-human-MSS+MSIvsPD (2).XLSX"

# acceso al archivo
original_data <- read_excel(file_path, sheet = "originalData")
targets <- read_excel(file_path, sheet = "targets")

## New names:
## • `Sample` -> `Sample...1`
## • `Sample` -> `Sample...2`

# Verificar los nombres de las columnas en 'targets' para ajustar los
nombres

sample_col <- colnames(targets)[1]
group_col <- colnames(targets)[4]

# Extraer matriz de expresión (abundancias)
#
expr_data <- as.matrix(original_data[, 6:ncol(original_data)])

# Remover las columnas innecesarias 'CLASS' y 'PHOSPHO' de expr_data
expr_data <- expr_data[, !(colnames(expr_data) %in% c("CLASS",
"PHOSPHO"))]

# Remover '_MSS' y '_PD' de los nombres de columna en expr_data
colnames(expr_data) <- sub("_MSS|_PD", "", colnames(expr_data))

# Renombrar filas con secuencias de fosfopeptidos en expr_data

rownames(expr_data) <-
make.unique(as.character(original_data$SequenceModifications))
```

```r
# Extraer metadatos de columnas (muestras)
col_data <- data.frame(
    Sample = targets[[sample_col]],
    Group = targets[[group_col]],
    row.names = targets[[sample_col]]
)

# Ajustar col_data para que solo contenga muestras presentes en expr_data
col_data <- col_data[rownames(col_data) %in% colnames(expr_data), ]

# Extraer metadatos de filas y eliminar duplicados
row_data <- original_data[, c("SequenceModifications", "Accession",
"Description")]

#  Verificar de que coincidan con expr_data
row_data$SequenceModifications <-
make.unique(as.character(row_data$SequenceModifications))
rownames(row_data) <- row_data$SequenceModifications
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```r
# Asegurarse de que rownames de row_data y expr_data coincidan
row_data <- row_data[rownames(expr_data), ]

# Summarized
se <- SummarizedExperiment(
    assays = list(counts = expr_data),
    rowData = row_data,
    colData = col_data
)
print(se)
```

```
## class: SummarizedExperiment
## dim: 1438 11
## metadata(0):
## assays(1): counts
## rownames(1438): LYPELSQYMGLSLNEEEIR[2] Phospho|[9] Oxidation
##   VDKVIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[35] Phospho ...
##   YQDEVFGGFVTEPQEESEEEVEEPEER[17] Phospho YSPSQNSPIHHIPSRR[1]
##   Phospho|[7] Phospho
## rowData names(3): SequenceModifications Accession Description
## colnames(11): M1_2 M5_1 ... M64_1 M64_2
## colData names(2): Sample Group
```

```r
summary(se)
```

```
## [1] "SummarizedExperiment object of length 1438 with 3 metadata
columns"
```

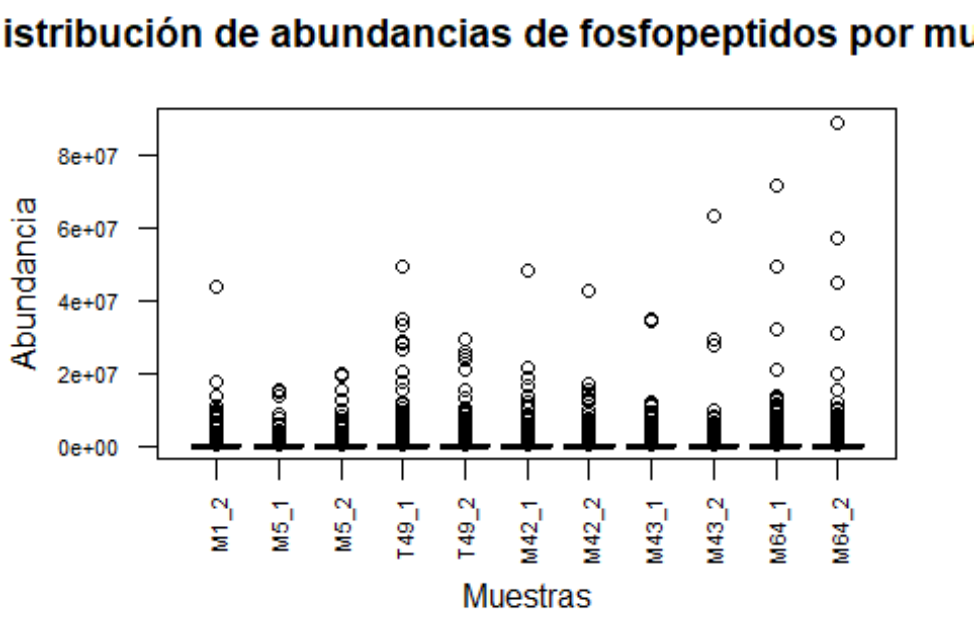
#4. Llevad a cabo una exploración del dataset que os proporcione una visión general del mismo en la línea de lo que hemos visto en las actividades

