UNIVERSIDAD DE GRANADA

Máster Universitario Oficial en Ciencia de Datos e Ingeniería de Computadores

Biología Computacional con Big Data-Omics e Ingeniería Biomédica

---

Guión II: Análisis de Expresión Diferencial empleando KnowSeq mediante Docker

---

Profesores:
Daniel Castillo Secilla
Luis Javier Herrera Maldonado
Ignacio Rojas Ruiz

18 de febrero de 2021

Índice

1. Objetivos

Durante el desarrollo de este guión se pretenden alcanzar los siguientes objetivos principales:

- Familiarizarse con el análisis de expresión diferencial con RNA-seq.
- Conocer y utilizar el entorno R, así como KnowSeq, un paquete R desarrollado para llevar a cabo análisis de expresión diferencial y de aprendizaje automático para datos de RNA-Seq.
- Familiarizarse con el uso de Docker, una herramienta cada día más usada para encapsular softwares y servicios, evitando problemas de dependencias y versiones.
- Llevar a cabo un análisis para el conjunto de datos de Lung Cancer disponible en SWAD.

2. Requisitos

Es necesario disponer de los siguientes pre-requisitos:

- Instalación de Docker en el host.
- Obtención de la imagen de KnowSeq disponible para Docker.
- Descarga del conjunto de datos disponible en SWAD.

3. Introducción

El principal objetivo del presente guión es la realización de un análisis completo de datos de RNA-seq, y más concretamente de expresión diferencial. RNA-seq está resultando de gran utilidad en esta última década para el conocimiento de la funcionalidad de sistemas biológicos completos a nivel genómico y molecular. Antes de la inclusión en el mercado de esta tecnología, estos estudios se realizaban mediante el uso de microarray, una tecnología muy confiable y de la que han salido un gran número de estudios, pero menos potente y precisa que RNA-seq [1].

Dentro del desarrollo de las tecnologías para el estudio de las diferentes ómicas, el proyecto Bioconductor ha supuesto un gran incremento en la utilización de estas en el estudio de dichas ómicas, principalmente por su apuesta por facilitar el análisis de datos de tanto de microarray como de datos provenientes de NGS (RNA-seq, DNA-seq, CHiP-seq, etc. . .) combinando además el conocimiento de diferentes áreas como la estadística, la informática o la biología molecular [2].

Una de las mayores utilidades de RNA-seq son los análisis de expresión diferencial. El objetivo de estos análisis es la identificación de conjuntos de genes que se expresan a diferentes niveles bajo diferentes condiciones [3].

Este tipo de estudios permiten detectar aquellos genes que pueden estar asociados a diferentes fenotipos, por ejemplo, células tumorales frente a células sanas. Sin embargo, incluso para los análisis más simples, es necesario tener especial cuidado en las estrategias elegidas para realizar una selección adecuada de los genes involucrados en la expresión diferencial.

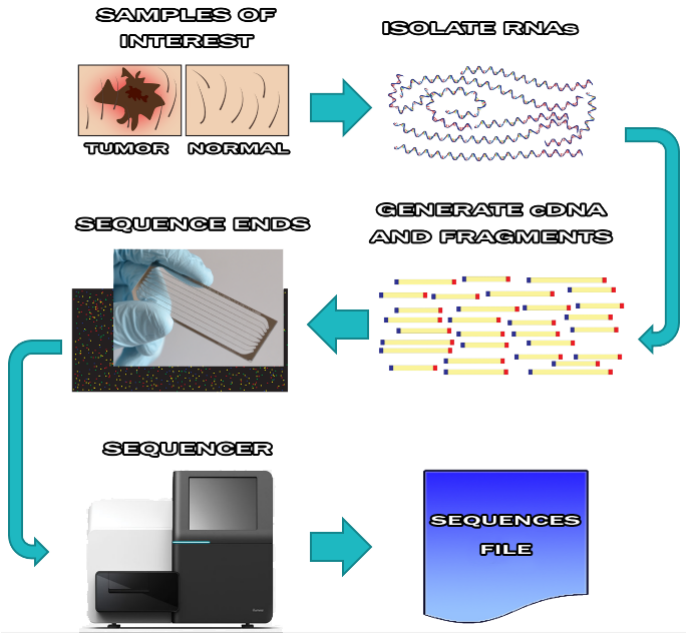

Figura 1: **Proceso de secuenciación de muestras de NGS. A partir de muestras de RNA procedentes del tejido deseado, se genera los fragmentos de ADN complementario a secuenciar. Dichos fragmentos son introducidos en un secuenciador que, finalmente crea un fichero (fastq) con los fragmentos de ADN secuenciados, dicho fichero está en "crudo" y necesita un preprocesamiento hasta llegar a los ficheros count, que finalmente son los que se usan para el estudio transcriptómico y de expresión de gen.**

3.1. Pipeline de extracción de expresión de gen

Una vez se tienen los datos crudos de una serie de pacientes, comienza un proceso con una carga computacional elevada para interpretar esos datos y conseguir finalmente una matriz con los valores de expresión de los genes con la que poder trabajar.

La salida de un secuenciar de nueva generación, se suele almacenar en un fichero SRA (Sequence Read Arhive), el cual contiene los datos en crudo secuenciados así como información sobre el alineamiento de secuencias, necesario para el posterior re-ensamblado del genoma. Este SRA suele ser un fichero binario o de texto, como los ficheros BAM o FASTQ, que se encapsula bajo este formato para homogeneizar la forma de trabajo de los científicos. Desde un fichero SRA se puede llegar a la expresión de gen pasando por una serie de pasos y herramientas, como se verá a continuación, siguiendo la Figura 2.

