Herramienta en línea para la realización de análisis de significación biológica con R

Antonio Morell Bennasser
Máster en Bioinformática y Bioestadística
Área 5

Alexandre Sánchez Pla
Alexandre Sánchez Pla

01/2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Herramienta en línea para la realización de análisis de significación biológica con R.*
Nombre del autor:	*Antonio Morell Bennasser*
Nombre del consultor/a:	*Alexandre Sánchez Pla*
Nombre del PRA:	*Alexandre Sánchez Pla*
Fecha de entrega (mm/aaaa):	01/2019
Titulación:	*Máster en Bioinformática y Bioestadística*
Área del Trabajo Final:	*Área 5*
Idioma del trabajo:	*Español*
Palabras clave	*R; Bioconductor; significación biológica; análisis funcional.*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

En la actualidad los análisis de significación biológica han ido cogiendo relevancia. Por esta razón, se ha realizado este proyecto, cuya finalidad es facilitar el acceso a este tipo de análisis mediante una herramienta en línea. Para ello se ha utilizado el lenguaje de programación R. Para implementar los análisis se han utilizado algunos de los paquetes contenidos en Bioconductor y para el desarrollo de la aplicación se ha utilizado el paquete Shiny. En primer lugar, se ha realizado un estudio para decidir que análisis implementar y que herramientas utilizar para llevarlo a cabo. Posteriormente, se ha realizado un diseño de la aplicación, donde se explica de forma detallada como implementar los análisis y que estructura debe tener la aplicación. Finalmente, se ha desarrollado un prototipo de la aplicación. Con este proyecto se ha obtenido un diseño de la aplicación y un prototipo sencillo e intuitivo que, aun sin ser una aplicación completa, permite ver el potencial que tiene. A partir del trabajo realizado en este proyecto, al haber utilizado software libre, la aplicación puede ser mejorada y ampliada, pudiendo llegar a obtenerse un producto de un gran interés que permitiría llevar los análisis de significación biológica a un público amplio gracias al poder y versatilidad que ofrece.

Abstract (in English, 250 words or less):

Nowadays the biological significance analysis had won relevancy. For this reason, it was made this project, with the aim of facilitating the access to this type of analysis with an online tool. For this it was used the programming language R. It was used some Bioconductor packages to implement the analysis and the Shiny package to produce the application. In the beginning it was done a study to decide which analyses to implement and which tools to do them. Then, it was done a design of the application, which explains in detail how to implement the analyses and the application's structure. Finally, it was made a prototype of the application. With this project it was obtained a design of the application and a simple and intuitive prototype that, without being a complete application, allows to see the potential that it has. From the work realized in this project, because of the use of open-source software, the application can be improved and extended, being able to obtain a very interesting product that allows to bring the biological significance analysis to a large public thanks to the power and versatility that it offers.

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación del Trabajo

Los análisis de datos de microarrays como de Next Generation Sequencing (NGS) desde hace algunos años se encuentran en auge debido a la importancia de la información que proporcionan. Con estos análisis se obtienen resultados de genes diferencialmente expresados, que, por si solos, no proporcionan una información que pueda interpretarse fácilmente dentro de un contexto biológico. Estos análisis deben acompañarse de análisis posteriores, concretamente de análisis de significación biológica, que proporcionan las herramientas necesarias para interpretar los resultados de expresión diferencial dotándolos, además, de un contexto biológico.

Existen un gran número de herramientas que permiten realizar análisis de significación biológica, sin embargo, en su gran mayoría, son de pago o el usuario debe tener suficientes conocimientos de programación para poder llevarlos a cabo. Mediante este proyecto se pretende diseñar e implementar una herramienta en línea con R, utilizando el paquete Shiny y algunos de los paquetes de Bioconductor, que permita la realización de análisis de significación biológica de forma fácil e intuitiva, pensada para que cualquier usuario, con o sin conocimientos de programación, pueda utilizarla y de forma totalmente gratuita, hecho que favorecería la investigación en este campo puesto que mejoraría el acceso a los análisis de significación biológica.

Por lo tanto, en este proyecto se pretende obtener un diseño completo y exhaustivo de la herramienta en línea y el desarrollo de esta, que permita mostrar las funcionalidades incluidas.

1.2 Objetivos del Trabajo

- El objetivo principal del proyecto es el diseño de una aplicación o herramienta en línea con R que permita la realización de análisis de significación biológica.

  Dentro de los objetivos específicos del proyecto se incluyen:

  - Definir los análisis que se implementarán en la aplicación, proponiendo los análisis que se crean necesarios para obtener una herramienta suficientemente completa para cubrir los principales análisis de significación biológica y proponiendo, además, algunos análisis complementarios.

  - Definir las herramientas necesarias para desarrollar la aplicación, como son el lenguaje de programación y los paquetes que se utilizarán de dicho lenguaje.

- Detallar de forma exhaustiva como debería ser la aplicación, detallando como se hará la carga de los datos para realizar los análisis, como se implementarán los análisis, que paquetes y funciones se utilizarán para cada tipo de análisis, como se estructurarán los análisis dentro de la aplicación, que paquetes de R/Shiny se utilizarán para la estructura y el aspecto estético y como se espera que sea el resultado final de la aplicación y por último como será la salida de los resultados obtenidos en los análisis.

- Como último objetivo, se pretende desarrollar la aplicación, escribiendo todo el código necesario para obtener la aplicación partiendo del diseño realizado.

1.3 Enfoque y método seguido

Para la obtención de una herramienta en línea para la realización de análisis de significación biológica, se pretende desarrollar un producto nuevo, pero basándose en otros ya existentes y utilizando herramientas ya disponibles. Para ello, en primer lugar, se debe buscar la información sobre los distintos tipos de análisis de significación biológica y determinar cuales son los de mayor interés. Posteriormente es necesario familiarizarse tanto con el entorno de programación (R), como las herramientas necesarias para realizar los análisis deseados (Bioconductor) y como implementarlos en la aplicación (Shiny).

Una vez familiarizado con los distintos campos, se procede a realizar el diseño de la aplicación, en el cual se incluyen, de forma detallada, los análisis implementados con las funciones y paquetes necesarios para realizar cada análisis, la estructura de la aplicación, como deberá verse una vez terminada y la forma en que se mostrarán los resultados de los distintos análisis, así como su descarga.

Por último, se desarrolla la aplicación, escribiendo todo el código necesario para implementar los análisis principales detallados en el diseño.

La razón por la cual se ha decidido realizar un nuevo producto, a partir del lenguaje R y los paquetes contenidos en Bioconductor, es que mediante estas tecnologías se puede obtener un producto de gran calidad, completo y versátil. Además, el paquete de R, Shiny, permite realizar aplicaciones que resultan muy intuitivas y fáciles de utilizar, tanto para personas con conocimientos previos como para personas que no los tienen, y visualmente atractivas. Otro de los beneficios de trabajar con este lenguaje de programación es que se trata de *software* libre, lo que supone que la herramienta desarrollada sea accesible para cualquiera que desee utilizarla y, además, ello permitiría que cualquier interesado pueda añadir funcionalidades, así como mejorar la aplicación existente.

1.4 Planificación del Trabajo

Para la realización del trabajo se ha utilizado un ordenador portátil MacBook Pro 13' de 2012 con un sistema operativo macOS Sierra versión 10.12.6. En cuanto al *software*, se ha utilizado el lenguaje R en su versión 3.5.1 mediante el entorno de desarrollo integrado (EDI) RStudio en su versión 1.1.463.

Los paquetes de R utilizados son: shiny (versión 1.2.0), shinydashboard (versión 0.7.1) y utils (versión 3.5.1). Los paquetes de Bioconductor utilizados son: clusterProfiler (versión 3.8.1), DOSE (versión 3.6.1), GOSemSim (versión 2.6.2), ChIPseeker (versión 1.16.1), pathview (versión 1.20.0), org.Hs.eg.db (versión 3.6.0), org.Mm.eg.db (versión 3.6.0), org.Rn.eg.db (versión 3.6.0), org.Sc.sgd.db (versión 3.6.0), org.Dm.eg.db (versión 3.6.0), org.Dr.eg.db (versión 3.6.0), org.At.eg.db (versión 3.6.0), org.Bt.eg.db (versión 3.6.0), org.Ce.eg.db (versión 3.6.0), org.Gg.eg.db (versión 3.6.0), org.Cf.eg.db (versión 3.6.0), org.Ss.eg.db (versión 3.6.0), org.Mmu.eg.db (versión 3.6.0), org.Eck12.eg.db (versión 3.6.0), org.Ag.eg.db (versión 3.6.0), org.Xl.eg.db (versión 3.6.0), org.Pt.eg.db (versión 3.6.0), org.Pf.plasmo.db (versión 3.6.0), org.EcSakai.eg.db (versión 3.6.0) y TxDb.Hsapiens.UCSC.hg19.knownGene (versión 3.2.2).

Para realizar este proyecto puede utilizarse cualquier ordenador con unas especificaciones y sistema operativo suficientes para poder utilizar la versión de R y RStudio comentadas, y los paquetes necesarios de R y Bioconductor.

Las tareas a realizar son:

1. Búsqueda de la información necesaria para:
    a. Determinar, mediante búsqueda bibliográfica, los diversos análisis, tanto principales como algunos de los análisis complementarios, que se implementarán en la aplicación.
    b. Determinar que paquetes de Bioconductor se utilizarán para implementar los análisis que se crean apropiados en la aplicación.
    c. Determinar que paquetes de R serán necesarios para el desarrollo de la aplicación y como deberá llevarse a cabo.
2. Diseñar la aplicación.
    a. Realizar el diseño de la aplicación de forma exhaustiva explicando los análisis que se implementarán, mediante que paquetes y funciones, y explicando la forma en que se estructurarán.
3. Desarrollo de la herramienta.
    a. Escribir todo el código necesario para el desarrollo de la aplicación, implementando los análisis que se haya decidido añadir, mediante los paquetes que se crean apropiados y utilizando el diseño realizado previamente.

b. Probar la aplicación, utilizando los datos que se crean apropiados para ello. Además, se eliminarán los posibles fallos que pueda tener el código y se mejorará el aspecto estético.

4. Implementar la herramienta en un servidor web.

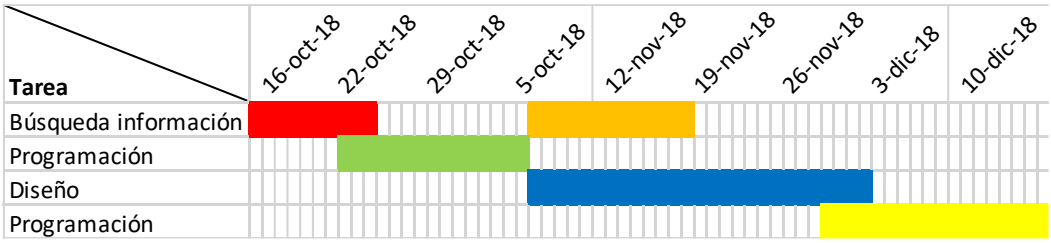

Figura 1. Temporización de las tareas realizadas en el proyecto.

1.5 Breve sumario de productos obtenidos

Se ha obtenido un diseño de la herramienta en línea en el cual se explican los análisis de significación biológica que son de mayor interés para implementarse en la aplicación. En el diseño también se explica que paquetes se ha decidido utilizar, tanto de R como de Bioconductor, para implementar en la aplicación y las funciones de cada paquete que se utilizarán para la realización de los diversos análisis. También se detalla la estructura que debe tomar la aplicación, es decir, el orden en que se añaden los análisis, que instrucciones podrá aplicar el usuario, como deberán cargarse los datos a analizar y como será la salida de los resultados obtenidos a partir de los diferentes análisis.

También se ha obtenido un prototipo sencillo de la aplicación. Esta aplicación desarrollada permite ver las partes comentadas en el diseño, tanto estructurales como estéticas. Aunque el código se desarrolló para realizar los análisis de principal interés, estos se añadieron, pero no da los resultados esperados al realizar los análisis. La aplicación permite la carga de los datos que se quieran analizar. Por lo tanto, se trata de un prototipo que permite ver como debería ser la aplicación deseada, como se realizará la carga de los datos, que análisis se han implementado, de que forma se han incluido en la aplicación y como se mostrarán los resultados.

1.6 Breve descripción de los otros capítulos de la memoria

En los siguientes capítulos se detalla el trabajo realizado. En primer lugar, se describe en que consisten los análisis de significación biológica, así como algunos de los diferentes tipos que se pueden encontrar. Seguidamente se describen algunas de las herramientas que se pueden utilizar para la realización de este tipo de análisis, centrándose en R y Bioconductor. Por último, se detalla el objeto de este proyecto, es decir, la herramienta en línea para los análisis de significación biológica, explicando los análisis que se consideran de

mayor interés, así como algunos análisis complementarios y el diseño de la aplicación.

2. Resto de capítulos

2.1 Análisis de significación biológica

A grandes rasgos, los análisis de datos de *microarray* y los análisis de datos de Next Generation Sequencing (NGS) permiten obtener conjuntos o listas de genes diferencialmente expresados al comparar dos o más grupos experimentales como pude ser un grupo sano o no tratado contra un grupo no sano o tratado [19]. Estos análisis aportan información relevante, sin embargo, resulta de mayor interés acompañar los resultados obtenidos con análisis de significación biológica, los cuales permiten dotar de un contexto biológico a los resultados de expresión diferencial. Un procedimiento común para aproximar una interpretación biológica de los resultados de expresión diferencial consiste en relacionar las listas de genes obtenidas con una o más bases de datos de anotaciones funcionales como por ejemplo la Gene Ontology (GO) o la Kyoto Encyclopedia of Genes and Genomes (KEGG), entre otras [19].

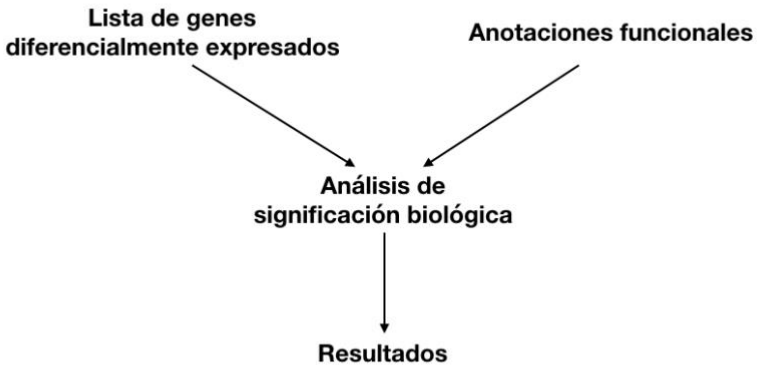

Figura 2. Esquema simple de la relación entre la lista de genes diferencialmente expresados y las anotaciones funcionales.

Las bases de datos de anotaciones proporcionan una información esencial para los análisis de significación biológica. Debido a la naturaleza compleja y distribuida de la investigación biológica, el conocimiento actual está extendido entre varias bases de datos redundantes mantenidas por grupos independientes [21]. Existen varias bases de datos de este tipo como son GO, KEGG, Database for Annotation, Visualization and Integrated Discovery (DAVID), Reactome o

Disease Ontology (DO), entre otras. Todas estas bases de datos tienen en común que proporcionan conocimientos sobre los identificadores de genes y proteínas obtenidos a partir de varios recursos genómicos públicos. Sin embargo, pueden diferir en cuanto a la actualidad de sus datos [23], lo que podría influir en la decisión utilizar una u otra a la hora de realizar nuestros análisis. Estas bases de datos también pueden resultar de mayor o menor interés en función del tipo de información que se pueda extraer de ellas, por ejemplo, el objetivo del GO Consortium es producir un vocabulario dinámico y controlado que pueda aplicarse a todos los eucariotas, para ello se construyen ontologías independientes: Proceso Biológico, Función Molecular y Componente Celular [3] [5]. La base de datos de Reactome proporciona detalles a nivel molecular de la transducción de señal, transporte, replicación de ADN, metabolismo y otros procesos celulares como una red ordenada de transformaciones moleculares, todos ellos procesos biológicos que forman parte del sistema humano [6]. Reactome permite representar diversos procesos del sistema humano, incluyendo rutas del metabolismo intermediario, rutas reguladoras y señales de transducción, así como otros procesos complejos como el ciclo celular [11].

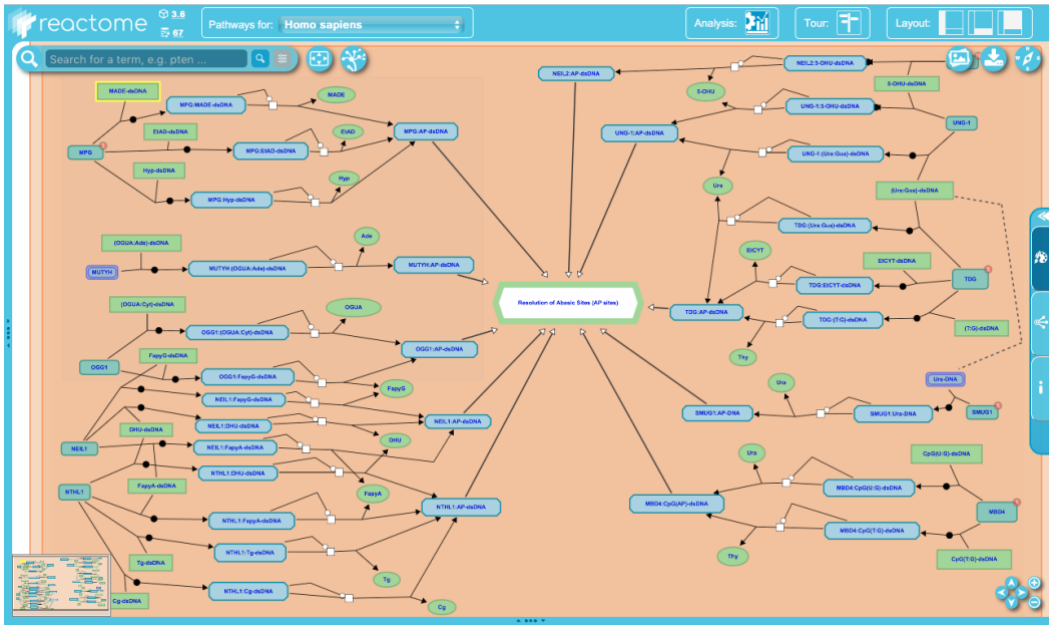

Figura 3. Ruta consultada en Reactome.
En la imagen se muestra una ruta consultada en la base de datos de Reactome, cual se muestran las proteína y moléculas, y sus relaciones e interacciones, implicadas en la reparación de la escisión de bases del ADN.