```
library(SummarizedExperiment)
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.4.1

library(stats)

# Estadísticas resumen para cada muestra
summary_stats <- summary(assay(se))
print(summary_stats)

##      M1_2               M5_1               M5_2              T49_1
##  Length:1438        Length:1438        Length:1438        Length:1438
##  Class :character   Class :character   Class :character   Class
:character
##  Mode  :character   Mode  :character   Mode  :character   Mode
:character
##     T49_2              M42_1              M42_2              M43_1
##  Length:1438        Length:1438        Length:1438        Length:1438
##  Class :character   Class :character   Class :character   Class
:character
##  Mode  :character   Mode  :character   Mode  :character   Mode
:character
##     M43_2              M64_1              M64_2
##  Length:1438        Length:1438        Length:1438
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character

apply(assay(se), 2, summary)

##        M1_2        M5_1        M5_2        T49_1       T49_2
M42_1
## Length "1438"      "1438"      "1438"      "1438"      "1438"
"1438"
## Class  "character" "character" "character" "character" "character"
"character"
## Mode   "character" "character" "character" "character" "character"
"character"
##        M42_2       M43_1       M43_2       M64_1       M64_2
## Length "1438"      "1438"      "1438"      "1438"      "1438"
## Class  "character" "character" "character" "character" "character"
## Mode   "character" "character" "character" "character" "character"
```

Para realizar un análisis cuantitativo se necesita convertir estas columnas de character a un formato numérico, por lo que se procede:

```
# Cargar los datos SummarizedExperiment
numeric_data <- as.matrix(assay(se))
```

```r
# Filtrar solo las columnas de muestras
sample_columns <- grep("M1|M5|T49|M42|M43|M64", colnames(numeric_data),
value = TRUE)
numeric_data <- numeric_data[, sample_columns]

numeric_data <- apply(numeric_data, 2, as.numeric)
numeric_data <- matrix(numeric_data, nrow = nrow(assay(se)), dimnames =
list(rownames(assay(se)), sample_columns))

# Crear el gráfico de caja para todas las muestras
par(mar = c(7, 4, 4, 2))
boxplot(numeric_data,
        main = "Distribución de abundancias de fosfopeptidos por
muestra",
        xlab = "Muestras", ylab = "Abundancia",
        las = 2, col = "lightblue",
        cex.axis = 0.7)
```




```r
# Obtener los datos numéricos
numeric_data <- as.matrix(assay(se))
sample_columns <- grep("M1|M5|T49|M42|M43|M64", colnames(numeric_data),
value = TRUE)
numeric_data <- numeric_data[, sample_columns]

# verificar  de que todos los datos sean numéricos
```

```r
numeric_data <- apply(numeric_data, 2, as.numeric)
numeric_data <- matrix(numeric_data, nrow = nrow(assay(se)), dimnames =
list(rownames(assay(se)), sample_columns))