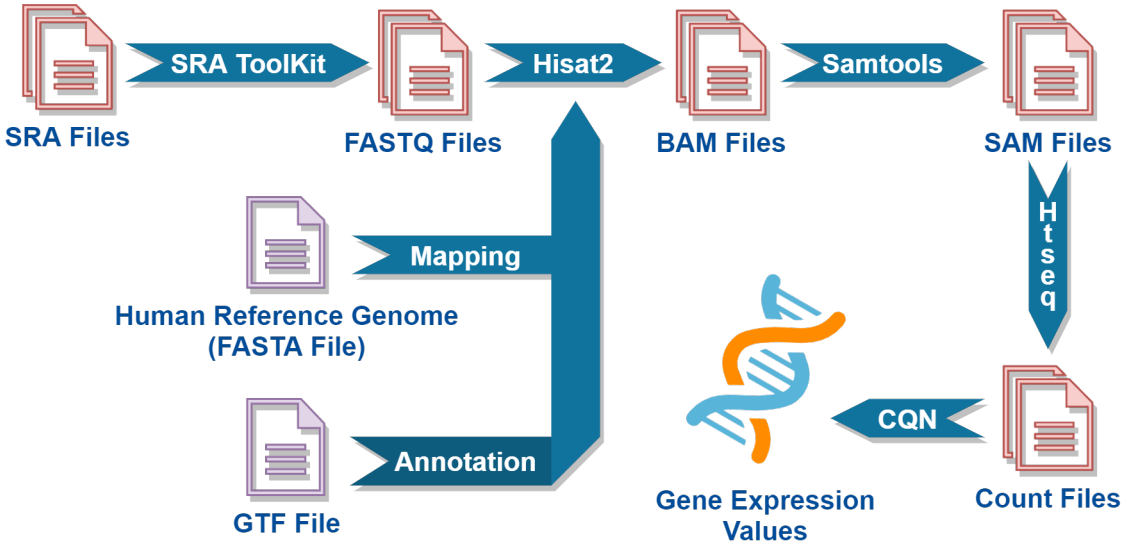

Figura 2: **Proceso de pre-procesamiento de datos crudos para llegar al valor de expresión de gen.**

3.1.1. De SRA a FASTQ (SRAToolKit)

Empezando desde un fichero SRA, el primer paso el conseguir el fichero FASTQ usando SRAToolKit, un conjunto de herramientas que permiten trabajar con ficheros SRA. Un fichero FASTQ es un fichero de texto que guarda 4 lineas por secuencia leída y cada línea a su vez contiene el nombre de la secuencia, la secuencia de nucleótidos, un signo para indicar que viene la calidad después y finalmente, la calidad de la secuencia leída. Este tipo de ficheros suelen ser muy pesado (en torno a 10 GB) ya que pueden llegar a contener decenas de millones de secuencias y 4 líneas por secuencia.

Formato interno de un fichero FASTQ

@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGT
+
$!'' * ((((***+))\%\%\%++)(\%\%\%\%),1***-+*''))**55CCF >>>>>>$
$CCCCCCC65$

3.1.2. De FASTQ a BAM/SAM (Hisat2)

El FASTQ guarda las secuencias en crudo para cada paciente o persona, pero de forma desordenada, obteniendo un puzzle de millones de piezas a ensamblar. Para ello, existen lo que se conoce como alineadores, cuya misión es re-ensamblar las secuencias para obtener la hebra de ADN de forma ordenada. No obstante, para poder decidir donde va cada pieza o secuencia, se necesita un referente o un modelo a seguir. Desde hace unos años, investigadores de todo el mundo usan genomas de diferentes lugares, etnias y sexos para determinar un genoma común que defina al ser humano. Dicho genoma es lo que se conoce como genoma de referencia humano y el alineador lo necesita para decidir donde va cada secuencia. Suele venir en formato FASTA, parecido al FASTQ pero sin incluir la calidad y, en el caso del genoma de referencia, ordenado. Junto al genoma de referencia, el alineador también necesitará información esencial para poder identificar que genes son los que hay en las secuencias y donde están localizados. Esto se consigo a través del fichero GTF (General Transfer Format),que complementa al genoma de referencia humano.

Formato interno de un fichero GTF

chr1 transcribed_unprocessed_pseudogene gene 11869 14409 . + . gene_id ENSG00000223972; gene_name DDX11L1; gene_source havana; gene_biotype transcribed_unprocessed_pseudogene;

chr1 processed_transcript transcript 11869 14409 . + . gene_id ENSG00000223972; transcript_id ENST00000456328; gene_name DDX11L1; gene_source havana

Con estos dos ficheros, se puede llevar a cabo el alineamiento de secuencias usando un alineador como Hisat2, aunque existen más, este es de los más usados y reconocidos. Una vez finalizado el proceso de alineamiento se obtendrá un fichero BAM (binary version of a SAM file) o SAM (Sequence Alignment Map). La principal diferencia es que el fichero SAM es un fichero de texto y se puede leer de forma sencilla mientras que el fichero BAM es un fichero binario. Estos ficheros contienen ya el genoma re-

emsablado y ordenado de una persona, permitiendo así trabajar con dichos datos.

Formato interno de un fichero BAM/SAM

https://davetang.org/wiki/tiki-index.php?page=SAM

3.1.3. De BAM/SAM a Count

Finalmente, después de este largo proceso de pre-procesamiento e interpretación, llega el momento de conseguir los ficheros Counts, con los que se puede calcular la expresión de gen. Estos ficheros son muy livianos a nivel de almacenamiento ya que solo guardan por cada fila dos campos: Identificador del gen y numero de veces que se ha leído en el genoma. Gracias al número de veces se puede saber posteriormente cual es la expresión de cada gen y si está expresión puede estar desembocando en la codificación de proteínas erróneas. Para ello, se usa una librería de Python llamada HTSEQ, que recorre el fichero BAM/SAM para ver las veces que aparece cada gen.