La base de datos de KEGG proporciona información para el análisis de datos sistemático de las funciones de los genes, relacionando información genómica con información funcional de orden superior. KEGG se divide en tres bases de datos, la base de datos GENES que contiene una colección de catálogos de genes para la totalidad de los genomas secuenciados y otros genomas parciales con anotaciones actualizadas de las funciones de los genes. La base de datos PATHWAY, donde se almacena la información funcional de orden superior y también contiene representaciones gráficas de procesos

celulares. Y, por último, la base de datos LIGAND que contiene la información relacionada con compuestos químicos, enzimas y reacciones enzimáticas [12].

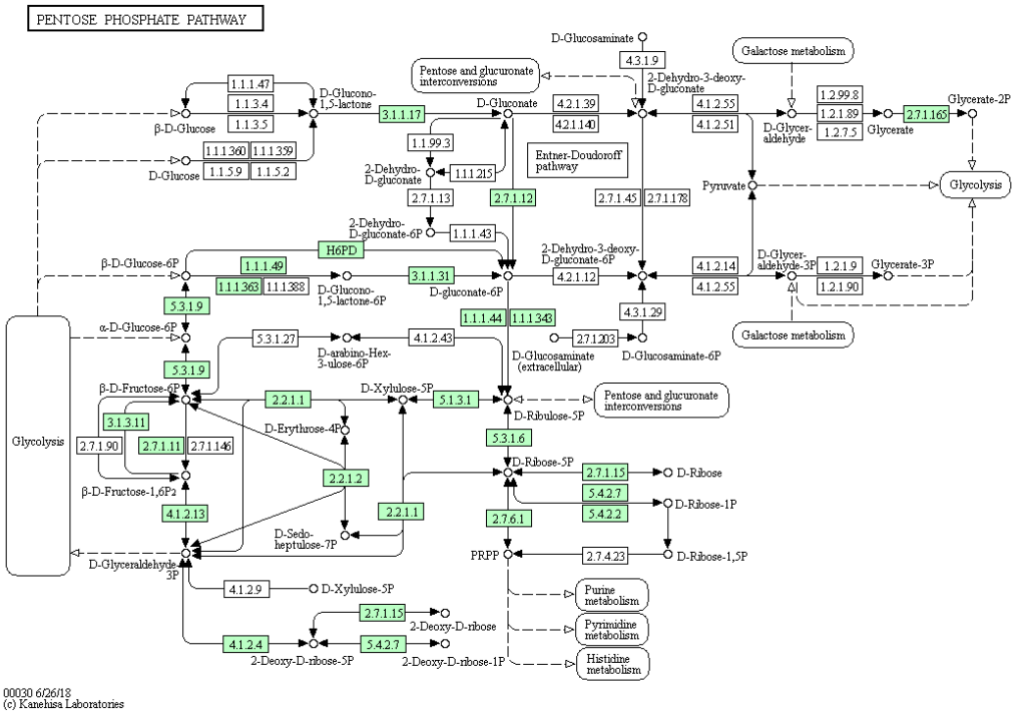

Figura 4. Ruta de las pentosas fosfato consultada en la base de datos de KEGG. En la imagen podemos ver la ruta o vía de las pentosas fosfato para la especie *Homo sapiens*, donde se muestran las proteínas implicadas, las moléculas que intervienen y todas sus relaciones. En los recuadros con números podemos consultar las proteínas implicadas, donde, además, se proporciona toda la información relacionada con esa proteína, como por ejemplo los genes que codifican para la proteína. En la imagen podemos ver perfectamente la interacción entre las tres bases de datos: GENES, PATHWAY y LIGAND.

La base de datos de DAVID recoge e integra diversos identificadores génicos, así como anotaciones disponibles públicamente, que posteriormente son centralizadas por los identificadores internos de DAVID de forma no redundante. La cobertura de anotaciones biológicas y la integración no redundante de IDs génicos de DAVID permite que los ID de los usuarios sean mapeados a través de la base de datos, proporcionando una cobertura exhaustiva de las anotaciones de asociaciones de genes [8]. La base de datos de DO representa la base de un conocimiento exhaustivo sobre enfermedades humanas hereditarias, del desarrollo y adquiridas, que permite conectar datos de variación genética, fenotipos, proteínas y fármacos con enfermedades humanas. Este proyecto fue creado para la clasificación de enfermedades unificando la representación de la gran variedad de terminologías y vocabulario de enfermedades en una ontología que permita la inferencia y razonamiento de las relaciones entre términos y conceptos y optimizando las anotaciones de enfermedades [15] [20].

2.1.1 Tipos de análisis

Existe actualmente un gran número de análisis de significación biológica que se pueden realizar a partir de listas de genes obtenidas, por ejemplo, en análisis de expresión diferencial, que se diferencian en función de los resultados que se quieran obtener, de las bases de datos de anotaciones consultadas o de las bases estadísticas de los mismos. A continuación, se comentan características de algunos de ellos.

La necesidad de realizar análisis funcionales a partir de datos de expresión génica en análisis de microarray y la aparición de GO dio lugar a los análisis de sobre-expresión (Over-Representation Analysis – ORA), los cuales evalúan estadísticamente la fracción de genes en una ruta específica, dentro del conjunto de genes, que muestran cambios de expresión. Las pruebas más utilizadas se basan en distribuciones hipergeométricas, chi-cuadrado o binomial. Sin embargo, estas pruebas presentan algunas limitaciones como, por ejemplo, debido a las diferencias en los estadísticos utilizados estas pruebas consideran el número de genes, únicamente, e ignora cualquier asociación entre ellos [14]. Por otro lado, las aproximaciones FCS (Functional Class Scoring) permiten reducir las limitaciones de los análisis ORA teniendo como hipótesis que grandes cambios en genes individuales pueden tener efectos significativos, pero también que cambios más débiles pero coordinados en conjuntos de genes relacionados funcionalmente pueden tener efectos significativos. Estos métodos, por lo general, utilizan variaciones de una estructura general que consiste en tres pasos: primero, se calcula un estadístico a nivel de gen utilizando las mediciones moleculares del experimento. Esto comprende el cálculo de la expresión diferencial en genes o proteínas individuales. En segundo lugar, los estadísticos para todos los genes en una ruta se agregan en un estadístico simple a nivel de ruta. Este estadístico puede ser multivariante y considerar las interdependencias entre genes, o puede ser univariante e ignorar las interdependencias entre genes. Por último, se evalúa la significación estadística del estadístico a nivel de ruta [14]. Los métodos ORA y FCS consideran únicamente el número de genes en una ruta o la co-expresión de los genes para identificar rutas significativas e ignoran la información adicional disponible en esas bases de datos. Existen métodos basados en topología de rutas (Pathway topology – TP) que se desarrollaron para utilizar la información adicional. Estos métodos son semejantes a los FCS en cuanto al funcionamiento, pero se diferencian de estos en el cálculo de los estadísticos a nivel de gen utilizando la topología de rutas [14].

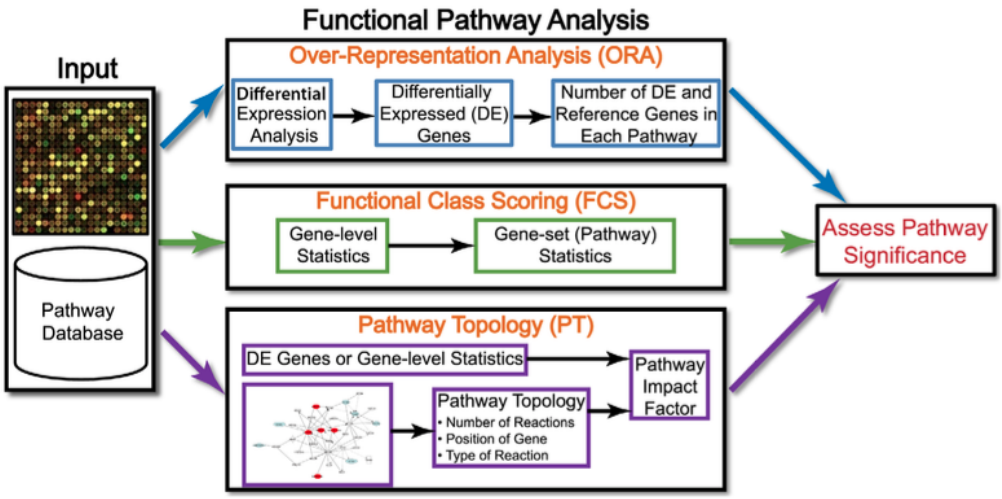

Figura 5. Esquema de los tipos de análisis utilizando datos de expresión génica como ejemplo.
Los datos generados a partir de tecnologías como *microarray*, proteómica o metabolómica, junto con las anotaciones funcionales representan los datos de entrada (input) de los métodos de análisis. Los métodos ORA requieren de una lista de genes diferencialmente expresados como input, los métodos FCS utilizan una matriz de datos completa y los métodos basados en TP utilizan como input, junto con las anotaciones funcionales, el número y tipo de interacciones entre los productos génicos. El resultado de cada análisis en una lista de rutas significativas. La imagen ha sido extraída de la referencia [14].

Dentro de los tres grandes grupos de tipos de análisis mencionados anteriormente, podemos encontrar algunos tipos de análisis muy comunes. Por ejemplo, el Gene Enrichment Analysis (GEA), el cual es un análisis que permite establecer si una categoría dada, por ejemplo, un proceso biológico o una ruta, aparece más (enriquecida) o menos (empobrecida) en la lista o conjunto de genes seleccionados que en la población de genes de la cual se obtuvieron [4] [19]. Otro análisis también extendido es el Gen Set Enrichment Analysis (GSEA), semejante a GEA pero que se diferencia de este en que requiere, además de la lista de genes, una variable numérica para darle un rango, normalmente el p-valor de un test de expresión diferencial [19]. GSEA permite determinar si parte de los genes de un conjunto dado tienden a darse por encima o por debajo de la lista de genes, en cuyo caso el conjunto de genes estará correlacionado con la distinción de clase fenotípica [22]. Dentro de los GSEA podemos diferenciar varios métodos que permiten medir los niveles de expresión de un gran número de genes simultáneamente en dos condiciones fenotípicas diferentes o identificar esos genes que son diferencialmente expresados entre enfermedades, combinando los datos de expresión con conjuntos de datos relacionados con la funcionalidad o la estructura y examinar la sobre- o sub-expresión de estos genes con respecto a genes diferencialmente expresados [1] [10].

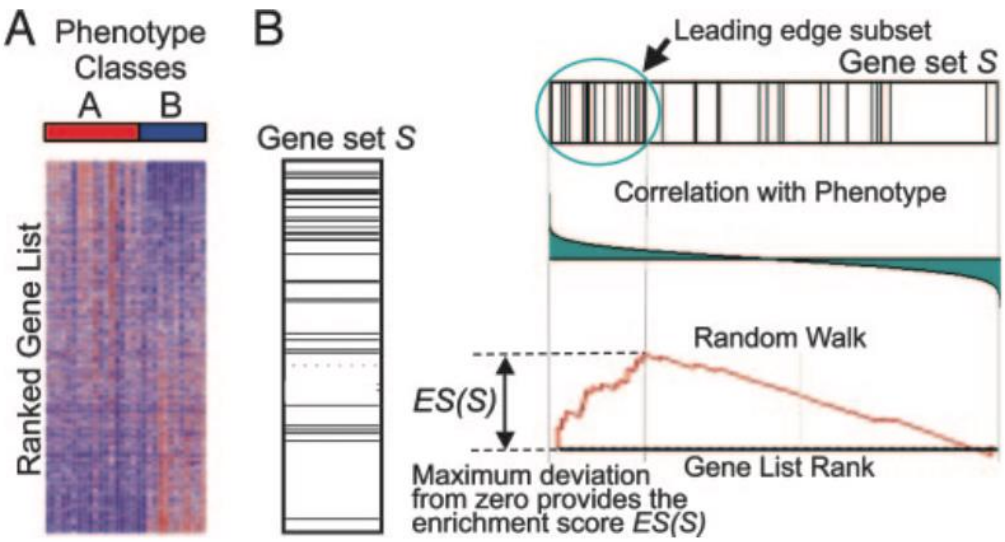

Figura 6. Representación de los resultados de un GSEA.
(A) Heat map donde se muestran los conjuntos de genes la matriz de expresión, ordenados en función de la correlación con las clases fenotípicas. (B) Representación de un conjunto de genes mediante un gráfico de tipo código de barras donde cada barra representa un gen, distribuidos en función de un estadístico; representación de la asociación de estos genes con el fenotipo; y la representación del *enrichment score* que refleja el grado en que el conjunto de genes está sobre-representado, lo que permitirá ver para cual de los grupos o fenotipos los genes del conjunto están sobre-expresados.

Existen otros métodos como el Network Enrichment Analysis (NEA) que permite cuantificar el grupo funcional de genes sobre- o sub-expresado entre los genes vecinos de una red de genes en lugar de en una lista o conjunto de genes diferencialmente expresados [2].

Tool category	Description	Indication and limitation	Sub-type of algorithms	Methods	Example tool
Class I: singular enrichment analysis (SEA)	Enrichment *P*-value is calculated on each term from the pre-selected interesting gene list. Then, enriched terms are listed in a simple linear text format. This strategy is the most traditional algorithm. It is still dominantly used by most of the enrichment analysis tools.	Capable of analyzing any gene list, which could be selected from any high-throughput biological studies/technologies (e.g. Microarray, ChIP-on-CHIP, ChIP-on-sequence, SNP array, EXON array, large scale sequence, etc.). However, the deeper inter-relationships among the terms may not be fully captured in linear format report.	Global reference background	Fisher's exact hypergeometric chi-square binomial	GoStat, GoMiner, GOTM, BinGO, GOtoolBox, GFinder, etc.
			Local reference background	Fisher's Exact hypergeometric chi-square binomial	DAVID, Onto-Express, GARBAN, FatiGO, etc.
			Neural network	Bayesian	BayGO
Class II: gene set enrichment analysis (GSEA)	Entire genes (without pre-selection) and associated experimental values are considered in the enrichment analysis. The unique features of this strategy are: (i) No need to pre-select interesting genes, as opposed to Classes I and II; (ii) Experimental values integrated into *P*-value calculation.	Suitable for pair-wide biological studies (e.g. disease versus control). Currently, may be difficult to be applied to the diverse data structures derived by a complex experimental design and some of the new technologies (e.g. SNP, EXON, Promoter arrays).	Based on ranked gene list	Kolmogorov–Smirnov-like	GSEA, CapMap, etc.
			Based on continuous gene values	*t*-Test permutation *Z*-score	FatiScan, ADGO, ermineJ, PAGE, iGA, GO-Mapper, GOdist, FINA, T-profiler, MetaGP, etc.
Class III: modular enrichment analysis (MEA)	This strategy inherits key spirit of SEA. However, the term–term/gene–gene relationships are considered into enrichment *P*-value calculation. The advantage of this strategy is that term–term/gene–gene relationship might contain unique biological meaning that is not held by a single term or gene. Such network/modular analysis is closer to the nature of biological data structure.	Capable of analyzing any gene lists, which could be selected from any high-throughput biological studies/technologies, like Class I. Emphasis on network relationships during analysis. 'Orphan' gene/term (with little relationships to other genes/terms), that sometimes could be very interesting, too, may be left out from the analysis.	Composite annotations	Measure enrichment on joint terms	ADGO, GeneCodis, ProfCom, etc.
			DAG Structure	Measure enrichment by considering parents-child relationships	topGO, Ontologizer, POSOC, etc.
			Global annotation relationship	Measure term–term global similarity with Kappa Statistics Czekanowski-Dice Pearson's correlation	DAVID, GoToolBox, etc.

Figura 7. Categorización de los análisis de enriquecimiento.
En la figura se muestra una categorización realizada por Huang et al [9], donde encontramos tres clases de análisis: SEA (descrito en este trabajo como GEA), GSEA y MEA. Para cada clase de análisis se ha realizado una breve descripción, se muestran su uso y sus limitaciones, que método estadístico utilizan y algunos ejemplos de herramientas que pueden utilizarse para realizarlos.

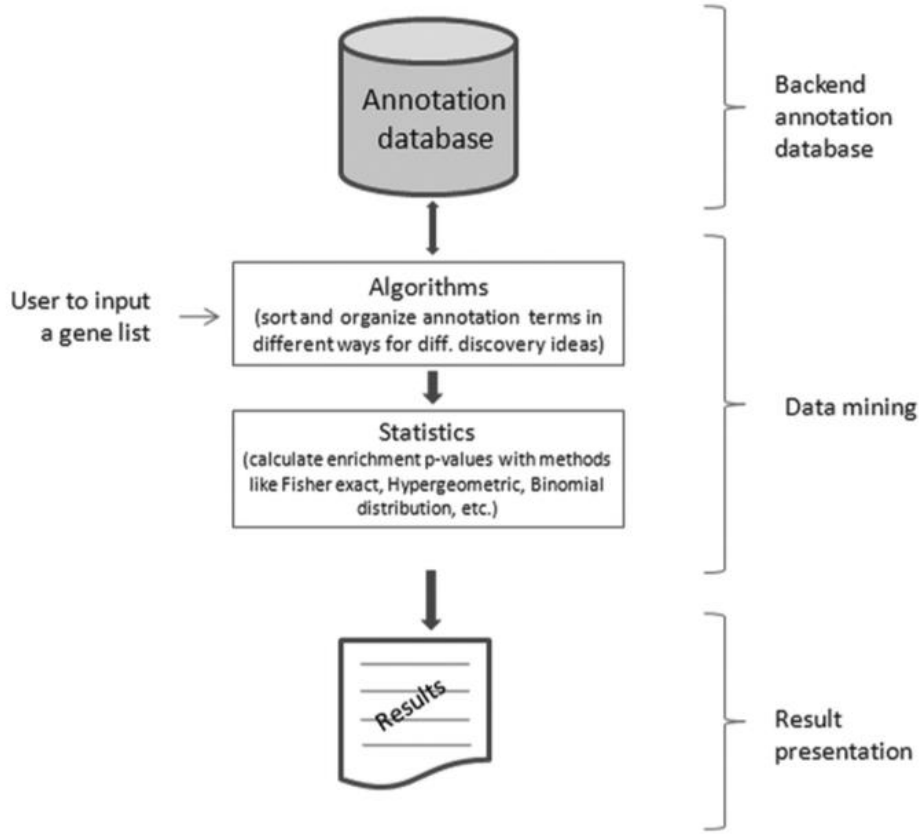

Figura 8. Estructura habitual de las herramientas para anàlisis de enriquecimiento.
La estructura puede variar en función de la herramienta, pero por lo general pueden ser descritas mediante tres capas principales: la base de datos de anotaciones, la minería de los datos y la representación de los resultados [9].