# transformación logarítmica
log_counts <- log2(numeric_data + 1)
head(log_counts)
```

```
##                                                      M1_2     M5_1
M5_2
## LYPELSQYMGLSLNEEEIR[2] Phospho|[9] Oxidation      15.44077  0.00000
12.61428
## VDKVIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[35] Phospho 15.39674 11.03827
15.61988
## VIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[32] Phospho    17.39326 16.23863
18.23088
## HADAEMTGYVVTR[6] Oxidation|[9] Phospho            17.15222 16.67022
16.21159
## HADAEMTGYVVTR[9] Phospho                          13.05846 15.13398
15.42860
## STGPGASLGTGYDR[12] Phospho                        17.99887 17.51287
16.93005
##                                                      T49_1    T49_2
M42_1
## LYPELSQYMGLSLNEEEIR[2] Phospho|[9] Oxidation      10.150784 14.42094
0.00000
## VDKVIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[35] Phospho  7.965366 11.66185
10.36293
## VIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[32] Phospho    16.587005 17.55812
14.60109
## HADAEMTGYVVTR[6] Oxidation|[9] Phospho            19.560815 18.87618
19.97028
## HADAEMTGYVVTR[9] Phospho                          15.802369 15.08011
14.37397
## STGPGASLGTGYDR[12] Phospho                        22.097463 21.29478
19.03136
##                                                     M42_2     M43_1
M43_2
## LYPELSQYMGLSLNEEEIR[2] Phospho|[9] Oxidation       0.00000  9.596014
11.06187
## VDKVIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[35] Phospho  0.00000  0.000000
0.00000
## VIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[32] Phospho    14.01445 12.442498
0.00000
## HADAEMTGYVVTR[6] Oxidation|[9] Phospho            20.15035 21.960223
22.22017
## HADAEMTGYVVTR[9] Phospho                          15.59516 19.345397
18.53304
## STGPGASLGTGYDR[12] Phospho                        18.72948 16.479321
16.01015
```

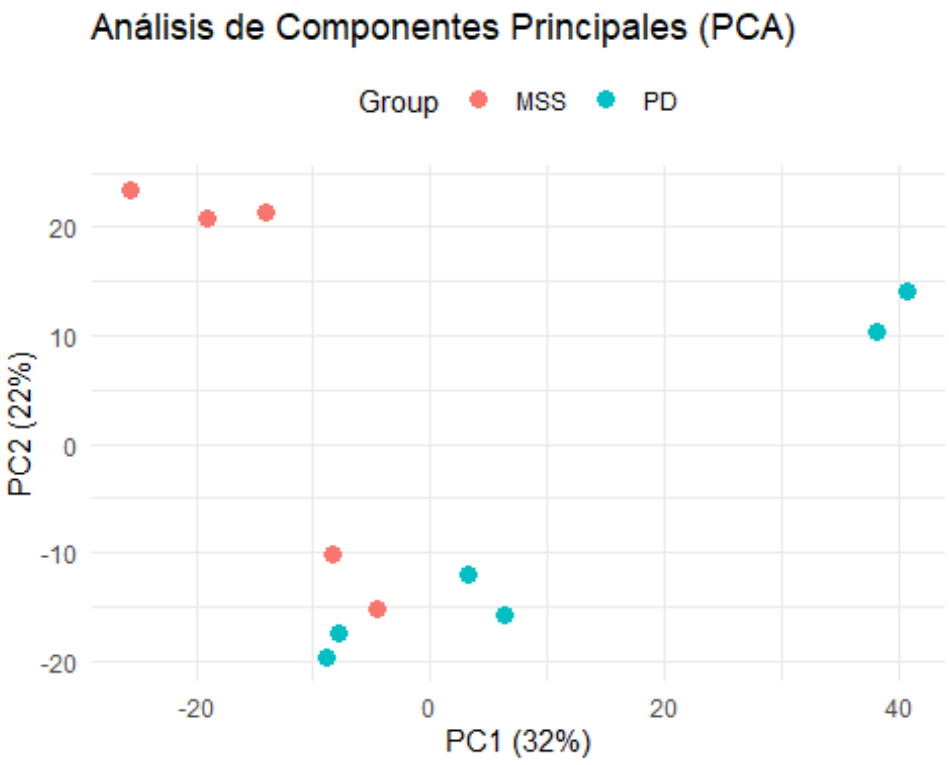
```
##                                                        M64_1     M64_2
## LYPELSQYMGLSLNEEEIR[2] Phospho|[9] Oxidation       10.83109 10.755647
## VDKVIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[35] Phospho  0.00000  9.803092
## VIQAQTAFSANPANPAILSEASAPIPHDGNLYPR[32] Phospho     11.67312 12.527222
## HADAEMTGYVVTR[6] Oxidation|[9] Phospho             21.56094 21.395769
## HADAEMTGYVVTR[9] Phospho                           17.96462 19.145111
## STGPGASLGTGYDR[12] Phospho                         17.89209 17.656715
```