Formato interno de un fichero Count

```
ENSG00000000003.13 2800
ENSG00000000005.5 0
ENSG00000000419.11 2309
ENSG00000000457.12 515
ENSG00000000460.15 665
ENSG00000000938.11 833
ENSG00000000971.14 3840
```

Una vez conseguidos los Count, se deben re-normalizar para obtener los valores equivalentes de expresión de gen. Para ello se puede usar el método CQN que partiendo de esos valores, de cual es la longitud de cada gen en el genoma y del contenido de Guanina-Citosina de cada gen, calcula la expresión.

4. KnowSeq. Software automatizado para análisis de RNA-Seq

Para llevar a cabo dichos análisis, se ha desarrollado una herramienta actualmente pública en el repositorio más importante en Bioinformática (Bioconductor). Dicha herramienta es conocida como KnowSeq y se diseñó con el propósito de dar a los expertos un paquete software que encapsule todos los pasos necesarios para llevar a

cabo análisis de biomarcardores y su posterior evaluación mediante Machine Learning.

KnowSeq es totalmente modular y permite modificar el pipeline propuesto dependiendo del uso y del estudio que se quiera abordar. Para más información se puede visitar la página oficial de KnowSeq en Bioconductor (https://bioconductor.org/packages/release/bioc/html/KnowSeq.html), así como su manual de usuario, que desgrana paso a paso cada uno de los procesos incluidos en ella (https://bioconductor.org/packages/release/bioc/vignettes/KnowSeq/inst/doc/KnowSeq.pdf).

Para tener una idea acerca de los procesos y sub-procesos implementados en la herramienta, la Figura 3 muestra gráficamente todo el pipeline y posibilidades que ofrece KnowSeq.

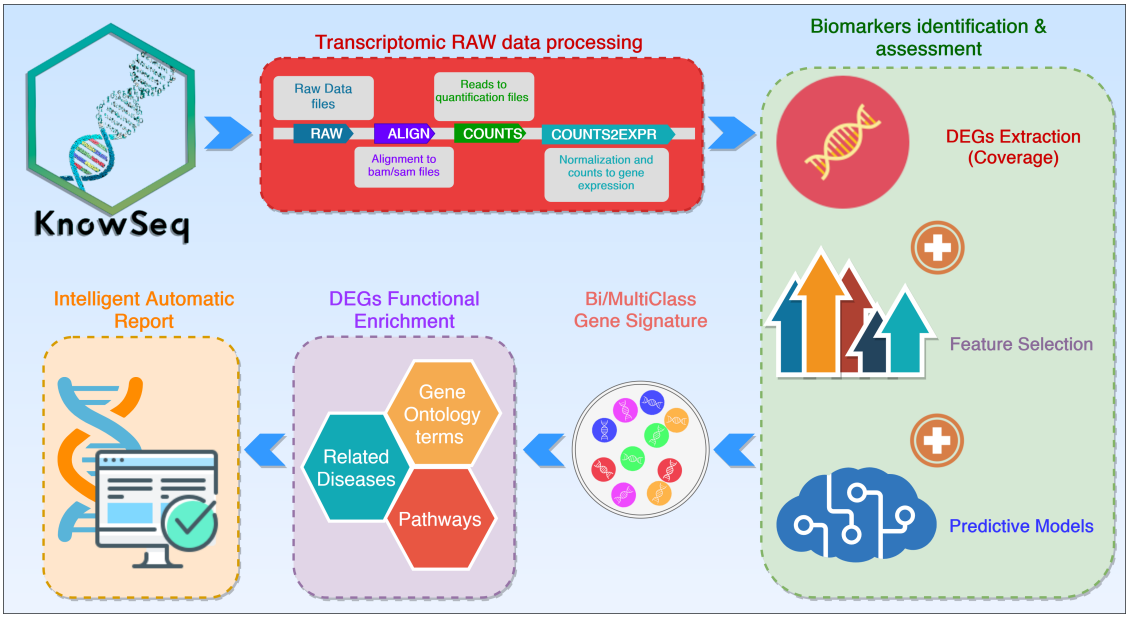

Figura 3: **Pipeline implementado en la herramienta KnowSeq.**

Para instalar localmente KnowSeq, se requiere la instalación de las herramientas de Bioconductor que permiten gestionar la instalación de los paquetes software que dicho repositorio contiene. Además, KnowSeq está disponible para R 4.0, lo que implica que en una versión anterior no se podrá instalar. Para llevar a cabo la instalación tanto de Bioconductor como de KnowSeq es necesario introducir en R el siguiente código:

```
# Instalamos BiocManager de Bioconductor
```