Existen otros tipos de análisis como son los denominados Semantic Similarity Analysis (SSA), que a partir de las anotaciones de GO, permite, dados dos o más términos, cuantificar la similitud entre los aspectos funcionales relacionados entre los términos. Además, estos análisis permiten evaluar la similitud de genes y proteínas basándose en sus anotaciones [7] [16]. Pueden darse similitudes entre términos de GO debido a las relaciones que se dan entre las diferentes ontologías: Función Molecular, Proceso biológico y Componente Celular. También pueden darse estas similitudes debido a la especificidad o generalidad de la ontología descrita, es decir, que un término que indica, por ejemplo, un proceso biológico general estará relacionado con un término que indique un proceso biológico más específico [5].

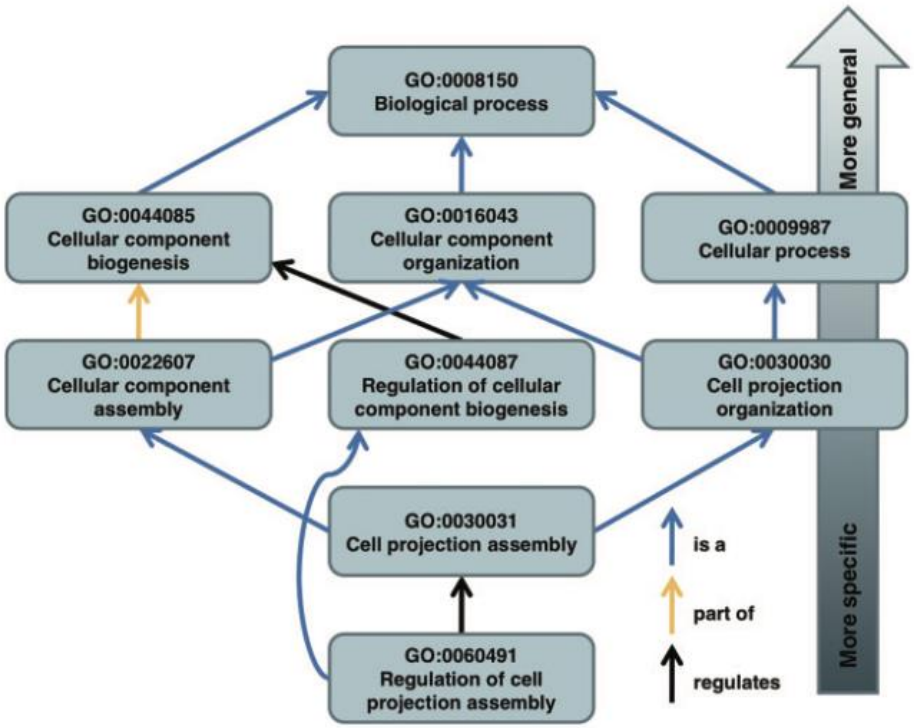

Figura 9. En la imagen se muestran algunas de las relaciones que pueden encontrarse entre distintos términos de GO [5].

2.2 Herramientas para el análisis de significación biológica

Podemos encontrar un gran número de herramientas que permiten realizar análisis de significación biológica, algunas de ellas son de acceso gratuito (Reactome o Bioconductor) y otras no (Ingenuity Pathway Analysis - IPA). También varían en función de los análisis que implementan, las bases de datos de anotaciones que consultan o la plataforma en la cual el usuario debe trabajar [9] [13], algunas de ellas requieren de conocimientos de programación y otras en cambio no. Seguidamente hablaremos de una en concreto, una de las herramientas más versátiles y completas para realizar casi cualquier tipo de análisis.

Enrichment tool name	Year of release	Key statistical method
FunSpec	2002	Hypergeometric
Onto-express	2002	Fisher's exact; hypergeometic; binomial; chi-square
EASE	2003	Fisher's exact (modified as EASE score)
FatiGO/FatiWise/FatiGO+	2003	Fisher's exact
FuncAssociate	2003	Fisher's exact
GARBAN	2003	Hypergeometric
GeneMerge	2003	Hypergeometric
GoMiner	2003	Fisher's exact
MAPPFinder	2003	*Z*-score; hypergeometric
CLENCH	2004	Hypergeometric; chi-square; binomial
GO::TermFinder	2004	hypergeometric
GOAL	2004	Permutation
GOArray	2004	Hypergeometric; *Z*-score; permutation

Figura 10. Ejemplos de herramientas para la realización de análisis de enriquecimiento y los métodos estadísticos que utilizan para llevarlos a cabo [9].

2.2.1 R y Bioconductor

R es un *software* gratuito que permite realizar análisis estadísticos y representaciones gráficas de alto nivel [18] (https://www.r-project.org). Por otro lado, Bioconductor (https://bioconductor.org) es un *software* abierto desarrollado para la realización de tareas de biología computacional y bioinformática, para ser utilizado como una extensión de R mediante numerosos paquetes.

Para este proyecto nos centraremos en el *software* R y en Bioconductor, puesto que ofrecen un gran abanico de paquetes y herramientas para el análisis de significación biológica y para su representación gráfica. Se trata de herramientas que, para usuarios con escasa formación o experiencia en programación, pueden resultar difíciles de manejar, pero, a pesar de ello, son unas de las herramientas más utilizadas ya que resultan de gran utilidad por el gran poder que ofrecen al usuario [19], siendo herramientas, además, muy versátiles y que, como ya se ha comentado, son de acceso libre y por lo tanto cualquier usuario puede acceder a ellas.

Dentro del proyecto de Bioconductor podemos encontrar un gran número de paquetes enfocados a los análisis de significación biológica, algunos de ellos para realizar análisis específicos y otros que permiten realizar más de uno de estos análisis. Podemos encontrar, por ejemplo, los paquetes edgeR, gage, goseq, GOstats o clusterProfiler, con los cuales podemos realizar gran parte de los análisis comentados anteriormente. Estos paquetes se diferencian unos de otros por los análisis que permiten realizar (GEA, GSEA, …), o por el número de análisis distintos que permiten realizar, que puede ser uno o varios, también por los datos a partir de los cuales se realiza el análisis (datos de *microarray* o datos de RNA-seq), las representaciones gráficas que permite realizar o los algoritmos utilizados para realizar los análisis, entre otras cosas.

2.3 Herramienta en línea para los análisis de significación biológica

En este proyecto se pretende desarrollar una herramienta en línea para el análisis de significación biológica con R/Bioconductor y Shiny (del cual hablaremos un poco más adelante). Esta herramienta en línea o aplicación debe permitir realizar los principales análisis de interés, así como otros análisis complementarios, de una forma sencilla e intuitiva, apta para usuarios con o sin conocimientos de programación y de acceso libre. Una de las razones para elegir el *software* libre, es que de esta forma se podrá dar un total acceso a la herramienta, tanto para la realización de los análisis como para mejorar o ampliar la propia herramienta. Seguidamente describiremos el contenido de la aplicación.

2.3.1 Análisis a implementar

En la aplicación, para poder cumplir con los requisitos necesarios que permitan realizar los principales análisis de interés, se implementarán los análisis GEA y GSEA, los cuales permitirán, por un lado, obtener, a partir de una lista de genes, resultados de sobre- o sub-expresión de términos tanto de GO como de KEGG y, por otro lado, dado un conjunto de genes, encontrar si estos tienen relación con las diferencias entre fenotipos. Otro análisis de interés es el análisis de similitudes semánticas o SSA, con el cual se podría estudiar la presencia o ausencia de similitudes entre los aspectos funcionales relacionados entre dos o más términos de GO y además puede servir para medir las

similitudes entre términos de GO para reducir la redundancia de los resultados de GEA de GO.

Por otro lado, con los tres tipos de análisis comentados se realizarían los principales análisis de interés, pero puede ser interesante también implementar otros análisis para desarrollar una aplicación más completa y versátil, que permita al usuario aplicar los análisis necesarios para su propio estudio y poder comparar también los resultados obtenidos mediante un análisis u otro.

Para realizar estos análisis, como ya se ha comentado, se pretende utilizar R junto a Bioconductor, y el principal paquete que se utilizará para implementar los análisis de significación biológica es clusterProfiler (vignette), puesto que se trata de un paquete muy amplio y completo que permite realizar varios tipos de análisis, GEA, GSEA, SSA, DO *analysis*, Reactome *analysis*, DAVID *analysis*, entre otros [25]. Para la representación gráfica de los resultados se utilizará el mismo paquete clusterProfiler y el paquete pathview [17].

Los análisis mencionados hasta ahora son útiles para analizar conjuntos de genes, pero actualmente, con el auge del NGS puede resultar de gran interés poder realizar los análisis GEA y GSEA u otros a partir de este tipo de datos. El paquete clusterProfiler junto con el paquete ChIPseeker (vignette) permiten llevarlos a cabo. Por lo tanto, debido a la importancia que están teniendo actualmente este tipo de estudios sería necesario implementar estos análisis en la aplicación, permitiendo realizar análisis funcionales de enriquecimiento para identificar los roles biológicos predominantes entre genes incorporando el conocimiento proporcionado por las bases de datos de ontologías mencionados (GO, KEGG, DO o Reactome).

2.3.2 Diseño de la aplicación

Para desarrollar la aplicación en línea, se utilizará principalmente el paquete Shiny de R además de Shinydashboard (un paquete que funciona junto a Shiny dándole una apariencia más estética). Se puede encontrar toda la información relacionada con ambos paquetes en: https://shiny.rstudio.com.

En cuanto al diseño de la aplicación, se pretende que esta sea lo más intuitiva posible, para permitir que tanto usuarios con conocimiento previos de R o sin ellos puedan utilizarla por igual. Para ello, esta aplicación aparecerá en una sola página, la cual dispondrá de un pequeño título con el nombre de la aplicación en la parte superior izquierda, y debajo de este título se dispondrán una serie de pestañas que albergarán una o varias subpáginas independientes cada una. En cada pestaña aparecerá el nombre de ésta, para poder distinguir unas de otras y al seleccionarlas se podrá navegar entre los distintos análisis implementados en la aplicación. En la primera pestaña se añadirá un

breve texto descriptivo explicando los diferentes análisis implementados en la aplicación.

Debajo de la pestaña de información habrá una pestaña de carga de datos, en la cual se añadirán las casillas necesarias para cargar los archivos con las listas de genes y los archivos con las poblaciones de genes que se utilizarán en los distintos análisis, y los archivos de anotaciones en caso de que sean necesarios. Los datos se cargarán mediante un botón de acceso a los directorios del dispositivo que se esté utilizando y seleccionándolos manualmente. Estos datos deberán presentarse en formato de texto separado por tabulaciones (.txt). Como los archivos son necesarios para cada tipo de análisis esta página estará ligada a todas las demás, de forma que tan solo sea necesaria una carga de los datos para poder realizar todos los análisis que se deseen.

En las siguientes dos pestañas se añadirán los análisis de GO y KEGG, las cuales dispondrán de subpestañas con los análisis GEA, GSEA y SSA para los análisis de GO, y GEA y GSEA para los análisis de KEGG, siendo estas subpestañas independientes unas de otras, pero ligadas a la carga de los datos (en caso de poder implementar otros análisis, estos se irán añadiendo a nuevas pestañas o subpestañas en función de la base de datos utilizada y el análisis a realizar).

Dentro de la página de cada tipo de análisis el espacio se dividirá en diferentes cajas. Cada caja dispondrá de un título a nivel informativo. En la primera de las cajas se añadirán los diferentes argumentos de cada función del análisis al que pertenezca, con una pequeña etiqueta explicativa para cada argumento. Cada uno de los argumentos dispondrá del valor por omisión dado por defecto en la propia función. De esta forma el usuario podrá decidir que valores u opciones utilizar para cada argumento de cada función a la hora de realizar los análisis deseados, otorgando una mayor libertad al usuario y mayor versatilidad a la aplicación.

Al utilizar el paquete clusterProfiler, la función para el análisis GEA de los términos de GO es enrichGO, por lo tanto, en la subpestaña destinada a este análisis aparecerá una caja con los distintos argumentos de la función enrichGO. De la misma forma se hará para el GSEA de GO, pero en este caso la función de clusterProfiler utilizada será gseGO. Para los GEA y GSEA de los términos de KEGG el diseño será el mismo que para los análisis de GO, pero utilizando las funciones enrichKEGG y gseKEGG respectivamente. Para el SSA de GO también se añadirán los argumentos de la misma forma, en este caso la función utilizada será mgoSim del paquete GOSemSim (vignette), que permite calcular las similitudes semánticas entre dos conjuntos de términos de GO [24]. El argumento semData de la función mgoSim se mantendrá por defecto como: hsGO <- godata(“org.Hs.eg.db”, ont = “MF”), lo que permitirá realizar estos análisis tan solo para la especie *Homo sapiens*. En cada caso, en la caja con los argumentos, en último lugar se añadirá un botón de acción o activación que servirá para ejecutar el análisis.

Dentro de la misma página de cada análisis, donde se dispondrá de la caja con los argumentos de la función, habrá otras cajas en las cuales se añadirán las representaciones de los resultados, por un lado, una caja en la cual se podrá ver la tabla de resultados de la función y, por otro lado, las representaciones gráficas de los resultados. Cada caja de cada representación o tabla tendrá un botón de descarga para que el usuario pueda disponer de los resultados obtenidos en sus análisis de forma externa a la aplicación en formato .csv para las tablas de resultados y en formato PDF para las representaciones gráficas.

La salida de los resultados será, por un lado, mostrándose directamente en pantalla automáticamente, una vez haya finalizado el análisis seleccionado y, por otro lado, como ya se ha comentado, la aplicación permitirá la descarga tanto de la tabla de resultados de cada análisis como de sus representaciones gráficas.

Las representaciones gráficas para mostrar serán las que proporciona el paquete clusterProfiler, mediante la función dotplot, con la cual se obtiene un gráfico de puntos mostrando la función biológica, el recuento de genes asociado y el *p*-valor de ajuste para los resultados de cada análisis. La función emapplot, que permite representar mapas de enriquecimiento para los resultados de GEA y GSEA, donde se pueden ver las relaciones entre las distintas funciones biológicas de los genes. La función cnetplot, que permite representar asociaciones complejas donde un gen puede pertenecer a múltiples categorías de anotación. La función gseaplot, para GSEA, que permite representar la calificación o puntuación de GSEA y su asociación de fenotipo. Por último, mediante el paquete pathview [17] se representarán las vías KEGG (KEGG *pathways*). Todas estas representaciones se añadirán con sus argumentos por defecto, y especificando aquellos que sean necesarios como el argumento en el cual deben añadirse los resultados de los análisis.

Para los demás análisis, en el caso de implementarlos en la aplicación, se añadirán a nuevas pestañas, una para cada análisis. De esta forma, se añadiría una pestaña para DO *analysis*, con el mismo diseño que las anteriores. En este caso presentaría tres subpestañas, una para GEA, otra para GSEA y una última para SSA, análisis que se realizarían mediante las funciones enrichDO (vignette), gseDO (vignette) y geneSim (vignette) respectivamente, del paquete DOSE [26]. Otro análisis complementario interesante es el Reactome analysis, con el cual se pueden realizar GEA y GSEA pero utilizando la base de datos de anotaciones de Reactome, mediante las funciones enrichPathway y gsePathway del paquete ReactomePA (vignette). El DAVID analysis permitiría realizar los análisis de enriquecimiento (GEA) para GO y KEGG, pero utilizando los resultados del servidor web de DAVID mediante la función enrichDAVID. Mediante la adición de estos tres análisis complementarios, podríamos obtener resultados más focalizados con los DO analysis u obtener resultados a partir de otras

herramientas como son Reactome y DAVID, y poder compararlos a los resultados de los análisis principales (GO y KEGG *analysis*). En los tres casos, el diseño de sus respectivas páginas, así como la salida de los resultados deberá ser la misma que para los análisis principales, es decir, respetando la misma estructura y presentación.

Para la realización de los análisis anteriores a partir de datos de NGS, se añadirá una pestaña denominada NGS analysis en la cual se añadirá una subpestaña para la preparación de los datos. Una vez preparados los datos, estos se podrán descargar para utilizarse posteriormente en los análisis ya implementados en la aplicación. Para realizar la preparación de los datos, en primer lugar, se deberá añadir en el código la función seq2gene del paquete ChIPseeker que vincula las regiones genómicas a los genes, y será el objeto vinculado a esta función el que deberá añadirse a las funciones respectivas de cada análisis.

2.3.3 Prototipo de la aplicación

Como ya se ha comentado, se ha realizado un prototipo de la aplicación, siguiendo el diseño realizado previamente. En este capítulo se quiere mostrar como es el resultado obtenido, que contiene la aplicación realizada, que análisis han sido implementados y que partes funcionan y cuales no.

Se puede encontrar el código de la aplicación almacenado en el siguiente enlace: https://github.com/A-Morell/App-TFM. En el enlace se encuentra un documento de lectura con una explicación breve de como ejecutar los archivos de R. En él se incluyen: un archivo para la instalación de los paquetes necesarios, un archivo para cargar los datos de prueba y un archivo con el código de la aplicación realizada.