```
# verificamos filas
log_counts_filtered <- log_counts[rowSums(log_counts != 0) > 0, ]

# Luego, verificamos columnas
log_counts_filtered <- log_counts_filtered[, colSums(log_counts_filtered
!= 0) > 0]

# Verificamos nuevamente si existen columnas con varianza cero
zero_variance_cols <- apply(log_counts_filtered, 2, var) == 0
log_counts_filtered <- log_counts_filtered[, !zero_variance_cols]

# Realizar el PCA sobre los datos filtrados y log-transformados
pca_result <- prcomp(t(log_counts_filtered), scale. = TRUE)
summary(pca_result)
```

```
## Importance of components:
##                             PC1     PC2     PC3      PC4      PC5
PC6
## Standard deviation     21.4302 17.7856 15.1849 11.60886 10.12279
8.06384
## Proportion of Variance  0.3198  0.2203  0.1606  0.09385  0.07136
0.04528
## Cumulative Proportion   0.3198  0.5401  0.7007  0.79452  0.86588
0.91116
##                             PC7     PC8     PC9    PC10      PC11
## Standard deviation     6.72869 6.06553 5.24728 4.24000 3.641e-14
## Proportion of Variance 0.03153 0.02562 0.01917 0.01252 0.000e+00
## Cumulative Proportion  0.94269 0.96831 0.98748 1.00000 1.000e+00
```

```
library(ggplot2)

pca_result <- prcomp(t(log_counts_filtered), scale. = TRUE)

#  PCA - graficar
pca_data <- as.data.frame(pca_result$x)
pca_data$Group <- colData(se)$Group  #



ggplot(pca_data, aes(x = PC1, y = PC2, color = Group)) +
    geom_point(size = 3) +
    labs(title = "Análisis de Componentes Principales (PCA)",
         x = paste0("PC1 (", round(100 *
```