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

# Instalamos KnowSeq
BiocManager::install("KnowSeq")
```

No obstante, se ha creado una versión encapsulada de la herramienta mediante Docker, sacando provecho de este método de virtualización software para evitar la instalación de dependencias necesarias. Para poder llevar a cabo la instalación de Docker en Windows es necesario contar con Windows 10 64-bit: Pro, Enterprise, or Education (Build 17134 or later) debido a que la tecnología Hyper-V es requerida para poder ejecutar correctamente Docker. No obstante, se puede llegar a instalar Docker en Windows Home siguiendo las directrices del siguiente enlace: https://docs.docker.com/docker-for-windows/install-windows-home. Si no se tiene una de dichas versiones de Windows o el proceso da error, se recomienda instalar KnowSeq localmente o crear una Máquina Virtual con Ubuntu para ello.

5. Preparando Docker

La idea detrás de Docker es crear contenedores ligeros y portables para las aplicaciones software que puedan ejecutarse en cualquier máquina con Docker instalado, independientemente del sistema operativo que la máquina tenga por debajo, facilitando así también los despliegues.

La principal diferencia con una Máquina Virtual es que los recursos del host son compartidos con el contenedor, evitando así el consumo de recursos que genera una Máquina Virtual al virtualizar el Hardware necesario.

Aunque Docker se pensó para contener pequeños microservicios (Pequeños servicios como BBDD, servidores web, o procesos para controlar tareas concretas), puede usarse y se usa para contener también despliegues de aplicaciones mayores e incluso Sistemas Operativos.

A modo de curiosidad, la imagen que usaremos contiene un RStudio server encapsulado con KnowSeq ya preinstalado para acceder desde cualquier navegador web, así como una serie de datos de ejemplo para trabajar con él.

5.1. Instalando Docker en Ubuntu

Paso 1: Instalando docker

Instalar Docker descargándolo mediante su sitio web oficial o, si se usa un entorno GNU/Linux mediante los siguientes comandos:

curl -fsSL https://download.docker.com/linux/ubuntu/gpg —
sudo apt-key add -

sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs)
stable"

sudo apt get update

apt search docker-ce; sudo apt install docker-ce

sudo docker run hello-world

Para instalar Docker es necesario seguir las instrucciones del cuadro indicativo del Paso 1. Una vez instalado, se probará el hello world de docker para asegurar que efectivamente esta todo correctamente instalado.

Una vez comprobada la instalación, el contenedor de KnowSeq puede ser descargado para trabajar con la herramienta. Siguiendo el comando del cuadro indicativo del Paso 2, hará falta indicarle a Docker la imagen de KnowSeq, así como el parámetro -it para entrar al modo interactivo. Si el parámetro -it no se indica, el contenedor se desplegará y automáticamente se cerrara ya que no hay ninguna tarea indicada para que realice.

Paso 2: Descargando y arracando contenedor de KnowSeq

Para descargar y arrancar el contenedor de KnowSeq, hará falta introducir el siguiente comando donde your_password es una contraseña a elección del usuario:

Docker run -e PASSWORD=your_password -p 8787:8787 casedugr/knowseqbiocomp

Este comando despliega un servidor web que contiene un RStudio server al que se puede acceder a través del puerto 8787 por cualquier navegador. ***El usuario para ingresar en el RStudio server es rstudio y la contraseña aquella que cada persona ponga al arrancar el docker de KnowSeq.***

5.2. Instalando Docker en Windows y MacOS

Para instalar Docker tanto en Windows como en MacOS es necesario descargar Docker Desktop, el asistente de Docker para dichos sistemas operativos. Se puede hacer a través de los siguientes enlaces:

- Para Windows: https://desktop.docker.com/win/stable/Docker%20Desktop%20Installer.exe
- Para MacOS: https://desktop.docker.com/mac/stable/Docker.dmg

Una vez instalado Docker Desktop, se puede usar Docker a través de una terminal de forma normal. La pantalla principal de Docker Desktop muestra los contenedores arrancados así como las imágenes en local y en remoto asociadas al perfil de cada persona (Figura 4). Una vez instalado, tan solo hay que pedir la imagen de KnowSeq mediante el comando Docker mostrado en el recuadro del paso 2 de la instalación para Ubuntu. Como se ve en la Figura 5, al no encontrarlo de manera local, lo busco en el servidor remoto y comienza la descarga del contenedor. Cuando finalice, automáticamente desplegara el contenedor y en el Docker Desktop nos aparecerá arrancado.

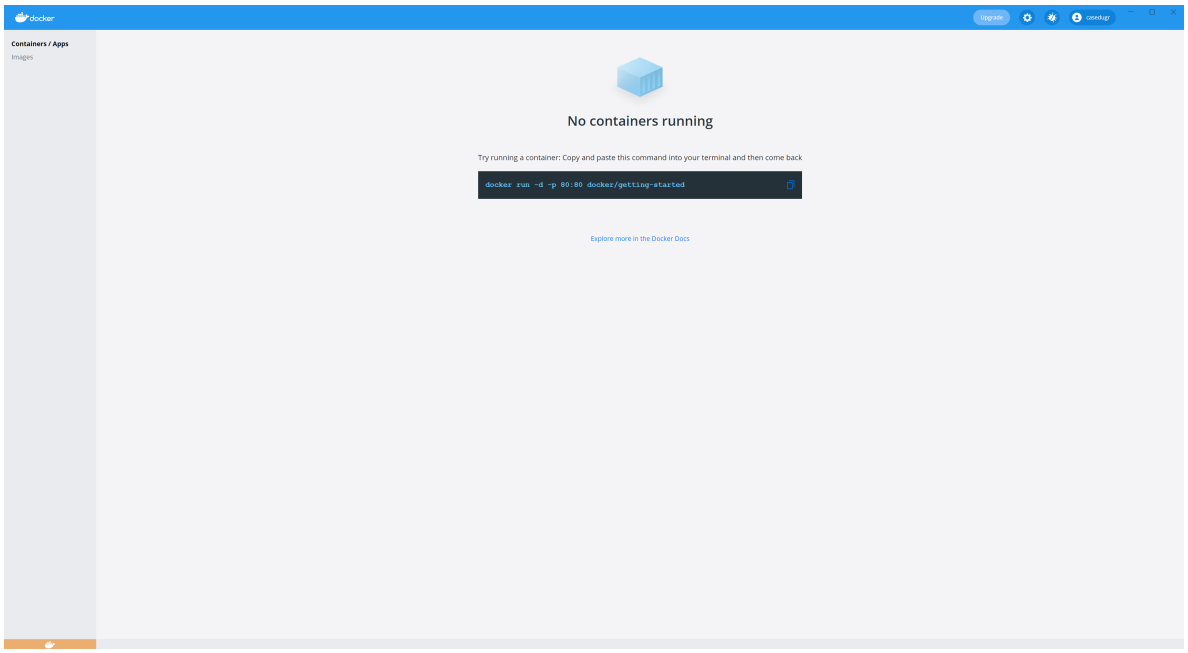

Figura 4: **Pantalla principal de Docker Desktop.**