Al ejecutar el código de la aplicación en RStudio, ésta se carga apareciendo la primera ventana:

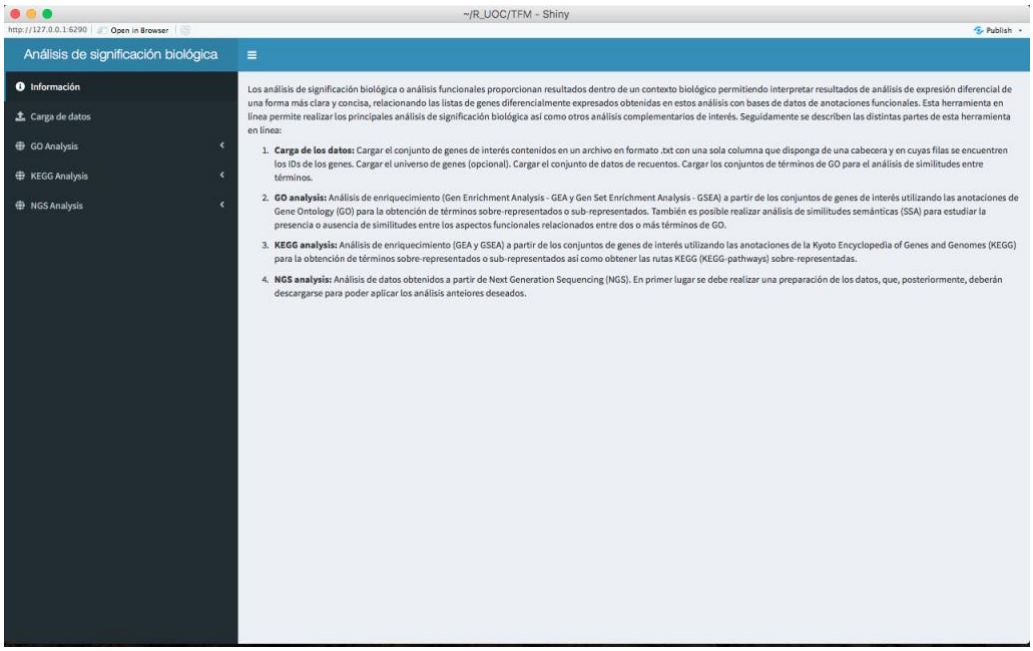

Figura 11. Página de información de la aplicación.

En esta primera ventana se incluye la información de la aplicación, donde se explica brevemente que son los análisis de significación biológica y la utilidad de esta aplicación. Seguidamente, se describen las distintas partes que forman la aplicación. En primer lugar, la carga de datos, donde se explica como deben cargarse y las opciones que presenta. En segundo lugar, los análisis para términos de GO, donde se explica que análisis están implementados y que podemos obtener con ellos. En tercer lugar, encontramos una explicación de los análisis para términos de KEGG, donde se mencionan los análisis implementados y que podemos obtener con ellos. Por último, se explica la forma de realizar análisis para datos de NGS.

A la izquierda de la ventana, señalado en rojo en la imagen, podemos encontrar las diferentes pestañas por las que podemos navegar y seleccionar la carga de datos o los diferentes análisis implementados.

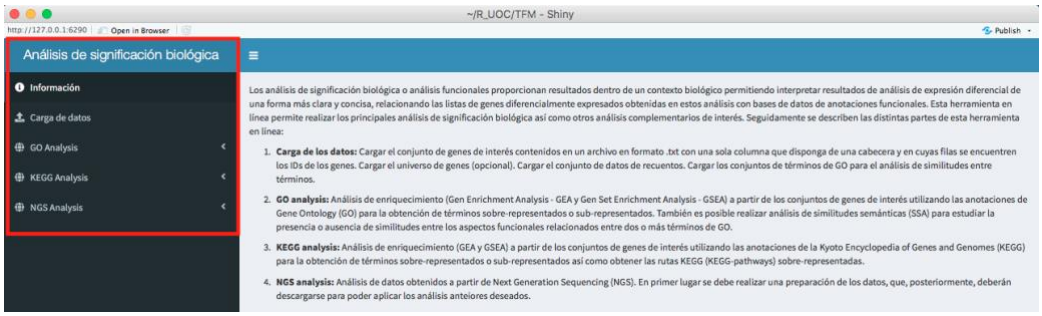

Figura 12. Selección de pestañas.

Si seleccionamos la pestaña de carga de datos nos aparece la siguiente ventana:

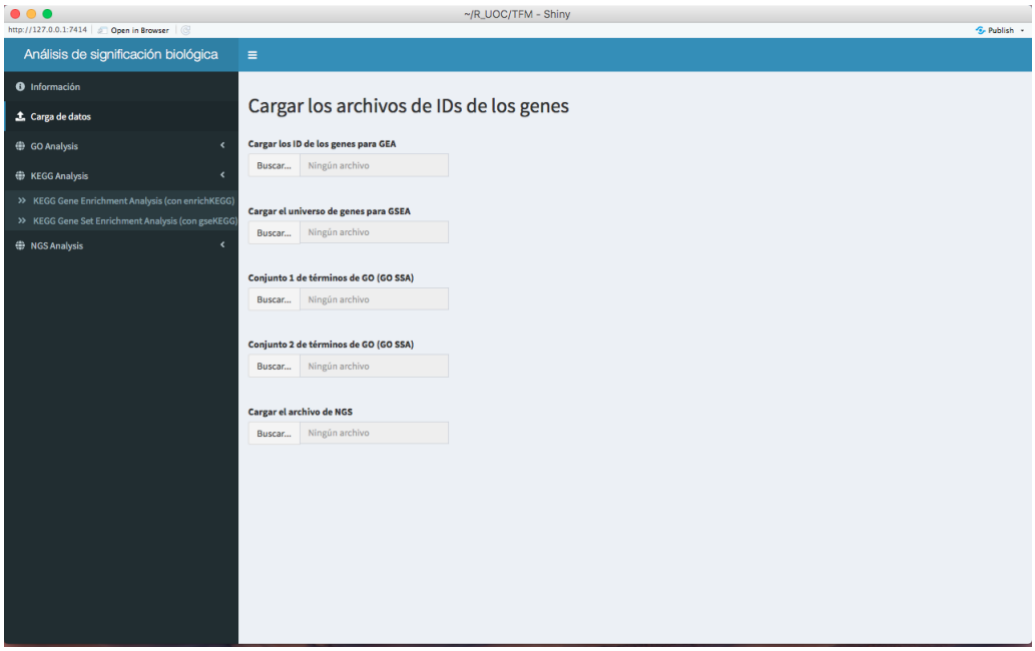

Figura 13. Ventana de carga de datos.

En esta ventana se encuentran las distintas opciones de carga de datos. Se ha incluido un título informativo “Cargar los archivos de IDs de los genes”, y debajo de este título se encuentran las diferentes opciones de carga. En cada caso se debe cargar un documento en formato .txt que debe incluir la lista de genes que queremos analizar. El archivo no debe incluir ni cabecera (*header*) ni nombres de fila.
En primer lugar, encontramos la opción de carga: Cargar los ID de los genes para GEA, estos son los datos que servirán para realizar los análisis de de enriquecimiento implementados, GO GEA y KEGG GEA. En segundo lugar, se encuentra la opción: Cargar el universo de genes para GSEA, que permite cargar un archivo que contenga con los genes para realizar los GSEA para GO y KEGG. Las dos siguientes opciones: Conjunto de términos de GO, permite cargar el archivo con los conjuntos de términos de GO que queremos comparar en un SSA para comprobar las similitudes semánticas. Y, por último, la opción Cargar el archivo de NGS nos permite cargar el archivo que contiene los resultados de un análisis de datos de NGS, es decir, un archivo de picos (*peaks*), que posteriormente deberemos procesar.

Para cargar los datos debemos clicar en “Buscar…”, que automáticamente nos abrirá un enlace al directorio de nuestro ordenador y donde deberemos seleccionar el archivo que queremos cargar.

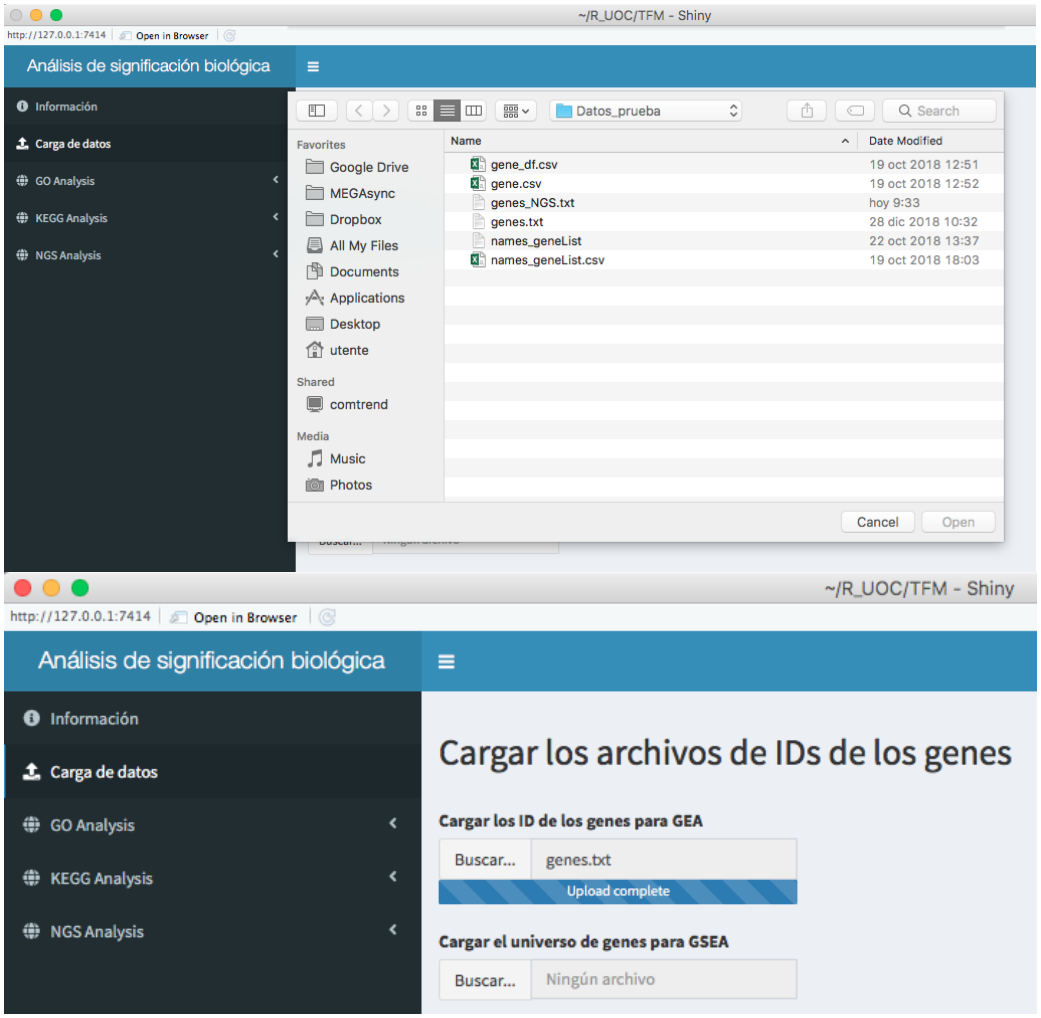

Figura 14. Carga de archivos en la aplicación.

Una vez cargados los datos, podemos realizar los análisis deseados, sin necesidad de cargar cada vez los datos para cada análisis, es decir, que podemos realizar todos los análisis deseados cargando una solo vez los datos.

En el código de la aplicación se encuentran los datos incluidos, de forma que se pueden realizar pruebas con los mismos datos de prueba sin necesidad de cargarlos.

Si nos dirigimos a la pestaña GO Analysis se desplegarán automáticamente las sub-pestañas en las cuales se encuentran los análisis para términos de GO. Si seleccionamos la primera sub-pestaña nos abre la página en la cual se encuentra el análisis GEA para términos de GO.

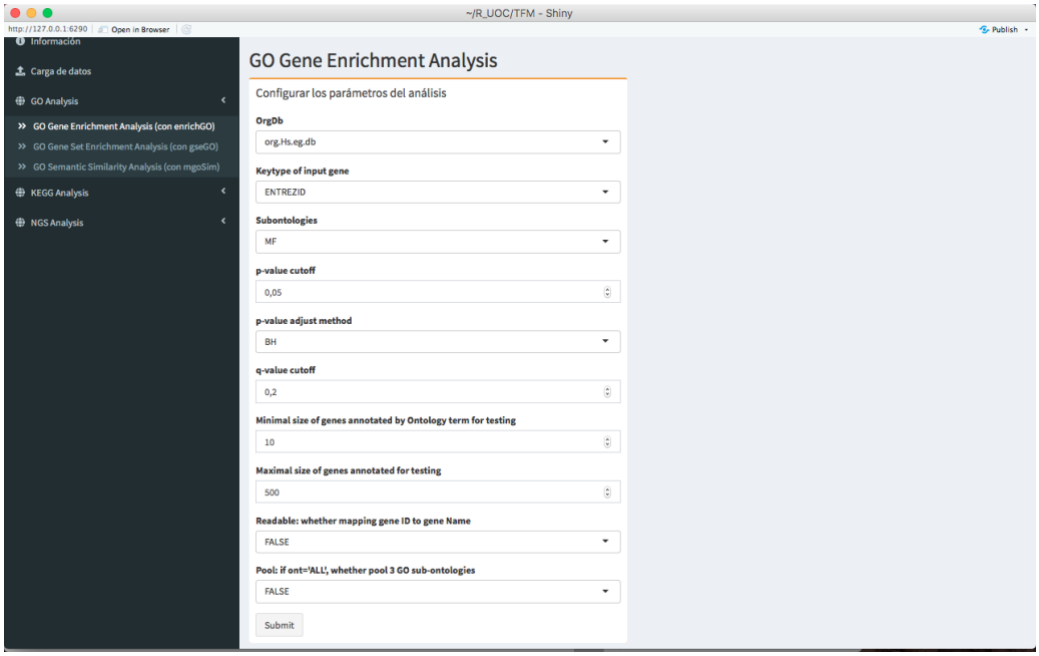

Figura 15. Ventana para el GO Gene Enrichment Analysis.

En esta ventana aparecen las distintas cajas en las que se incluyen los argumentos de la función enrichGO para realizar el GEA y los resultados obtenidos del análisis, tanto una tabla de resultados como las representaciones gráficas. En la imagen anterior se puede ver una de las cajas, concretamente la que contiene los argumentos de la función enrichGO. Al clicar en los diferentes argumentos, se pueden seleccionar o insertar los parámetros que interesen.

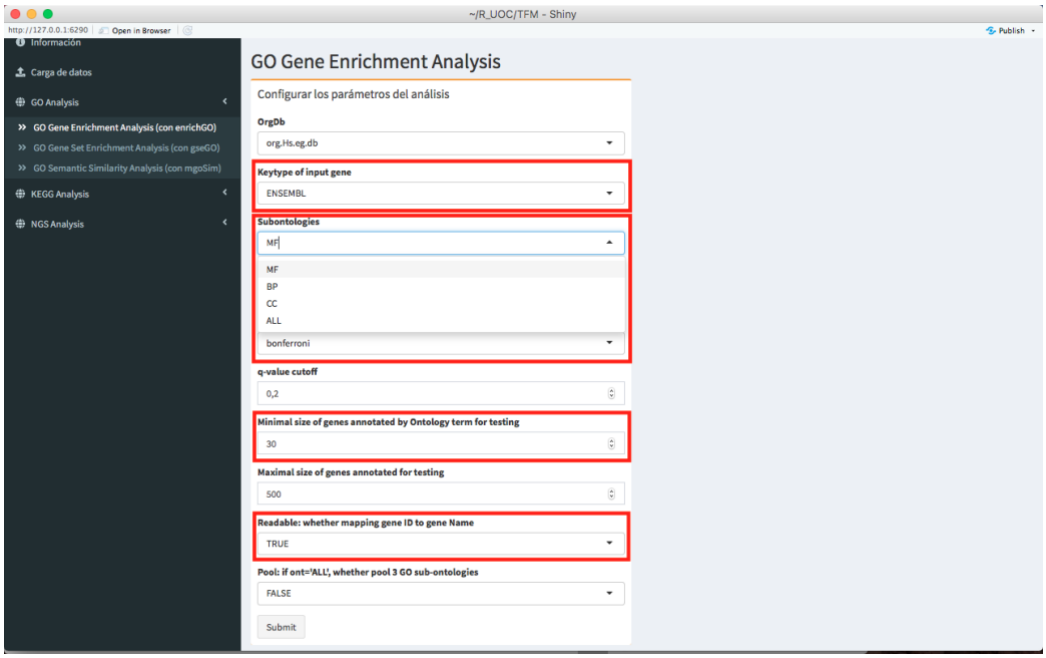

Figura 16. Selección de parámetros para los argumentos de una función.

En la figura 13 se muestran algunos argumentos modificados con respecto a la figura 12, también se muestra la selección del argumento Subontologies. Esta acción, la selección de los parámetros deseados de

cada argumento de una función, se puede realizar para cada tipo de análisis implementado. Si el usuario lo desea puede optar por no modificar los valores de los argumentos, dejando seleccionados los que se encuentran por defecto, es decir, las funciones están programadas de forma que los valores por defecto que se ven en pantalla sean los que se usen en el análisis en caso de que el usuario no los modifique.

Una vez seleccionados los argumentos debe clicarse sobre el botón Submit para ejecutar el análisis.

Si navegamos por la misma ventana, debajo de la caja con los argumentos, se encuentran las cajas con los resultados.