```
summary(pca_result)$importance[2, 1], 1), "%)"),
         y = paste0("PC2 (", round(100 *
summary(pca_result)$importance[2, 2], 1), "%)")) +
    theme_minimal() +
    theme(legend.position = "top")
```



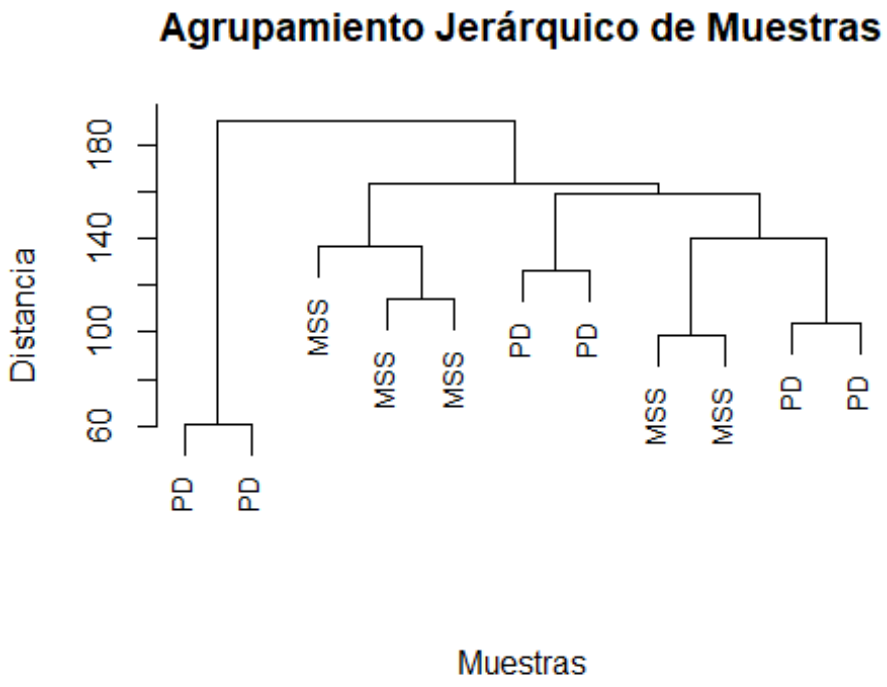
El análisis de componentes principales (PCA) se utilizó como herramienta para reducir la dimensionalidad del conjunto de datos y facilitar la visualización de patrones de agrupamiento entre las muestras. Este enfoque permitió identificar hallazgos clave:

1. Los dos primeros componentes principales (PC1 y PC2) lograron capturar aproximadamente el 55% de la variación total en el conjunto de datos. Esta proporción es significativa y sugiere que una gran parte de la información relevante para distinguir entre los grupos MSS y PD se encuentra representada en estos dos componentes.

2. En el espacio de PCA, se observó una separación clara entre los grupos MSS y PD. Las muestras MSS se agruparon de manera compacta, lo cual sugiere una homogeneidad en sus patrones de fosforilación. En contraste, las muestras PD mostraron una dispersión mayor, lo que indica una mayor heterogeneidad en sus perfiles de fosfopeptidos. Esta diferencia podría reflejar una activación diferencial de rutas de señalización en los tumores PD, lo cual estaría asociado con mecanismos de resistencia a tratamientos o una mayor agresividad tumoral.

3. La separación evidente entre MSS y PD en el PCA sugiere la existencia de fosfopeptidos cuya abundancia es específica de cada grupo. Estos fosfopeptidos específicos podrían actuar como biomarcadores moleculares, permitiendo caracterizar los dos subtipos y contribuyendo a una mejor estratificación de los pacientes.

```
#  distancia euclidiana y el agrupamiento jerárquico

dist_matrix <- dist(t(log_counts))
hclust_result <- hclust(dist_matrix, method = "average")

# Dendrograma
plot(hclust_result, labels = colData(se)$Group,
     main = "Agrupamiento Jerárquico de Muestras",
     xlab = "Muestras", ylab = "Distancia",
     sub = "", cex = 0.8)
```



Las muestras de MSS se agruparon de manera consistente, confirmando la homogeneidad observada en el análisis de PCA. Las muestras de PD, aunque también formaron un grupo, presentaron subagrupaciones dentro de su conjunto, lo cual sugiere cierta variabilidad interna en el grupo PD.

La variabilidad observada dentro del grupo PD podría estar asociada con diferencias en la activación de diversas rutas de señalización fosforiladas. Este hallazgo sugiere

que los tumores PD podrían abarcar subtipos moleculares adicionales o exhibir un perfil de señalización más complejo, lo que apunta a una posible diversidad funcional que podría influir en su comportamiento biológico y en su respuesta a tratamientos.

#5. Análisis

Las muestras del grupo PD mostraron una variabilidad notablemente mayor en las abundancias de fosfopeptidos en comparación con el grupo MSS. Esta variabilidad podría reflejar diferencias biológicas significativas en la activación de rutas de señalización entre los dos subtipos tumorales.

La separación identificada en el análisis PCA sugiere la existencia de fosfopeptidos que permiten distinguir con claridad entre tumores MSS y PD. Estos fosfopeptidos podrían actuar como marcadores moleculares, facilitando una caracterización más precisa de los subtipos de tumores. La heterogeneidad observada en el grupo PD podría indicar que este subtipo activa rutas de señalización específicas, distintas a las del grupo MSS. Estos hallazgos abren la posibilidad de desarrollar terapias dirigidas más eficaces y aportan información valiosa para comprender las bases moleculares que diferencian estos subtipos tumorales.

#6. Referencias

Buffalo, V., 2015. Bioinformatics Data Skills: Reproducible and Robust Research with Open Source Tools. O'Reilly Media.