```
C:\Users\dacac>docker run -e PASSWORD=password -p 8787:8787 casedugr/knowseq
Unable to find image 'casedugr/knowseq:latest' locally
latest: Pulling from casedugr/knowseq
a4a2a29f9ba4: Downloading [=======================================>           ]  22.93MB/28.56MB
127c9761dcba: Download complete
d13bf203e905: Download complete
4039240d2e0b: Download complete
2d7ea8bbfdbf: Download complete
857718e0a33c: Waiting
adbba7016584: Waiting
b8d21fca149f: Waiting
433d77cb45e8: Waiting
580f03cd1fbf: Waiting
b13c6974238b: Waiting
8b721174fdd4: Waiting
7ef04a439d3a: Waiting
fdb573142915: Waiting
4f26a39bc1e3: Waiting
```

Figura 5: **Comando y proceso de descarga del contenedor de KnowSeq.**

5.3. Usando KnowSeq a través del RStudio server desplegado

Una vez instalado y arrancado independientemente del Sistema Operativo utilizado, para tener acceso al RStudio server desplegado tan solo se tiene que utilizar la URL `http://localhost:8787` en cualquier navegador web. Al acceder a dicha dirección, automáticamente se tiene que mostrar la pantalla de login del RStudio desplegado tal y como se ve en la Figura 6.

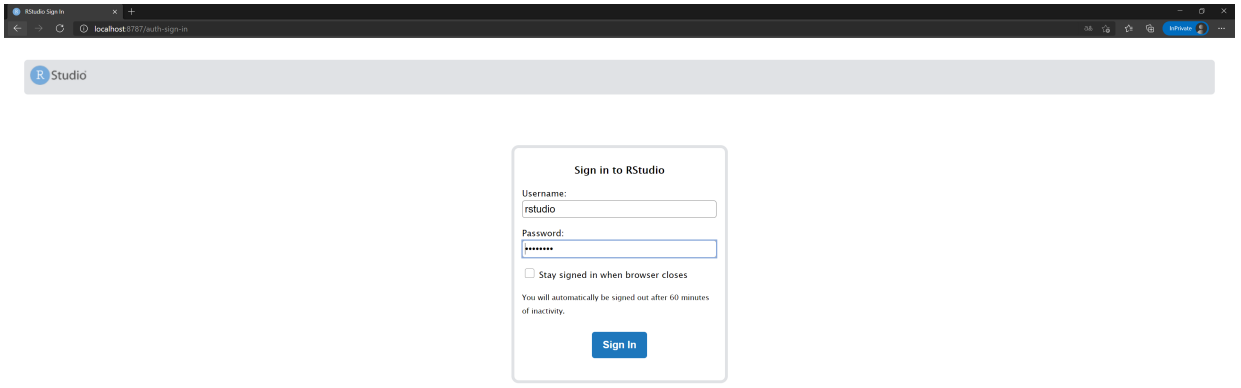

Figura 6: **Pantalla de login del Rstudio Server desplegado.**

Finalmente, si el login es correcto, entraremos a la interfaz normal de RStudio donde tenemos ya precargados diversos elementos como son un conjunto de datos, un script de ejemplo de KnowSeq y un CSV con información de las muestras.

6. Análisis de muestras de Breast Cancer con KnowSeq

Una vez KnowSeq esté disponible ya sea en local o mediante el despliegue del contenedor Docker, se va a realizar el análisis de expresión diferencial de las muestras de Breast cancer propuestas y subidas a SWAD. El objetivo principal de este análisis

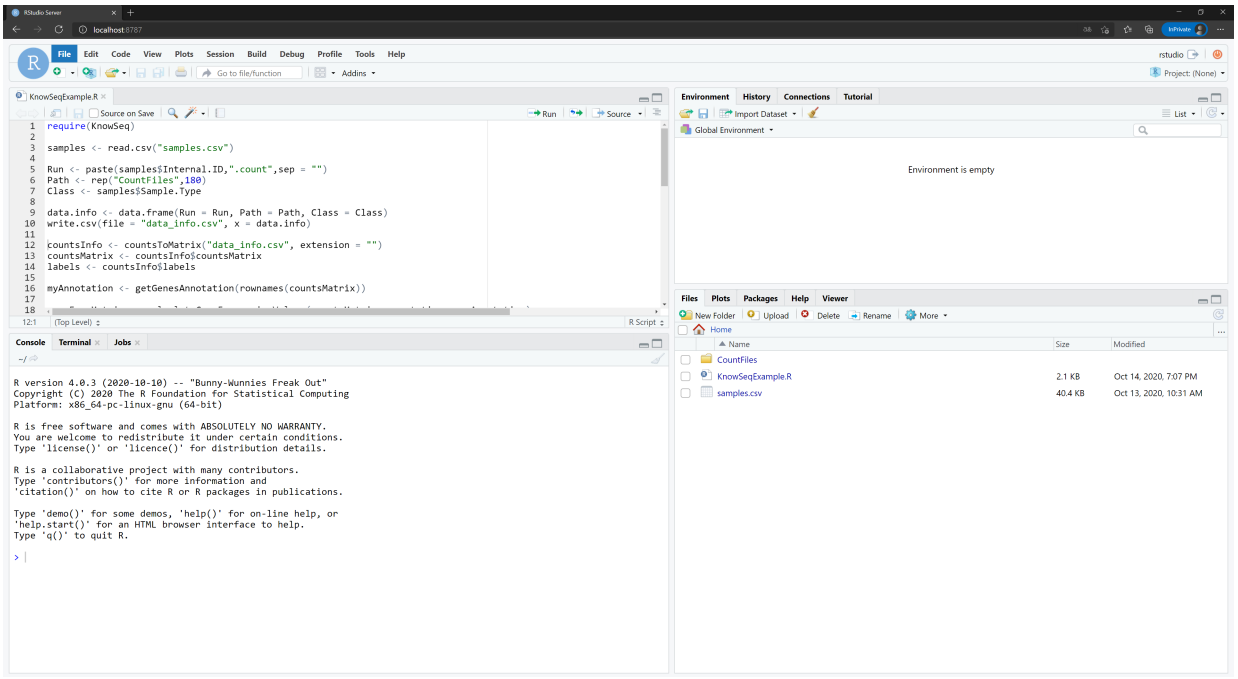

Figura 7: **Interfaz del Rstudio Server con KnowSeq.**

es encontrar un conjunto robusto de biomarcadores (Genes en este caso) con el potencial de discernir muestras de los grupos abordados en el análisis (Primary Tumor y Solid Tissue Normal). Este tipo de muestras son muy comunes cuando se realizan análisis de expresión de gen. Primary Tumor es tejido tumoral extraído directamente del tumor a estudiar. Por otro lado, Solid Tissue Normal es tejido normal adyacente al tumor del cual se ha extraído la muestra de Primary Tumor.

6.1. Preparando los datos

Lo primero que se ha de hacer una vez se ha descargado el dataset del SWAD es necesario descomprimirlo para extraer todos los ficheros count y el csv con la información de las muestras. Esto se puede hacer con una descompresión desde el propio R aprovechando justo después para indicar que el directorio de trabajo va a ser la carpeta descomprimida. Para ello, se ejecuta al siguiente código:

```
# Descomprimimos los datos si lo hacemos en local
unzip("BreastData.zip")

# Listamos los archivos y comprobamos
list.files()

```

```
# Establecemos la carpeta descomprimida como directorio de trabajo
setwd("BreastData")

# Comprobamos que se ha establecido correctamente
list.files()
```

Una vez se tienen los datos correctamente extraídos y el directorio de trabajo fijado, se ha de leer el fichero *BreastSamplesInfo.csv*, el cual contiene toda la información referente a los ficheros counts y los labels de cada muestra. En dicho fichero, la columna File.Name representa el nombre del fichero count para cada uno de los pacientes dados, que se encuentran dentro de la carpeta BreastCountFiles. Finalmente la columna Sample.Type nos indica el label de cada muestra (Primary Tumor o Solid Tissue Normal). El resto de columnas son meramente informativas y no son necesarias para llevar a cabo el análisis. Para más comodidad se crearan variables concretas para cada una de las tres columnas necesarias.

Posteriormente se creará también un DataFrame a partir de las variables que representan dicha información necesaria. Finalmente, se exportará el DataFrame creado a un CSV para tener esa información guardada y lista para los pasos posteriores. Todo ello se puede realizar ejecutando el siguiente código:

```
# Leemos el CSV
samples <- read.csv("BreastSamplesInfo.csv")

# Creamos las variables necesarias
Run <- paste(samples$Internal.ID,".count",sep = "")
Path <- rep("BreastCountFiles",180)
Class <- samples$Sample.Type

# Imprimimos numero de muestras por clase
table(Class)

# Exportamos DataFrame a CSV
data.info <- data.frame(Run = Run, Path = Path, Class = Class)
write.csv(file = "data_info.csv", x = data.info)
```

6.2. Extrayendo los valores de expresión de gen

Una vez se tienen los datos cargados y el CSV creado, es necesario unir todos los counts en una misma matriz para poder así trabajar con ella. Para ello, KnowSeq cuenta con la función *countsToMatrix*, la cual hace un merge de todos los ficheros aunándolos en una misma variable. Además, esta función también nos devolverá los labels de las muestras. Para ello se debe ejecutar el siguiente código:

```
# Cargamos y aunamos los ficheros count
```

```
countsInfo <- countsToMatrix("data_info.csv", extension = "")

# Exportamos tanto la matriz de datos como los labels a nuevas
    variables
countsMatrix <- countsInfo$countsMatrix
labels <- countsInfo$labels
```

Cuando la matriz de counts esta preparada, se puede empezar a trabajar con los datos de los genes propiamente dichos. Cada count está formado por dos columnas, la primera de ellas indica el identificador de cada gen según la notación Ensembl. La segunda columna indica el número de reads de cada gen o las veces que se lee ese gen en el genoma. Para el cálculo de los valores de expresión de gen hay que normalizar esos reads en función del contenido de Guanina-Citosina (GC Content) de cada gen. Además hay que cambiar los Ensembl ID de los genes por los Gene Symbols, o el nombre propiamente para cada gen. Por ello, es necesario consultar el gene Symbol y el GC Content. Este proceso puede hacerse mediante la función de KnowSeq *getGenesAnnotation*. Dicha función recibe los Ensembl IDs y el genoma de referencia con el que fueron alineadas las muestras (38 para este estudio). Una vez se tiene eso, mediante la función *calculateGeneExpressionValues* se extraen los valores de expresión de gen. Para llevar a cabo todo ello se debe ejecutar el siguiente código:

```
# Consultamos los Gene Symbols y el GC content de cada gen
myAnnotation <- getGenesAnnotation(rownames(countsMatrix))

# Calculamos los valores de expresion usando la matriz de count y la
    anotacion previamente adquirida
geneExprMatrix <- calculateGeneExpressionValues(countsMatrix,
    annotation = myAnnotation)
```

Lo ideal una vez se consigue la matriz de expresión es realizar un análisis de calidad en busca de posibles outliers. Para ello la función *RNAseqQA* incorpora diferentes métodos de búsqueda de los mismos. Esta función evalúa 3 métodos diferentes de detección de outliers y mediante un Majority Voting elimina aquellas muestras marcadas como outliers por al menos 2 de los 3 métodos. Para hacer el análisis de calidad se ha de ejecutar el siguiente código:

```
# Realizamos el analisis de calidad
QAResults <- RNAseqQA(geneExprMatrix, toRemoval = TRUE, toPNG=FALSE,
    toPDF=FALSE)

qualityMatrix <- QAResults$matrix
qualityLabels <- labels[-which(QAResults$outliers[1]==colnames(
    geneExprMatrix))]
```

Aparte de eliminar los posibles outliers presentes en las muestras, es necesario tratar el efecto batch, una desviación intrínseca a los datos debido a diversos factores en el proceso de adquisición de las muestras. Si se conocen los batches o grupos de batches existentes en los datos, se aplica un algoritmos basado en el teorema de Bayes llamado ComBat. En este caso, como no se conoce si existe siquiera efecto batch entre las muestras, se debe de aplicar otro tipo de algoritmos que buscan posibles variables afectadas por estas desviaciones. Para ello existen dos grandes algoritmos, Surrogate Variable Analysis (SVA) y RUV (Remove Unwanted Variation). KnowSeq implementa ComBat y SVA como métodos de tratamiento de efecto batch mediante la función *batchEffectRemoval*. Para aplicar SVA en este caso, primero hay que crear un modelo que tendrá en cuenta que variables están afectadas para que posteriormente, la función que extrae los genes diferencialmente destacados tenga en cuenta dichas variables surrogadas.

Una vez se ha creado el modelo de SVA, es necesario imponer ciertas restricciones estadísticas para así filtrar los genes que no albergan diferencia alguna entre muestras sanas y tumorales. Para ello, se suele utilizar el Log-Fold Change (LFC) que mide la magnitud de la diferencia de expresión de cada gen en la muestra tumoral con respecto a la sana. Cuanto mayor sea dicha magnitud, más diferencia a nivel de expresión existe. Además si esa magnitud es negativa significa que el gen esta inhibido en la muestra tumoral con respecto a la normal. Por otro lado, si esa magnitud es positiva significa que el gen esta sobre-expresado en la muestra tumoral con respecto a la normal. Para llevar a cabo tanto el modelo del algoritmo SVA como la extracción de genes destacados mediante la función *DEGsExtraction*, hay que ejecutar el siguiente código:

```
# Creamos el modelo SVA de variables surrogadas para tratar el efecto batch
batchMod <- batchEffectRemoval(qualityMatrix, qualityLabels, method = "sva")

dataPlot(qualityMatrix, qualityLabels, mode = "orderedBoxplot")

# Extraemos los genes diferencialmente destacados teniendo en cuenta las correccion mediante SVA
DEGsInfo <- DEGsExtraction(qualityMatrix, qualityLabels, lfc = 3.5, pvalue = 0.001, svaCorrection = TRUE, svaMod = batchMod)

# Extraemos la tabla de estadisticas de los genes diferencialmente expresados, asi como la matriz ya filtrada con dichos genes.
topTable <- DEGsInfo$Table
DEGsMatrix <- DEGsInfo$DEGsMatrix
```

Una vez se tienen los genes diferencialmente expresados, es recomendable realizar una serie de comprobaciones gráficas para comprobar si realmente se observan cambios

significativos entre ambos grupos (Tumor y Normal) mediante la funcion *dataPlot* con distintos modos.

```
# Plotting the expression of the first 12 DEGs separatelly for all the samples
dataPlot(DEGsMatrix[1:12,], labels, mode = "genesBoxplot",toPNG = TRUE, toPDF = TRUE)

# Plotting the heatmap of the first 12 DEGs separatelly for all the samples
dataPlot(DEGsMatrix[1:12,], labels, mode = "heatmap",toPNG = TRUE, toPDF = TRUE)
```

A partir de este punto, se puede diseñar un proceso de aprendizaje máquina que permita evaluar dichos biomarcadores y ver su potencial ante muestra nunca antes vistas.

```
# Se preparan tanto la matriz como los labels
MLMatrix <- t(DEGsMatrix)
MLLabels <- qualityLabels

# Llevamos a cabo un proceso de Seleccion de Caracteristicas
FSRanking <- featureSelection(MLMatrix, MLLabels, mode = "mrmr", vars_selected = colnames(MLMatrix))

# Evaluamos los biomarcadores mediante un proceso de validacion cruzada
knn_trn <- knn_trn(MLMatrix, MLLabels, vars_selected = names(FSRanking))
knn_results <- rbind(knn_trn$accuracyInfo$meanAccuracy, knn_trn$sensitivityInfo$meanSensitivity,knn_trn$specificityInfo$meanSpecificity)
```

Una vez se tienen los resultados, es bueno llevar a cabo una representación gráfica de los mismos.