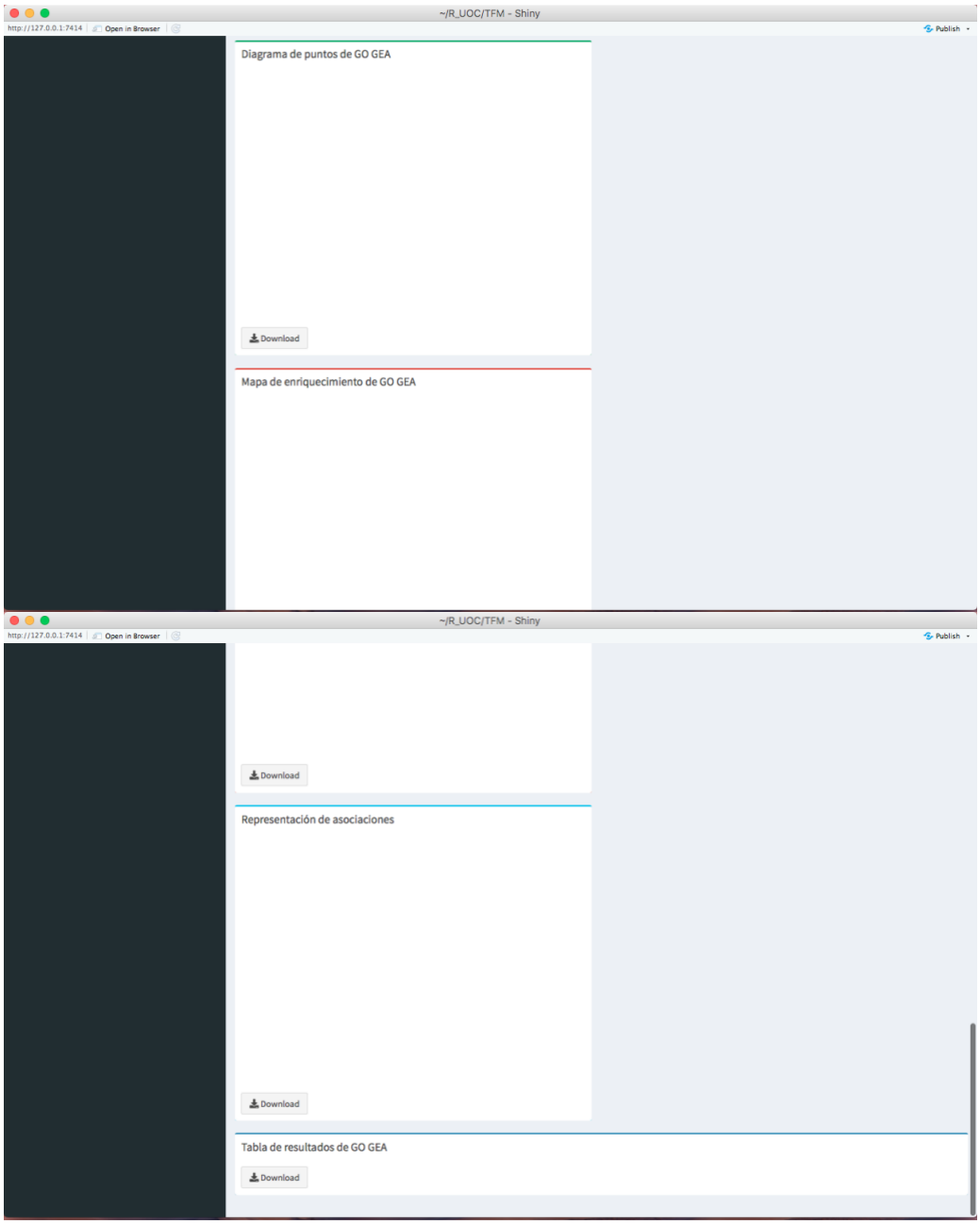

Figura 17. Cajas de resultados del análisis de enriquecimiento para términos de GO.

Para el GO Gene Enrichment Analysis se obtiene una tabla con los resultados del análisis, que contendrá diversas columnas en las que se incluyen los ID de GO de los genes analizados y los *p*-valores obtenidos para cada gen. También se representará un diagrama de puntos, un mapa de enriquecimiento y la representación de las asociaciones. Cada una de las cajas con los resultados presenta un botón de descarga, con el cual pueden descargarse los distintos resultados y representaciones del análisis.

Si se selecciona, en la página principal, la sub-pestaña GO Gen Set Enrichment Analysis, se accede a la ventana para la realización de este análisis, en la cual encontramos las diferentes cajas que contienen los argumentos de la función gseGO, los resultados del análisis y las representaciones gráficas, de la misma forma que se ha mostrado para el análisis anterior. En esta ventana se incluye una representación más que se corresponde con la puntuación de GSEA y la asociación de fenotipo. Para mostrar esta representación es necesario introducir el Gene Set ID y clicar en Submit.

En la pestaña KEGG Analysis se encuentran los análisis para términos de KEGG, siguiendo la misma estructura que en la pestaña GO Analysys.

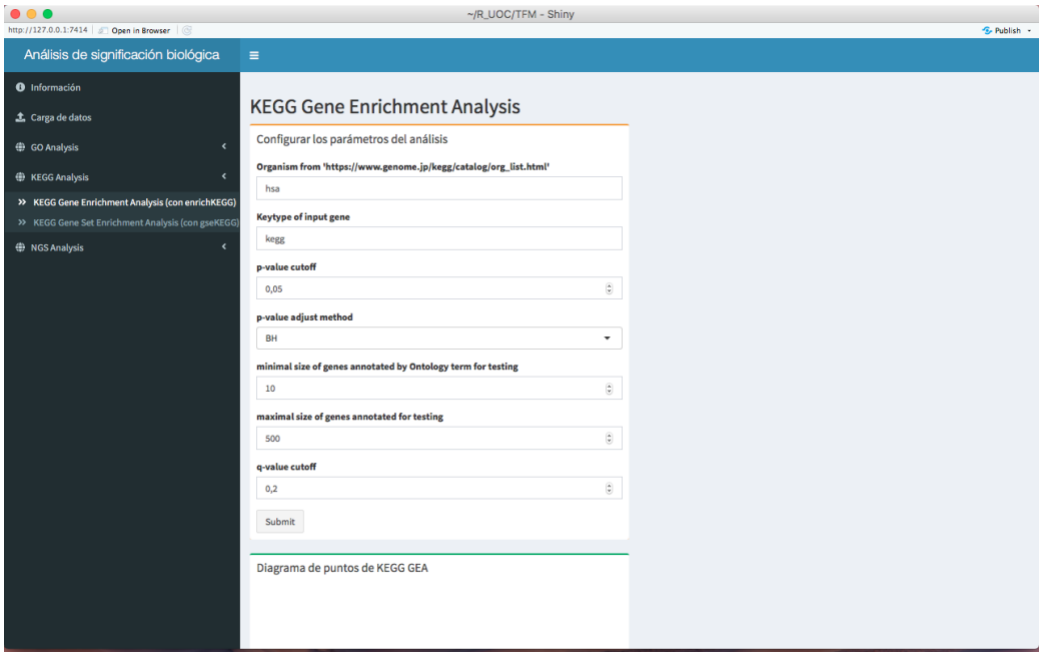

Figura 18. Ventana de KEGG Analysis de la aplicación.

Como se puede apreciar en la figura 15 presenta la misma estructura de cajas, tanto para la selección de argumentos como para los resultados y representaciones. Esta ventana presenta una segunda caja de selección de argumentos, que en este caso sirve para obtener la representación mediante la función pathview. En esta caja, el botón de descarga está ausente y en su lugar se encuentra un botón de Submit, esto se debe a que la función pathview descarga las representaciones de forma automática.

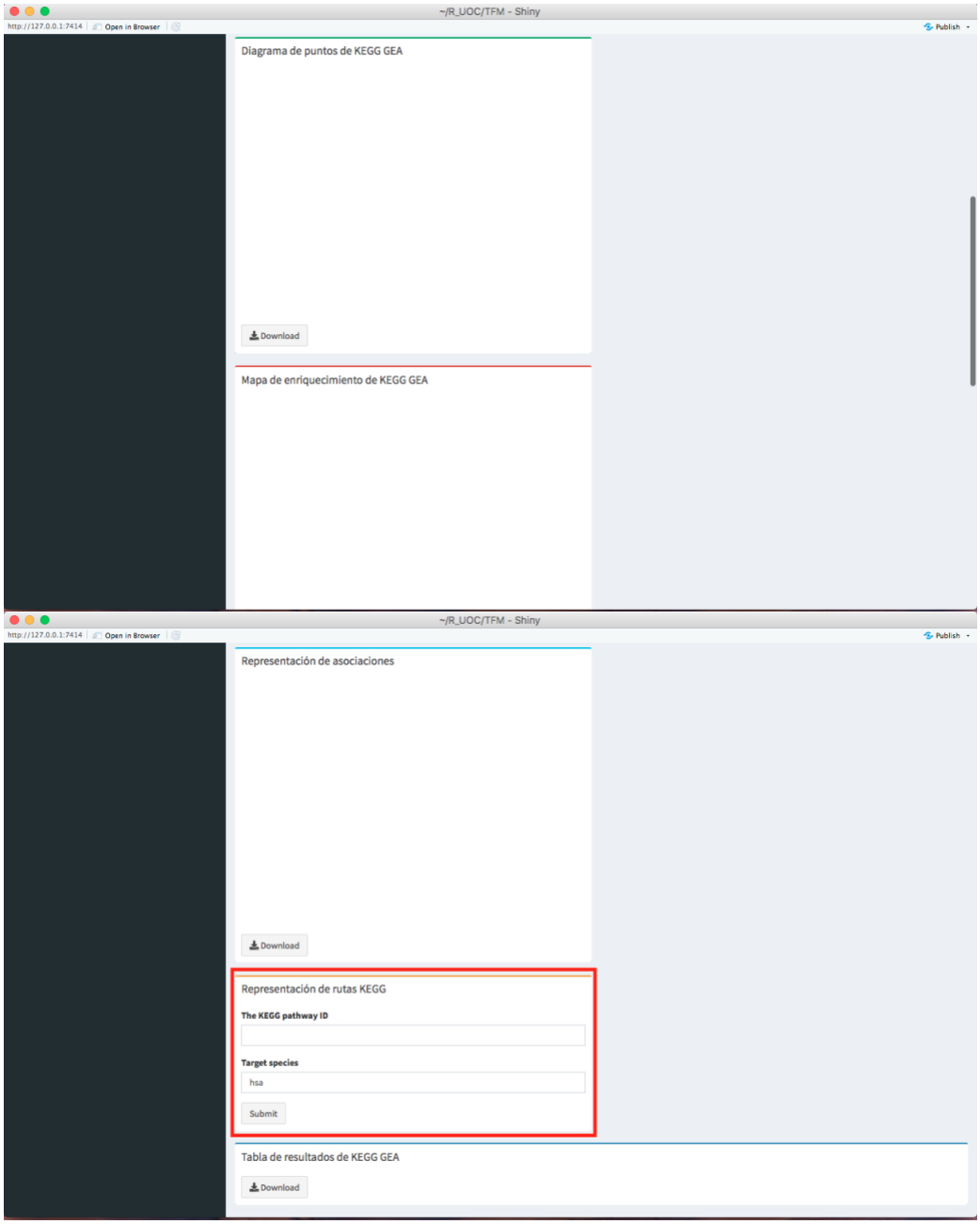

Figura 19. Cajas de la ventana KEGG Gene Enrichment Analysis.

La ventana contenida en la sub-pestaña KEGG Gene Set Enrichment Analysis presenta la caja con los argumentos de la función gseKEGG y las cajas para la tabla de resultados y las representaciones.

Por último, se encuentra la pestaña NGS Analysis con la sub-pestaña Preparación de los datos.

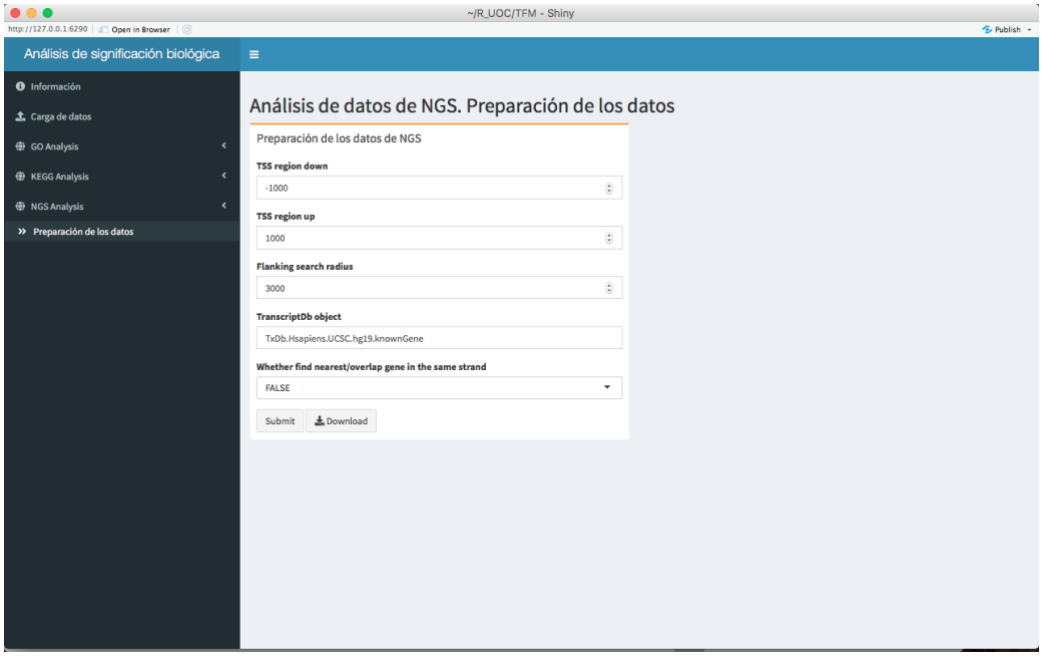

Figura 20. Ventana para los NGS Analysis.

En esta ventana no se encuentran los análisis ni tampoco la representación de los resultados. Aquí se ha incluido una sola caja que contiene los argumentos que podemos modificar de la función seq2gene que permite obtener listas de genes a partir de conjuntos de picos.

Para llevar a cabo los análisis de datos de NGS, en primer lugar, debemos cargar el conjunto de datos que contiene los picos en la opción Cargar el archivo de NGS de la ventana de carga de datos. Una vez cargado el archivo de picos debemos seleccionar los argumentos en la venta Preparación de datos (figura 17) o dejar los que están por defecto y clicar en Submit. Esta acción ejecutará la función seq2gene. Y, finalmente, para obtener el archivo con los genes se debe clicar el botón Download.

Una vez realizada la preparación de los datos y descargado el archivo con los genes, éste deberá cargarse mediante la opción para los análisis GEA y/o mediante la opción para los análisis GSEA, y una vez cargada la lista de genes ya se podrán realizar los análisis deseados, navegando entre los distintos análisis de términos de GO o de términos de KEGG.

Debido a errores en el código, que no se han podido subsanar, la aplicación no proporciona todos los resultados esperados. Si realizamos, por ejemplo, un análisis de enriquecimiento para términos de GO con los datos de prueba, sí aparecen resultados, así como al realizar un análisis KEGG GSEA:

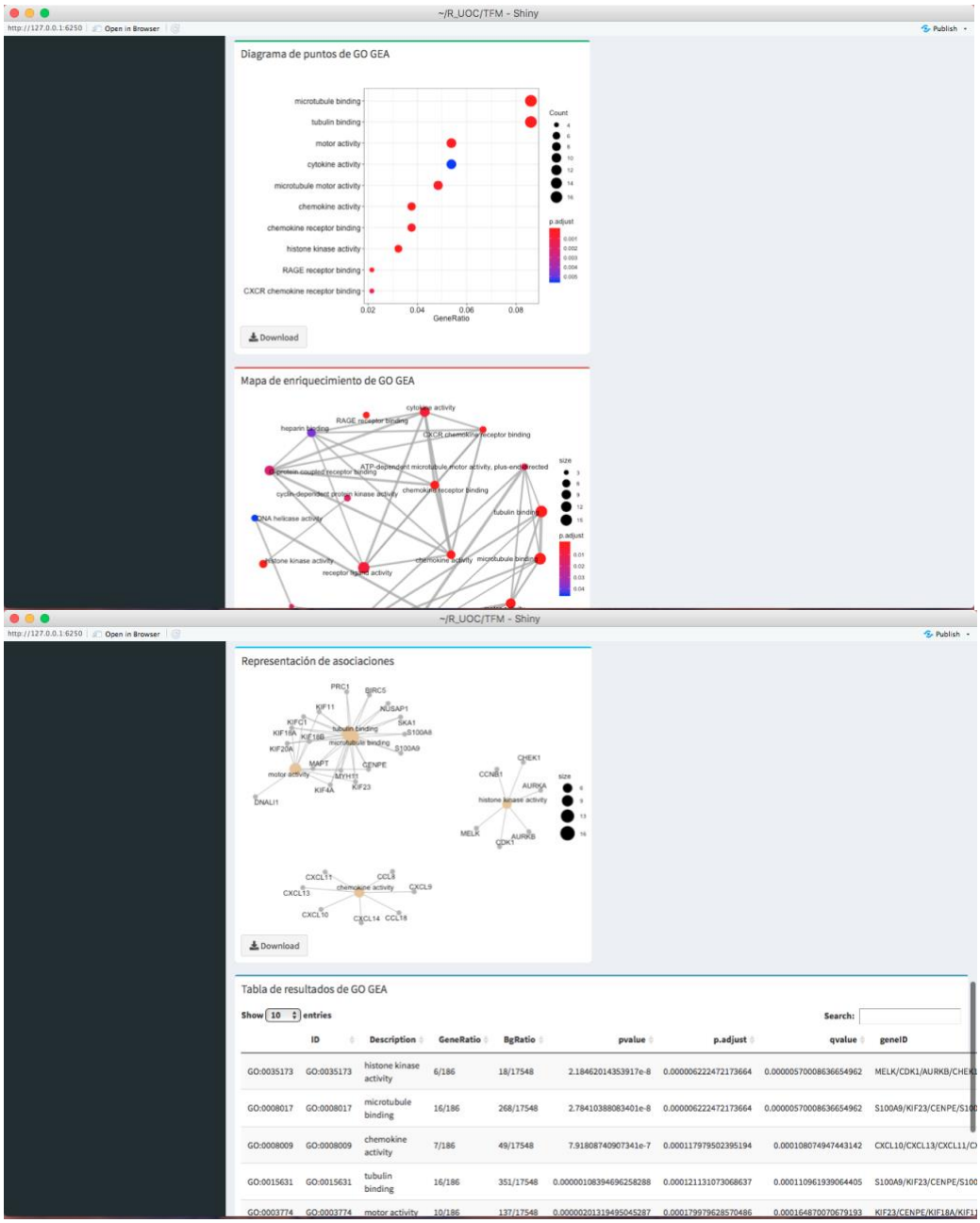

Figura 21. Resultados obtenidos en un análisis de enriquecimiento de GO.
Puede apreciarse en la imagen que la tabla de resultados no se ajusta correctamente a la ventana, y que por esta razón aparecen cortados.