```
dataPlot(knn_results, MLLabels, legend = c("Mean Accuracy","Mean Sensitivity","Mean Specificity"), mode = "classResults")
dataPlot(knn_trn, MLLabels, mode = "heatmapResults")
dataPlot(knn_results[,1:4], MLLabels, legend = c("Mean Accuracy","Mean Sensitivity","Mean Specificity"), mode = "classResults")

dataPlot(t(MLMatrix[,names(FSRanking[1:3])]), MLLabels, mode = "heatmap")
dataPlot(knn_trn$cfMats[[3]]$table, MLLabels, mode = "confusionMatrix")
dataPlot(t(MLMatrix[,names(FSRanking[1:3])]), MLLabels, mode = "genesBoxplot")
```

Finalmente, para conocer información relevante a nivel biológico de los genes finales, se usaran varias funciones incluidas en KnowSeq para ello.

```
# Cogemos los identificadores ENTREZ
entrezAnnotation <- getGenesAnnotation(names(FSRanking[1:3]),
    attributes = c("external_gene_name","entrezgene_id"), filter = "
    external_gene_name")

# Se descarga informacion sobre los Gene Ontology
GOs <- geneOntologyEnrichment(as.character(entrezAnnotation$entrezgene_
    id), geneType = "ENTREZ_GENE_ID")

# Se descarga informacion sobre los Pathways
pathways <- DEGsToPathways(entrezAnnotation$external_gene_name)

# Se descarga informacion sobre las enfermedades relacionadas
diseases <- DEGsToDiseases(entrezAnnotation$external_gene_name,
    getEvidences = TRUE)
```

Además, todo el estudio se puede hacer de forma automática con un report inteligente incluido en el paquete R, que sirve como informe en formato HTML.

```
# Lanzamos el report automatico
knowseqReport(geneExprMatrix,labels,'knowSeq-report',clasifAlgs=c('knn'
    ), qualityAnalysis = F,
                disease='breast cancer', maxGenes = 12)
```

7. Ejercicio multiclase de Cáncer de Pulmón

Como ejercicio a desarrollar en clase se propone llevar a cabo un estudio de cáncer de pulmón, dados los counts que se encuentran en el fichero Zip LungData del SWAD o dentro del docker de KnowSeq. Junto con los counts se proporciona un script en R que servirá a modo de esqueleto para que los alumnos lo rellenen utilizando la herramienta KnowSeq tal como se vio en el ejemplo guiado de Cáncer de mama. No obstante, se ha de tener en cuenta que habrá pasos extra en los cuales se tendrá que indagar haciendo uso de la documentación de la librería (https://bioconductor.org/packages/release/bioc/manuals/KnowSeq/man/KnowSeq.pdf). Como trabajo opcional, se deja a libre elección del alumnado usar algoritmos diferentes tanto de aprendizaje supervisado como no supervisado. No hay un objetivo único en cuanto a resultados que entregar, el propio objetivo del ejercicio reside en enseñar como varían los resultados en función de los parámetros elegidos según las consideraciones que cada persona toma durante el proceso y exponer al final que aproximación obtiene mejor ratio de clasificación/relación con la enfermedad.

El estudio en cuestión cuenta con tres clases, dos tipos de cáncer de pulmón (Adenocarcinoma y Squamous Cell Carcinoma) y una clase sana (Healthy). La extracción de genes con la capacidad de diferenciar entre los tres estados abordados permitiría

elaborar una firma genética para el problema dado y evaluar biológicamente dichos genes a posteriori.

8. Autoría

Para la reproducción del pipeline utilizado en este trabajo, es necesario la cita de este documento o de los artículos de investigación en los que se ha aplicado previamente:

- Castillo, D., Gálvez, J. M., Herrera, L. J., San Román, B., Rojas, F., & Rojas, I. (2017). Integration of RNA-Seq data with heterogeneous microarray data for breast cancer profiling. BMC bioinformatics, 18(1), 506: https://rdcu.be/bjTwh
- Gálvez, J. M., Castillo, D., Herrera, L. J., San Román, B., Valenzuela, O., Ortuño, F. M., & Rojas, I. (2018). Multiclass classification for skin cancer profiling based on the integration of heterogeneous gene expression series. PloS one, 13(5), e0196836.: https://doi.org/10.1371/journal.pone.0196836
- Castillo, D., Galvez, J. M., Herrera, L. J., Rojas, F., Valenzuela, O., Caba, O., ... & Rojas, I. (2019). Leukemia multiclass assessment and classification from Microarray and RNA-seq technologies integration at gene expression level. PloS one, 14(2). https://doi.org/10.1371/journal.pone.0212127
- Castillo-Secilla, D., Galvez, J.M, Ortuno, F.M., Herrera, L.J. & Rojas, I. KnowSeq R/bioc package: Beyond the traditional RNA-seq pipeline. A breast cancer case study., 08 November 2019, PREPRINT (Version 1) available at Research Square https://doi.org/10.21203/rs.2.16962/v1
- Gálvez, J. M., Castillo, D., Herrera, L. J., Valenzuela, O., Caba, O., Prados, J. C., ... & Rojas, I. (2019). Towards Improving Skin Cancer Diagnosis by Integrating Microarray and RNA-seq Datasets. IEEE Journal of Biomedical and Health Informatics. https://doi.org/10.1109/JBHI.2019.2953978

Referencias

[1] F. Hahne, W. Huber, R. Gentleman and S. Falcon Bioconductor Case Studies, 1st Edition Springer Publishing Company, Incorporated, 2008.

[2] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, et al. Bioconductor: open software development for computational biology and bioinformatics Genome biology 5 (10) (2004) R80.

[3] J. W. Davis, Bioinformatics and computational biology solutions using r and bioconductor, Journal of the American Statistical Association 102 (477).