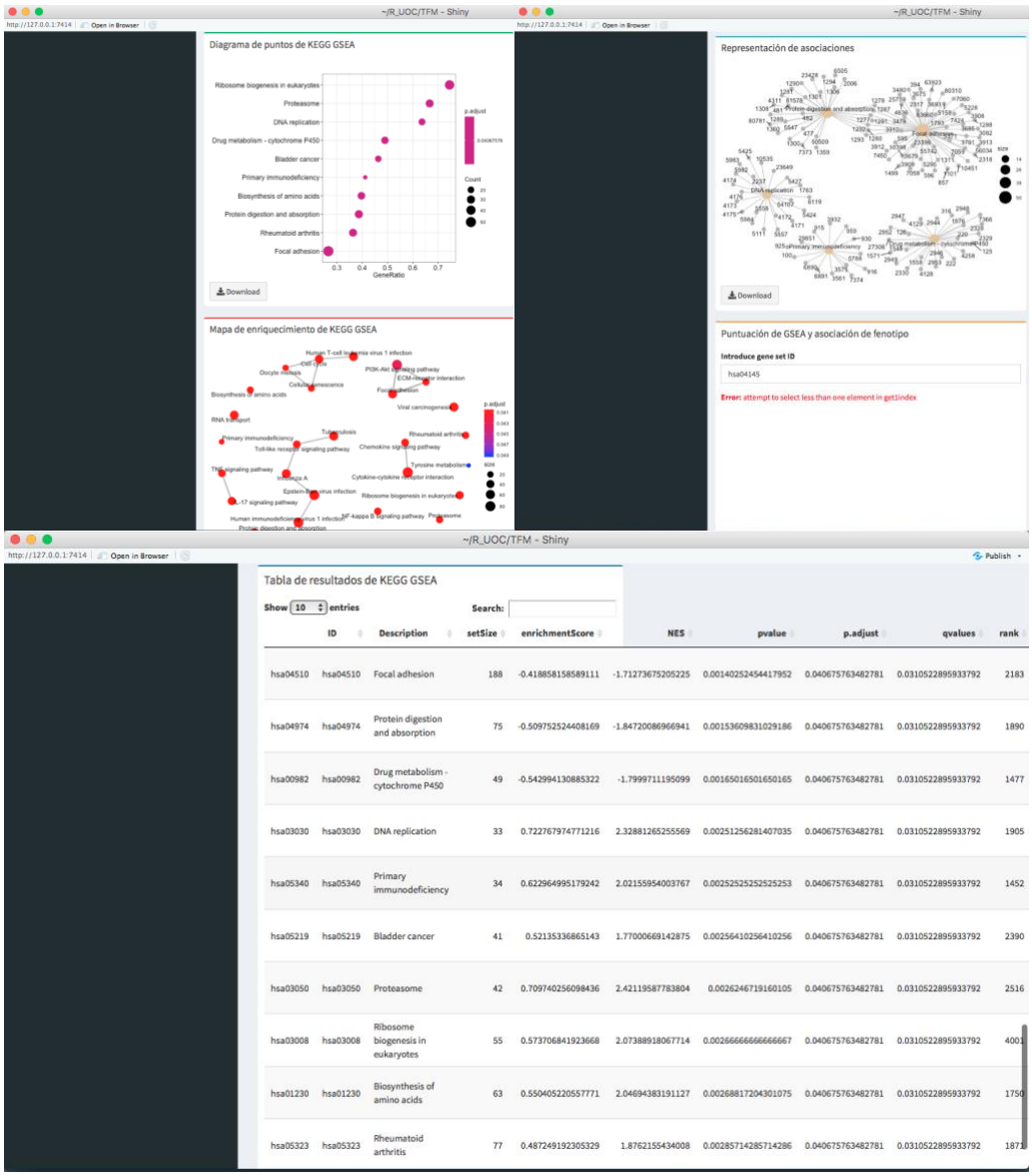

Figura 22. Resultados obtenidos en un análisis GSEA de KEGG.

Al igual que con el análisis de enriquecimiento de GO, la tabla de resultados aparece cortada. Además, debido a algún error de código no se muestra la representación de la puntuación de GSEA y la asociación de fenotipo, aunque sí es posible visualizar la representación descargándola.

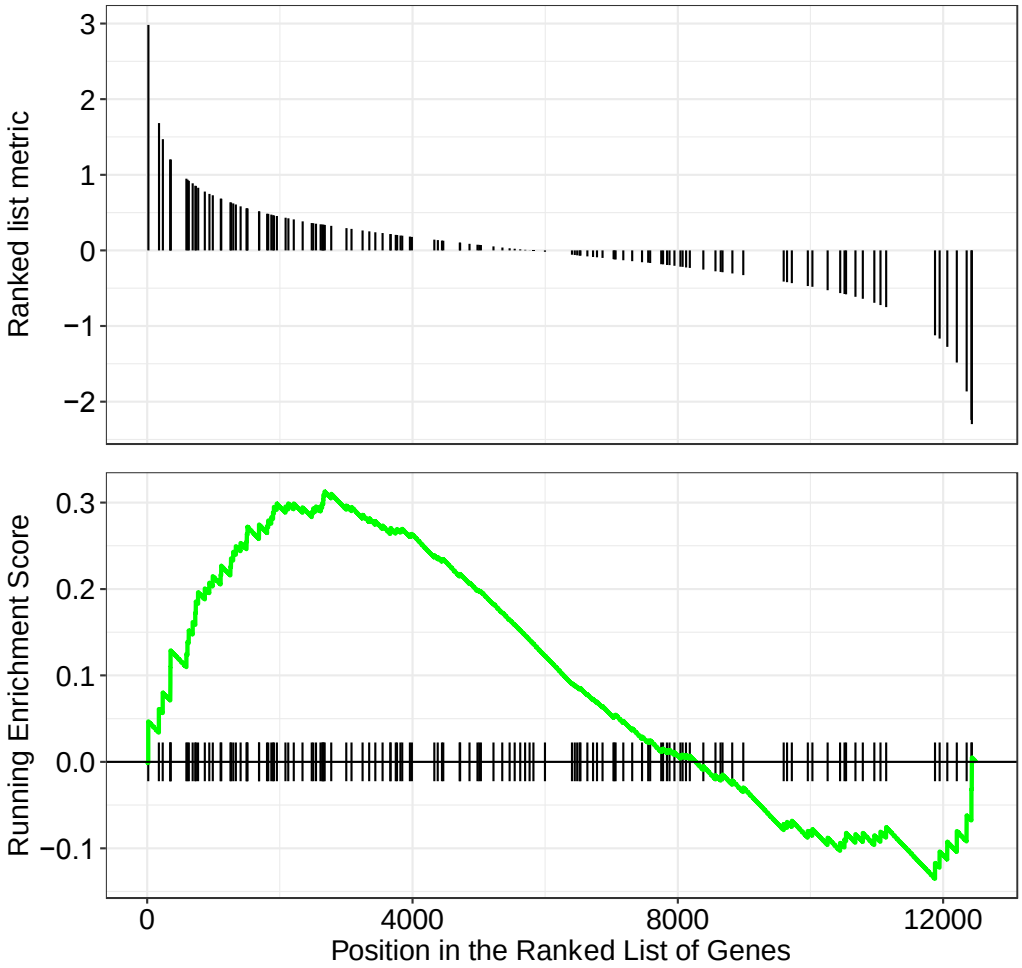

Figura 23. Representación de la puntuación de GSEA y la asociación de fenotipo. Obtenida en el análisis KEGG GSEA mostrado en la figura 22, representación obtenida mediante descarga.

Todos los resultados obtenidos en los dos análisis realizados, tanto la tabla de resultados como las representaciones gráficas, se descargan correctamente al utilizar el botón de descarga situado en sus respectivas cajas. Cabe destacar, que la tabla de resultados se descarga en formato .csv pero los campos no quedan bien separados.

Al intentar realizar los análisis GO GSEA, GO SSA, KEGG GEA y la conversión de datos de picos a genes aparecen distintos errores que impiden realizar dichos análisis. Sin embargo, con los análisis realizados, mostrados en las figuras 22 y 23 podemos ver cuales son los resultados esperados para estos análisis.

A continuación, se muestran los resultados esperados de realizar la representación gráfica con la función pathvisio en un análisis KEGG GEA.

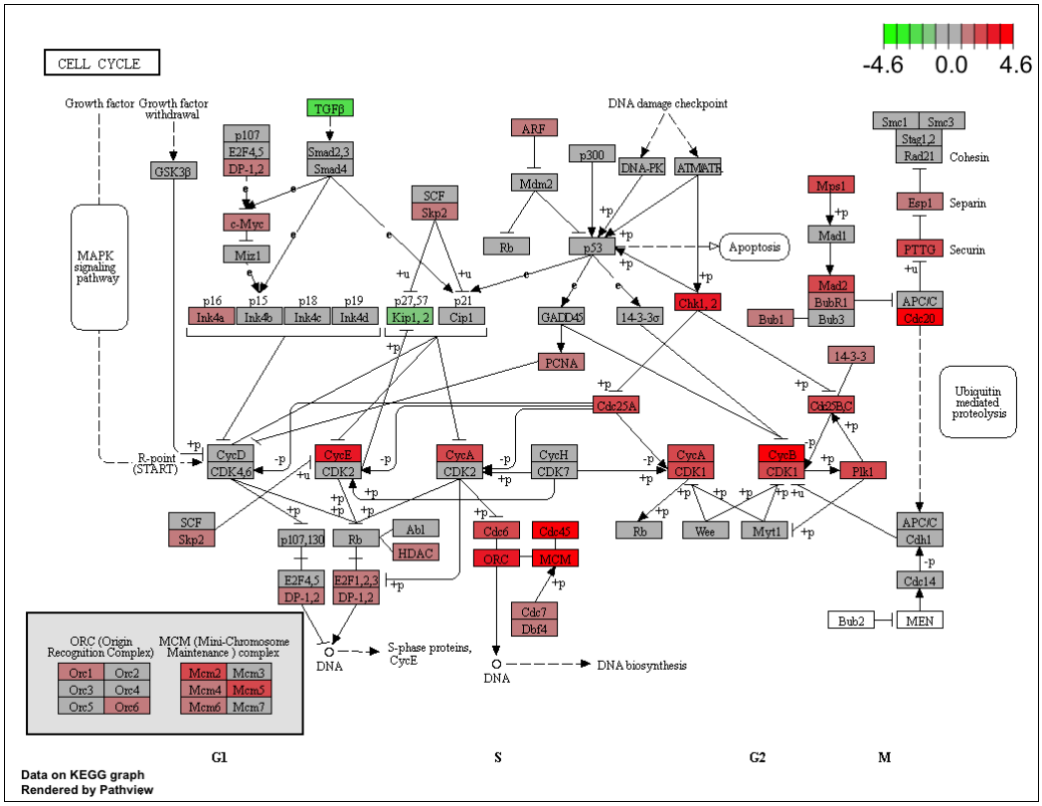

Figura 24. Resultado que se obtiene a partir de la función pathview para un análisis KEGG GSEA.

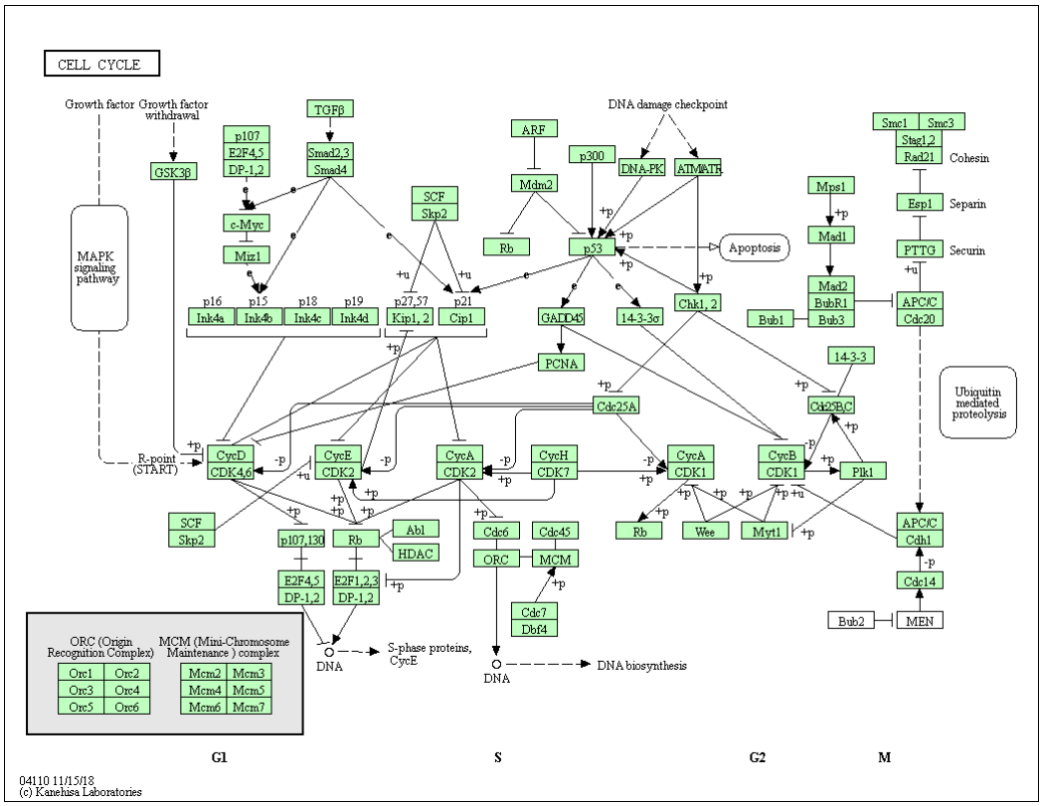

Figura 25. Resultado que se obtiene a partir de la función pathview para un análisis KEGG GSEA.

Se ha buscado la causa de los distintos errores que aparecen y se han solucionado otros, se han intentado eliminar para hacer funcionar correctamente todos los análisis implementados en la aplicación, pero no se ha conseguido. Aun así, ésta permite realizar algunos de los análisis de forma correcta y completa, solo con algún fallo en la representación de resultados.

3. Conclusiones

Se trata de un trabajo complejo y extenso, en el cual no se ha profundizado todo lo posible debido a falta de tiempo y conocimiento, pero mediante el cual se ha podido aprender mucho en el campo de la bioinformática. En este proyecto se ha podido profundizar principalmente en la genómica, gracias al estudio de los análisis de significación biológica, donde se han podido conocer algunos de los posibles análisis que se pueden realizar y que podemos obtener de ellos. Aunque este proyecto abarca una pequeña parte de los análisis existentes hoy en día, se ha podido trabajar con aquellos de mayor interés y que actualmente son los más utilizados.

Por otro lado, se ha podido profundizar y, sobre todo, mejorar en el manejo del lenguaje de programación R y concretamente en el uso del paquete Shiny para el desarrollo de aplicaciones en línea. Con los conocimientos adquiridos se ha conseguido realizar una aplicación sencilla que, aunque incompleta, permite realizar algunas de las funciones propuestas y mostrar las posibilidades y el potencial que tiene ésta.

Por otro lado, también se han mejorado las capacidades para llevar a cabo un proyecto de bioinformática, aprendiendo a realizar una planificación inicial adecuada del proyecto, así como ser capaz, a partir de una idea inicial, de desarrollar un trabajo con un gran potencial.

En cuanto a los objetivos marcados para el proyecto, se han cumplido parcialmente. El primer objetivo marcado era el diseño de la aplicación, el cual sí se ha cumplido, al menos para los análisis que se consideran de principal interés, incluyendo además algunos complementarios. El segundo objetivo marcado era el desarrollo de la propia aplicación diseñada previamente. En este caso, el objetivo no se ha cumplido en su totalidad, puesto que tan solo se ha obtenido un prototipo que solo permite ver algunas de las funcionalidades incluidas en el diseño y no permite realizar los análisis propuestos, puesto que, aunque se ha escrito el código con la idea de que puedan realizarse, no se obtienen los resultados esperados. La razón por la cual no se ha alcanzado el objetivo del desarrollo de la aplicación es debido a una mala planificación inicial del proyecto y debido también al escaso conocimiento del autor en la utilización de algunos de los paquetes de R que permiten la realización de aplicaciones como son Shiny o shinydashboard.

Como se ha comentado en el párrafo anterior, se dieron algunos problemas en la planificación al inicio del proyecto que supusieron un cambio en los objetivos iniciales hacia los objetivos finales mencionados en la introducción de esta memoria. En un principio, se inició el proyecto buscando información sobre los análisis de significación biológica y sobre los paquetes de R y Bioconductor necesarios para realizarlos.

Seguidamente se inició a programar la aplicación basando los análisis a implementar en función de los paquetes de Bioconductor escogidos y sin realizar un diseño previo de la aplicación. Más adelante, se tuvo que corregir esta planificación, para acabar siguiendo la propuesta en esta memoria y, desde ese momento, se inició con el diseño de la aplicación para decidir que análisis implementar y posteriormente que paquetes de R y Bioconductor utilizar para implementar los análisis. Este hecho no ha supuesto un retraso notable en el desarrollo del proyecto, puesto que el código escrito de la aplicación, realizado antes del cambio en la planificación, fue de utilidad a la hora de desarrollarla posteriormente al diseño. Este error y el cambio en la planificación se deben a que el autor nunca había trabajado en proyectos en los que se debía desarrollar un *software* o aplicación y, por lo tanto, al inicio de este, se desconocía que procedimiento se debía seguir y como debía ser una planificación adecuada.

A partir del trabajo ya realizado, con una buena planificación y los conocimientos adecuados puede realizarse un trabajo muy interesante y completo. En primer lugar, sería necesario completar la aplicación realizada, eliminando los errores y añadiendo el código necesario para su correcto funcionamiento. Gracias al gran número de paquetes y funciones disponibles en R y Bioconductor y a su gran versatilidad, además de los análisis principales propuestos, pueden añadirse a esta aplicación un gran número de análisis complementarios que permitan a los usuarios realizar los análisis que deseen y que sean adecuados para sus estudios.

En el diseño de la aplicación se han propuesto algunos de los análisis complementarios que podrían ser implementados, pero existen otros como el NEA (sobre el cual se habla en el apartado 2.1.1). También podrían añadirse funciones que permitan realizar los análisis implementados, pero utilizando otras bases de datos de anotaciones no propuestas en el trabajo, como son DisGeNet o la Network of Cancer Genes (NCG) que contienen información sobre conjuntos de genes relacionados o asociados con enfermedades humanas (DisGeNet) o sobre genes asociados al cáncer (NCG).

La aplicación propuesta permite realizar análisis de conjuntos de genes de las especies que tienen disponible un objeto OrgDb en la base de datos de Bioconductor, la cual proporciona el objeto OrgDb para alrededor de 20 especies. El paquete clusterProfiler proporciona funciones para realizar GEA (mediante la función enricher) y GSEA (mediante la función GSEA) utilizando conjuntos de datos en formato *data.frame* que contengan las anotaciones, en lugar de utilizar un objeto OrgDb. Añadiendo esta funcionalidad, permitiría que los usuarios interesados pudiesen utilizar sus propias anotaciones para realizar los análisis.

Otra funcionalidad que proporciona el paquete clusterProfiler y puede resultar interesante es la conversión de los tipos de identificadores de los

genes (*gene* IDs), que permitiría al usuario cambiar los ID de su conjunto de genes u obtener, a partir de sus ID, otro tipo de identificador. Esto permitiría al usuario, en caso de disponer de unos ID no soportados por la aplicación, poder convertirlos fácilmente para poder utilizar los análisis sin problemas. Esto puede realizarse mediante la función bitr contenida en el paquete clusterProfiler.

Como ya se ha comentado, se trata de una aplicación con un gran potencial que puede solucionar un problema de accesibilidad a este tipo de herramientas, permitiendo realizar análisis de significación biológica a cualquier usuario sin ningún tipo de restricción. Además, esta aplicación no se ve limitada por el tiempo, es decir, que al ser desarrollada en *software* libre puede ir evolucionando en el modo en que lo haga el *software* en que ha sido desarrollada y los paquetes que se han utilizado tanto para realizar los análisis como para realizar la propia aplicación. El hecho de tratarse de un *software* libre también permite que la aplicación sea mejorada y ampliada, pudiendo adaptarse a las necesidades de cada usuario. En definitiva, se trata de un proyecto que pretende facilitar la accesibilidad a los análisis de significación biológica, aportando un producto que, con el tiempo, puede ser capaz de adaptarse a las necesidades de cada usuario proporcionando las herramientas adecuadas para cada tipo de estudio y permitiendo realizar los análisis implementados de forma fácil e intuitiva.

4. Glosario

- Análisis de significación biológica: análisis que permiten dotar de un contexto biológico a los resultados de expresión diferencial. Existen un gran número de ellos que varían en función de las bases estadísticas y en función de las bases de datos consultadas.

- Bioconductor: se trata de un proyecto de *software* libre para el análisis de datos genómicos, basado principalmente en el lenguaje de programación R. Es un repositorio que contienen numerosos paquetes enfocados a la realización de trabajos de bioinformática con R.

- DAVID: del inglés Database for Annotation, Visualization and Integrated Discovery, consiste en una base de datos que proporciona un conjunto de herramientas de anotaciones para la interpretación biológica de grandes conjuntos o listas de genes.

- DO: del inglés Disease Ontology, base de datos de anotaciones de términos relacionados con enfermedades humanas.

- GEA: del inglés Gen Enrichment Análysis, se trata de un tipo de análisis de significación biológica que permite determinar aquellos genes dentro de un conjunto que se encuentran sobre- o sub-expresados.

- GO: Gene Ontology, consiste en una base de datos de anotaciones que permite la interpretación biológica de conjuntos de genes definiendo los conceptos utilizados para describir sus funciones y las relaciones entre estos conceptos.

- GSEA: del inglés Gen Set Enrichment Analysis, es un tipo de análisis de significación biológica que permite identificar clases de genes o proteínas que estén sobre- o sub-expresados dentro de un gran conjunto, y que pueden tener una asociación con determinados fenotipos, por ejemplo, de enfermedades.

- KEGG: Kyoto Encyclopedia of Genes and Genomes, consiste en una base de datos que proporciona los recursos necesarios para la interpretación de las funciones y utilidades del sistema biológico, desde un punto de vista molecular.

- Microarray: se trata de una herramienta utilizada para la detección de la expresión de un gran número de genes al mismo tiempo. Los análisis de microarray permiten detectar las diferencias de expresión entre los genes de estudio.

- NGS: del inglés Next Generation Sequencing, se trata de una tecnología que permite secuenciar genomas de forma rápida y sin

costes demasiado elevados, mejorando enormemente las tecnologías precedentes. Esta tecnología a supuesto una revolución en estudios de genómica.

- Objeto: en la programación orientada a objetos, un objeto es una unidad que presenta un comportamiento determinado por los datos almacenados en él o por las tareas realizables durante su ejecución.

- Paquete (*package*): *software* donde se almacenan recursos que proporcionan algunas funcionalidades como parte de un sistema o *software* mayor.

- R: lenguage de programación especialmente diseñado para análisis estadísticos.

- Reactome: base de datos de rutas (*pathways*) que proporciona herramientas bioinformáticas para el análisis, interpretación y visualización de rutas.

- Ruta (*pathway*): en biología, se habla de rutas para referirse a series de interacciones entre moléculas que ocurren en el interior de las células para obtener determinados productos o cambios en una célula. Pueden ser rutas metabólicas, rutas genómicas, entre otras.

- RStudio: consiste en un entorno de desarrollo integrado que permite trabajar con el lenguaje R para desarrollar proyectos, como, por ejemplo, aplicaciones.

- SSA: del inglés Semantyc Similarities Analysis, análisis que permite comprobar similitudes funcionales entre términos de GO o de DO.

5. Bibliografía

[1] Abatangelo, L., Maglietta, R., Distaso, A., D'Addabbo, A., Creanza, T. M., Mukherjee, S., & Ancona, N. (2009). Comparative study of gene set enrichment methods. *BMC bioinformatics*, *10*(1), 275.

[2] Alexeyenko, A., Lee, W., Pernemalm, M., Guegan, J., Dessen, P., Lazar, V., ... & Pawitan, Y. (2012). Network enrichment analysis: extension of gene-set enrichment analysis to gene networks. *BMC bioinformatics*, *13*(1), 226.

[3] Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., ... & Harris, M. A. (2000). Gene Ontology: tool for the unification of biology. *Nature genetics*, *25*(1), 25.

[4] Chen, Y., McCarthy, D., Robinson, M., & Smyth, G. K. (2014). edgeR: differential expression analysis of digital gene expression data. *Bioconductor User's Guide*, 1-78.

[5] du Plessis, L., Škunca, N., & Dessimoz, C. (2011). The what, where, how and why of gene ontology—a primer for bioinformaticians. *Briefings in bioinformatics*, *12*(6), 723-735.

[6] Fabregat, A., Sidiropoulos, K., Garapati, P., Gillespie, M., Hausmann, K., Haw, R., ... & Matthews, L. (2015). The reactome pathway knowledgebase. *Nucleic acids research*, *44*(D1), D481-D487.

[7] Guzzi, P. H., Mina, M., Guerra, C., & Cannataro, M. (2011). Semantic similarity analysis of protein data: assessment with biological features and issues. *Briefings in bioinformatics*, *13*(5), 569-585.

[8] Huang, D. W., Sherman, B. T., & Lempicki, R. A. (2008). Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature protocols*, *4*(1), 44.

[9] Huang, D. W., Sherman, B. T., & Lempicki, R. A. (2008). Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. *Nucleic acids research*, *37*(1), 1-13.

[10] Jiang, Z., & Gentleman, R. (2006). Extensions to gene set enrichment. *Bioinformatics*, *23*(3), 306-313.

[11] Joshi-Tope, G., Gillespie, M., Vastrik, I., D'Eustachio, P., Schmidt, E., de Bono, B., ... & Lewis, S. (2005). Reactome: a knowledgebase of biological pathways. *Nucleic acids research*, *33*(suppl_1), D428-D432.

[12] Kanehisa, M., & Goto, S. (2000). KEGG: kyoto encyclopedia of genes and genomes. *Nucleic acids research*, *28*(1), 27-30.

[13] Khatri, P., & Drăghici, S. (2005). Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, *21*(18), 3587-3595.

[14] Khatri, P., Sirota, M., & Butte, A. J. (2012). Ten years of pathway analysis: current approaches and outstanding challenges. *PLoS computational biology*, *8*(2), e1002375.

[15] Kibbe, W. A., Arze, C., Felix, V., Mitraka, E., Bolton, E., Fu, G., ... & Parkinson, H. (2014). Disease Ontology 2015 update: an expanded and updated database of human diseases for linking biomedical knowledge through disease data. *Nucleic acids research*, *43*(D1), D1071-D1078.

[16] Lord, P. W., Stevens, R. D., Brass, A., & Goble, C. A. (2003). Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics*, *19*(10), 1275-1283.

[17] Luo, W. (2018). Pathview: pathway based data integration and visualization.

[18] R Core Team (2018). R: A language and environment for statistical computing. *R Fundation for Statistical Computing, Vienna, Austria.*

[19] Sánchez, A., & de Villa, M. C. (2008). A tutorial review of microarray data analysis. *Universitat de Barcelona*.

[20] Schriml, L. M., Arze, C., Nadendla, S., Chang, Y. W. W., Mazaitis, M., Felix, V., ... & Kibbe, W. A. (2011). Disease Ontology: a backbone for disease semantic integration. *Nucleic acids research*, *40*(D1), D940-D946.

[21] Sherman, B. T., Huang, D. W., Tan, Q., Guo, Y., Bour, S., Liu, D., ... & Lempicki, R. A. (2007). DAVID Knowledgebase: a gene-centered database integrating heterogeneous gene annotation resources to facilitate high-throughput gene functional analysis. *BMC bioinformatics*, *8*(1), 426.

[22] Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., ... & Mesirov, J. P. (2005). Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, *102*(43), 15545-15550.

[23] Wadi, L., Meyer, M., Weiser, J., Stein, L. D., & Reimand, J. (2016). Impact of outdated gene annotations on pathway enrichment analysis. *Nature methods*, *13*(9), 705.

[24] Yu, G., Li, F., Qin, Y., Bo, X., Wu, Y., & Wang, S. (2010). GOSemSim: an R package for measuring semantic similarity among GO terms and gene products. *Bioinformatics*, *26*(7), 976-978.

[25] Yu, G., Wang, L. G., Han, Y., & He, Q. Y. (2012). clusterProfiler: an R package for comparing biological themes among gene clusters. *Omics: a journal of integrative biology*, *16*(5), 284-287.

[26] Yu, G., Wang, L. G., Yan, G. R., & He, Q. Y. (2014). DOSE: an R/Bioconductor package for disease ontology semantic and enrichment analysis. *Bioinformatics*, *31*(4), 608-609.

Páginas web consultadas para el desarrollo de la aplicación:

https://shiny.rstudio.com/reference/shiny/1.0.1/fileInput.html (última visita: 18 oct 2018)

https://shiny.rstudio.com/articles/understanding-reactivity.html (última visita: 18 oct 2018)

https://shiny.rstudio.com/gallery/navbar-example.html (última visita: 19 oct 2018)

https://github.com/GuangchuangYu/DOSE/blob/master/vignettes/enrichmentAnalysis.Rmd (última visita: 19 oct 2018)

https://shiny.rstudio.com/reference/shiny/0.14/icon.html (última visita: 22 oct 2018)

https://fontawesome.com/icons?d=gallery&m=free (última visita: 24 oct 2018)

https://rstudio.github.io/shinydashboard/behavior.html#know-which-menuitem-or-menusubitem-is-currently-selected (última visita: 22 oct 2018)

https://shiny.rstudio.com/articles/action-buttons.html (última visita: 22 oct 2018)

https://rstudio.github.io/shinydashboard/structure.html (última visita: 24 oct 2018)

https://github.com/GuangchuangYu/clusterProfiler/issues/115 (última visita: 25 oct 2018)

https://guangchuangyu.github.io/2016/05/convert-biological-id-with-kegg-api-using-clusterprofiler/ (última visita: 25 oct 2018)

https://shiny.rstudio.com/articles/download.html (última visita: 17 dic 2018)

6. Anexos

6.1 Descarga de los datos de prueba

Se trata de unos conjuntos de genes incluidos en los paquetes DOSE y ChIPseeker, que pueden utilizarse para realizar pruebas en la aplicación. Antes de ejecutar el código para descargar los datos de prueba, es necesario instalar y cargar los paquetes DOSE y ChIPseeker mediante el código incluido en el apartado 6.2.

```
# Datos de prueba

data(geneList, package="DOSE")
genes <- names(geneList)[abs(geneList) > 2]

files_NGS <- getSampleFiles()
peak <- readPeakFile(files[[4]])
genes_NGS <- seq2gene(peak, tssRegion = c(-1000, 1000), flankDistance = 3000, TxDb = txdb)

write.table(genes, file = "genes.txt", sep = "\t", dec = ".", col.names = FALSE, row.names = FALSE)
write.table(genes_NGS, file = "genes_NGS.txt", sep = "\t", dec = ".", col.names = FALSE, row.names = FALSE)
```

6.2 Código en R de la aplicación

En este apartado se incluye el código para instalar y cargar todos los paquetes necesarios y todo el código realizado durante el desarrollo de la aplicación.

```
# Herramienta en línea para la realización de análisis de significación biológica
#
# Máster en Bioinformática y Bioestadística - UOC
#
# Antonio Morell Bennasser


#Instalación de paquetes

install.packages(c("shiny", "shinydashboard", "utils"), dep = TRUE)

source("http://bioconductor.org/biocLite.R")
biocLite(c("clusterProfiler", "DOSE", "GOSemSim", "ChIPseeker", "pathview",
"org.Hs.eg.db", "org.Mm.eg.db",
        "org.Rn.eg.db", "org.Sc.sgd.db", "org.Dm.eg.db", "org.Dr.eg.db",
"org.At.eg.db", "org.Bt.eg.db",
        "org.Ce.eg.db", "org.Gg.eg.db", "org.Cf.eg.db", "org.Ss.eg.db",
"org.Mmu.eg.db", "org.Eck12.eg.db",
```

```
        "org.Ag.eg.db", "org.Xl.eg.db", "org.Pt.eg.db", "org.Pf.plasmo.db",
"org.EcSakai.eg.db",
        "TxDb.Hsapiens.UCSC.hg19.knownGene"))

## Carga de los paquetes

require(shiny)
require(shinydashboard)
require(utils)

require(clusterProfiler)
require(DOSE)
require(GOSemSim)
require(ChIPseeker)
require(pathview)

require(org.Hs.eg.db)
require(org.Mm.eg.db)
require(org.Rn.eg.db)
require(org.Sc.sgd.db)
require(org.Dm.eg.db)
require(org.Dr.eg.db)
require(org.At.eg.db)
require(org.Bt.eg.db)
require(org.Ce.eg.db)
require(org.Gg.eg.db)
require(org.Cf.eg.db)
require(org.Ss.eg.db)
require(org.Mmu.eg.db)
require(org.Eck12.eg.db)
require(org.Ag.eg.db)
require(org.Xl.eg.db)
require(org.Pt.eg.db)
require(org.Pf.plasmo.db)
require(org.EcSakai.eg.db)

require(TxDb.Hsapiens.UCSC.hg19.knownGene)

## UI & Server

# Definimos la UI
ui <- dashboardPage(

  dashboardHeader(title = "Análisis de significación biológica", titleWidth = 350),

    # Generamos la barra lateral
    dashboardSidebar(width = 350,

      # Definimos la barra lateral y los menús
      sidebarMenu(
```

```
        menuItem(text = "Información", tabName = "info", icon = icon("fas fa-info-
circle")),
        menuItem(text = "Carga de datos", tabName = "upload", icon =
icon("upload")),
        menuItem(text = "GO Analysis", tabName = "goanalysis", icon = icon("fas
fa-globe"),
                 menuSubItem(text = "GO Gene Enrichment Analysis (con
enrichGO)", tabName = "GOgea"),
                 menuSubItem(text = "GO Gene Set Enrichment Analysis (con
gseGO)", tabName = "GOgsea"),
                 menuSubItem(text = "GO Semantic Similarity Analysis (con
mgoSim)", tabName = "GOssa")
                 ),
        menuItem(text = "KEGG Analysis", tabName = "kegganalysis", icon =
icon("fas fa-globe"),
                 menuSubItem(text = "KEGG Gene Enrichment Analysis (con
enrichKEGG)", tabName = "KEGGgea"),
                 menuSubItem(text = "KEGG Gene Set Enrichment Analysis (con
gseKEGG)", tabName = "KEGGgsea")
                 ),
        menuItem(text = "NGS Analysis", tabName = "ngsanalysis", icon =
icon("fas fa-globe"),
                 menuSubItem(text = "Preparación de los datos", tabName =
"NGSdata")
        )
      )
    ),

    # Generamos los paneles y definimos cada apartado
    dashboardBody(
      tabItems(
        tabItem(tabName = "info",
                p("Los análisis de significación biológica o análisis funcionales
proporcionan resultados
                 dentro de un contexto biológico permitiendo interpretar resultados
de análisis de expresión
                 diferencial de una forma más clara y concisa, relacionando las
listas de genes
                 diferencialmente expresados obtenidas en estos análisis con bases
de datos de anotaciones
                 funcionales. Esta herramienta en línea permite realizar los
principales análisis de
                 significación biológica así como otros análisis complementarios de
interés. Seguidamente se
                 describen las distintas partes de esta herramienta en línea:"),
                tags$ol(
                 tags$li(p(strong("Carga de los datos: "), " Cargar el conjunto de
genes de interés contenidos
```

```
                    en un archivo en formato .txt con una sola columna que
disponga de una cabecera y
                    en cuyas filas se encuentren los IDs de los genes. Cargar el
universo de genes
                    (opcional). Cargar el conjunto de datos de recuentos. Cargar
los conjuntos de
                    términos de GO para el análisis de similitudes entre
términos.")
                  ),
            tags$li(p(strong("GO analysis: "), " Análisis de enriquecimiento (Gen
Enrichment Analysis -
                    GEA y Gen Set Enrichment Analysis - GSEA) a partir de los
conjuntos de genes de
                    interés utilizando las anotaciones de Gene Ontology (GO)
para la obtención de
                    términos sobre-representados o sub-representados. También
es posible realizar
                    análisis de similitudes semánticas (SSA) para estudiar la
presencia o ausencia de
                    similitudes entre los aspectos funcionales relacionados entre
dos o más términos de
                    GO.")
                  ),
            tags$li(p(strong("KEGG analysis: "), " Análisis de enriquecimiento
(GEA y GSEA) a partir de
                    los conjuntos de genes de interés utilizando las anotaciones
de la Kyoto
                    Encyclopedia of Genes and Genomes (KEGG) para la
obtención de términos
                    sobre-representados o sub-representados así como obtener
las rutas KEGG
                    (KEGG-pathways) sobre-representadas.")
            ),
            tags$li(p(strong("NGS analysis: "), " Análisis de datos obtenidos a
partir de Next Generation
                    Sequencing (NGS). En primer lugar se debe realizar una
preparación de los datos,
                    que, posteriormente, deberán descargarse para poder aplicar
los análisis anteiores
                    deseados.")
            )
          )
          ),

      # Panel de carga de datos
      tabItem(tabName = "upload",
          h2("Cargar los archivos de IDs de los genes"),
          br(),
          fileInput(inputId = "geneFile",
                  label = "Cargar los ID de los genes",
```

```
                accept = c("text", ".txt"),
                buttonLabel = "Buscar...",
                placeholder = "Ningún archivo"),
        fileInput(inputId = "universeFile",
                label = "Cargar el universo de genes (opcional)",
                accept = c("text", ".txt"),
                buttonLabel = "Buscar...",
                placeholder = "Ningún archivo"),
        fileInput(inputId = "GO1",
                label = "Conjunto 1 de términos de GO (GO SSA)",
                accept = c("text", ".txt"),
                buttonLabel = "Buscar...",
                placeholder = "Ningún archivo"),
        fileInput(inputId = "GO2",
                label = "Conjunto 2 de términos de GO (GO SSA)",
                accept = c("text", ".txt"),
                buttonLabel = "Buscar...",
                placeholder = "Ningún archivo"),
        fileInput(inputId = "peakFile",
                label = "Cargar el archivo de NGS",
                accept = c("text", ".txt"),
                buttonLabel = "Buscar...",
                placeholder = "Ningún archivo")
        ),

    # GO Analysis
    ## Panel para el análisis de enriquecimiento GEA de GO (enrichGO)
    tabItem(tabName = "GOgea",
        h2("GO Gene Enrichment Analysis"),
        fluidRow(
          box(title = "Configurar los parámetros del análisis",
            status = "warning",
            selectInput(inputId = "OrgDb",
                    label = "OrgDb",
                    choices = c("org.Hs.eg.db", "org.Mm.eg.db",
"org.Sc.sgd.db"),
                    selected = "org.Hs.eg.db"),
            selectInput(inputId = "keyType",
                    label = "Keytype of input gene",
                    choices = c(keytypes(org.Hs.eg.db),
                            "MGI", "COMMON", "DESCRIPTION",
"INTERPRO", "ORF", "SGD", "SMART"),
                    selected = "ENTREZID"),
            selectInput(inputId = "ont",
                    label = "Subontologies",
                    choices = c("MF", "BP", "CC", "ALL"),
                    selected = "MF"),
            numericInput(inputId = "pvalueCutoff",
                     label = "p-value cutoff",
                     value = 0.05,
```

```
                        step = 0.01),
                selectInput(inputId = "pAdjMethod",
                        label = "p-value adjust method",
                        choices = c("holm", "hochberg", "hommel", "bonferroni",
"BH", "BY", "fdr", "none"),
                        selected = "BH"),
                numericInput(inputId = "qvalueCutoff",
                        label = "q-value cutoff",
                        value = 0.2,
                        step = 0.1),
                numericInput(inputId = "minGSSize",
                        label = "Minimal size of genes annotated by Ontology
term for testing",
                        value = 10,
                        step = 1),
                numericInput(inputId = "maxGSSize",
                        label = "Maximal size of genes annotated for testing",
                        value = 500,
                        step = 1),
                selectInput(inputId = "readable",
                        label = "Readable: whether mapping gene ID to gene
Name",
                        choices = c("TRUE", "FALSE"),
                        selected = "FALSE"),
                selectInput(inputId = "pool",
                        label = "Pool: if ont='ALL', whether pool 3 GO sub-
ontologies",
                        choices = c("TRUE", "FALSE"),
                        selected = "FALSE"),
                actionButton(inputId = "submit1",
                        label = "Submit")
                )
            ),
          fluidRow(
            box(title = "Tabla de resultados de GO GEA",
                status = "primary",
                tableOutput(outputId = "enrichGO_results"),
                downloadButton(outputId = "download_GOGEA_res", label =
"Download")
                ),
            box(title = "Diagrama de puntos de GO GEA",
                status = "success",
                plotOutput(outputId = "dotplot_GOgea"),
                downloadButton(outputId = "download_GOGEA_dot", label =
"Download")
                ),
            box(title = "Mapa de enriquecimiento de GO GEA",
                status = "danger",
                plotOutput(outputId = "emapplot_GOgea"),
```

```
                downloadButton(outputId = "download_GOGEA_map", label =
"Download")
                ),
              box(title = "Representación de asociaciones",
                status = "info",
                plotOutput(outputId = "cnetplot_GOgea"),
                downloadButton(outputId = "download_GOGEA_cnet", label =
"Download")
                )
            )
      ),

      ## Panel para el análisis de enriquecimiento GSEA de GO (gseGO)
      tabItem(tabName = "GOgsea",
            h2("GO Gene Set Enrichment Analysis"),
            fluidRow(
              box(title = "Configurar los parámetros del análisis",
                status = "warning",
                selectInput(inputId = "ont_gse",
                            label = "Subontologies",
                            choices = c("BP", "MF", "CC", "GO"),
                            selected = "MF"),
                selectInput(inputId = "OrgDb_gse",
                            label = "OrgDb",
                            choices = c("org.Hs.eg.db", "org.Mm.eg.db",
"org.Sc.sgd.db"),
                            selected = "org.Hs.eg.db"),
                textInput(inputId = "keyType_gse",
                          label = "Keytype of input gene",
                          value = "ENTREZID"),
                numericInput(inputId = "exponent",
                             label = "Exponent: weight of each step",
                             value = 1,
                             step = 0.5),
                numericInput(inputId = "nPerm",
                             label = "Permutation numbers",
                             value = 1000,
                             step = 100),
                numericInput(inputId = "minGSSize_gse",
                             label = "Minimal size of each geneSet for analyzing",
                             value = 10,
                             step = 1),
                numericInput(inputId = "maxGSSize_gse",
                             label = "Maximal size of genes annotated for testing",
                             value = 500,
                             step = 1),
                numericInput(inputId = "pvalueCutoff_gse",
                             label = "p-value cutoff",
                             value = 0.05,
                             step = 0.01),
```

```
                selectInput(inputId = "pAdjMethod_gse",
                            label = "p-value adjust method",
                            choices = c("holm", "hochberg", "hommel", "bonferroni",
"BH", "BY", "fdr", "none"),
                            selected = "BH"),
                selectInput(inputId = "verbose",
                            label = "Print message or not",
                            choices = c("TRUE", "FALSE"),
                            selected = "TRUE"),
                selectInput(inputId = "seed",
                            label = "Seed",
                            choices = c("TRUE", "FALSE"),
                            selected = "FALSE"),
                selectInput(inputId = "by",
                            label = "By one of 'fgsea' or 'DOSE'",
                            choices = c("fgsea", "DOSE"),
                            selected = "fgsea"),
                actionButton(inputId = "submit2",
                             label = "Submit")
              )
            ),
            fluidRow(
              box(title = "Tabla de resultados de GO GSEA",
                  status = "primary",
                  tableOutput(outputId = "gseGO_results"),
                  downloadButton(outputId = "download_GOGSEA_res", label =
"Download")
              ),
              box(title = "Diagrama de puntos de GO GSEA",
                  status = "success",
                  plotOutput(outputId = "dotplot_GOgsea"),
                  downloadButton(outputId = "download_GOGSEA_dot", label =
"Download")
                  ),
              box(title = "Mapa de enriquecimiento de GO GSEA",
                  status = "danger",
                  plotOutput(outputId = "emapplot_GOgsea"),
                  downloadButton(outputId = "download_GOGSEA_map", label =
"Download")
              ),
              box(title = "Representación de asociaciones",
                  status = "info",
                  plotOutput(outputId = "cnetplot_GOgsea"),
                  downloadButton(outputId = "download_GOGSEA_cnet", label =
"Download")
              ),
              box(title = "Puntuación de GSEA y asociación de fenotipo",
                  status = "warning",
                  plotOutput(outputId = "gseaplot_GOgsea"),
```

```
                downloadButton(outputId = "download_GOGSEA_fen", label =
"Download")
                 )
               )
        ),

        ## Panel para el test de similitudes semánticas de GO (mgoSim)
        tabItem(tabName = "GOssa",
               h2("GO Semantic Similarity Analysis"),
               fluidRow(
                 box(title = "Configurar los parámetros del análisis",
                    status = "warning",
                    selectInput(inputId = "measure",
                                label = "Choose one measure method",
                                choices = c("Resnik", "Lin", "Rel", "Jiang", "Wang"),
                                selected = "Wang"),
                    selectInput(inputId = "combine",
                                label = "Choose one combine method",
                                choices = c("max", "avg", "rcmax", "BMA"),
                                selected = "BMA"),
                    actionButton(inputId = "submit3",
                                 label = "Submit")
                 )
               ),
               fluidRow(
                 box(title = "Resultados de GO SSA",
                    status = "primary",
                    plotOutput(outputId = "mgoSim_results"),
                    downloadButton(outputId = "download_GOSSA_res", label =
"Download")
                 )
               )
        ),

        # KEEG Analysis
        ## Panel para el test de sobre-representación de KEGG (enrichKEGG)
        tabItem(tabName = "KEGGgea",
               h2("KEGG Gene Enrichment Analysis"),
               fluidRow(
                 box(title = "Configurar los parámetros del análisis",
                    status = "warning",
                    textInput(inputId = "organism",
                              label = "Organism from
'https://www.genome.jp/kegg/catalog/org_list.html'",
                              value = "hsa"),
                    textInput(inputId = "keyType",
                              label = "Keytype of input gene",
                              value = "kegg"),
                    numericInput(inputId = "pvalueCutoff",
                                 label = "p-value cutoff",
```

```
                        value = 0.05,
                        step = 0.01),
                    selectInput(inputId = "pAdjMethod",
                                label = "p-value adjust method",
                                choices = c("holm", "hochberg", "hommel", "bonferroni",
"BH", "BY", "fdr", "none"),
                                selected = "BH"),
                    numericInput(inputId = "minGSSize",
                                 label = "minimal size of genes annotated by Ontology
term for testing",
                                 value = 10,
                                 step = 1),
                    numericInput(inputId = "maxGSSize",
                                 label = "maximal size of genes annotated for testing",
                                 value = 500,
                                 step = 1),
                    numericInput(inputId = "qvalueCutoff",
                                 label = "q-value cutoff",
                                 value = 0.2,
                                 step = 0.1),
                    actionButton(inputId = "submit4",
                                 label = "Submit")
                  )
                ),
                fluidRow(
                  box(title = "Tabla de resultados de KEGG GEA",
                      status = "primary",
                      verbatimTextOutput(outputId = "enrichKEGG_results"),
                      downloadButton(outputId = "download_KEGGGEA_res", label =
"Download")
                  ),
                  box(title = "Diagrama de puntos de KEGG GEA",
                      status = "success",
                      plotOutput(outputId = "dotplot_KEGGgea"),
                      downloadButton(outputId = "download_KEGGGEA_dot", label =
"Download")
                  ),
                  box(title = "Mapa de enriquecimiento de KEGG GEA",
                      status = "danger",
                      plotOutput(outputId = "emapplot_KEGGgea"),
                      downloadButton(outputId = "download_KEGGGEA_map", label =
"Download")
                  ),
                  box(title = "Representación de asociaciones",
                      status = "info",
                      plotOutput(outputId = "cnetplot_KEGGgea"),
                      downloadButton(outputId = "download_KEGGGEA_cnet", label =
"Download")
                  ),
                  box(title = "Representación de rutas KEGG",
```

```
                status = "warning",
                textInput(inputId = "pathwayId",
                        label = "The KEGG pathway ID"),
                textInput(inputId = "species",
                        label = "Target species",
                        value = "hsa"),
                actionButton(inputId = "submit5",
                          label = "Submit")
                )
            )
        ),

        ## Panel para el análisis de enriquecimiento GSEA de KEGG (gseKEGG)
        tabItem(tabName = "KEGGgsea",
              h2("KEGG Gene Set Enrichment Analysis"),
              fluidRow(
                box(title = "Configurar los parámetros del análisis",
                status = "warning",
                textInput(inputId = "organism_gse",
                        label = "Organism from
'https://www.genome.jp/kegg/catalog/org_list.html'",
                        value = "hsa"),
                selectInput(inputId = "keyType_KEGGgse",
                        label = "Keytype of input gene",
                        choices = c("kegg", "ncbi-geneid", "ncib-proteinid",
"uniprot"),
                        selected = "kegg"),
                numericInput(inputId = "exponent_gse",
                          label = "Exponent: weight of each step",
                          value = 1,
                          step = 0.5),
                numericInput(inputId = "nPerm_gse",
                          label = "Permutation numbers",
                          value = 1000,
                          step = 100),
                numericInput(inputId = "minGSSize_KEGGgse",
                          label = "Minimal size of each geneSet for analyzing",
                          value = 10,
                          step = 1),
                numericInput(inputId = "maxGSSize_KEGGgse",
                          label = "Maximal size of genes annotated for testing",
                          value = 500,
                          step = 1),
                numericInput(inputId = "pvalueCutoff_KEGGgse",
                          label = "p-value cutoff",
                          value = 0.05,
                          step = 0.01),
                selectInput(inputId = "pAdjMethod_KEGGgse",
                         label = "p-value adjust method",
```

```
                        choices = c("holm", "hochberg", "hommel", "bonferroni",
"BH", "BY", "fdr", "none"),
                        selected = "BH"),
                    selectInput(inputId = "verbose_gse",
                        label = "Print message or not",
                        choices = c("TRUE", "FALSE"),
                        selected = "TRUE"),
                    selectInput(inputId = "use_internal_data",
                        label = "Use KEGG.db or latest online KEGG data",
                        choices = c("TRUE", "FALSE"),
                        selected = "FALSE"),
                    selectInput(inputId = "seed_gse",
                        label = "Seed",
                        choices = c("TRUE", "FALSE"),
                        selected = "FALSE"),
                    selectInput(inputId = "by_gsea",
                        label = "By one of 'fgsea' or 'DOSE'",
                        choices = c("fgsea", "DOSE"),
                        selected = "fgsea"),
                    actionButton(inputId = "submit6",
                         label = "Submit")
                )
              ),
              fluidRow(
                box(title = "Tabla de resultados de KEGG GSEA",
                    status = "primary",
                    tableOutput(outputId = "gseKEGG_results"),
                    downloadButton(outputId = "download_KEGGGSEA_res", label =
"Download")
                ),
                box(title = "Diagrama de puntos de KEGG GSEA",
                    status = "success",
                    plotOutput(outputId = "dotplot_KEGGgsea"),
                    downloadButton(outputId = "download_KEGGGSEA_dot", label =
"Download")
                ),
                box(title = "Mapa de enriquecimiento de KEGG GSEA",
                    status = "danger",
                    plotOutput(outputId = "emapplot_KEGGgsea"),
                    downloadButton(outputId = "download_KEGGGSEA_map", label
= "Download")
                ),
                box(title = "Representación de asociaciones",
                    status = "info",
                    plotOutput(outputId = "cnetplot_KEGGgsea"),
                    downloadButton(outputId = "download_KEGGGSEA_cnet", label
= "Download")
                ),
                box(title = "Puntuación de GSEA y asociación de fenotipo",
                    status = "warning",
```

```
                plotOutput(outputId = "gseaplot_KEGGgsea"),
                downloadButton(outputId = "download_KEGGGSEA_fen", label =
"Download")
              )
            )
      ),

      ## Panel para la preparación de los datos de NGS
      tabItem(tabName = "NGSdata",
            h2("Análisis de datos de NGS. Preparación de los datos"),
            fluidRow(
              box(title = "Preparación de los datos de NGS",
                status = "warning",
                numericInput(inputId = "tssRegion_down",
                             label = "TSS region down",
                             value = -1000,
                             step = 500),
                numericInput(inputId = "tssRegion_up",
                             label = "TSS region up",
                             value = 1000,
                             step = 500),
                numericInput(inputId = "flankDistance",
                             label = "Flanking search radius",
                             value = 3000,
                             step = 500),
                textInput(inputId = "TxDb",
                             label = "TranscriptDb object",
                             value = "TxDb.Hsapiens.UCSC.hg19.knownGene"),
                selectInput(inputId = "sameStrand",
                            label = "Whether find nearest/overlap gene in the same
strand",
                            choices = c("TRUE", "FALSE"),
                            selected = "FALSE"),
                actionButton(inputId = "submit7",
                             label = "Submit"),
                downloadButton(outputId = "downloadGeneFile", label =
"Download")
              )
            )
      )
    )
  )



# Define server logic required to draw a histogram
server <- function(input, output) {

  # Cargamos los datos del usuario
```

```
## Carga del archivo de datos de los ID de los genes
genesID <- reactive({
  genesFile <- input$genesFile
  if (!is.null(genesFile)) {
    genesID <- read.table(genesFile$datapath)
    genesID <- as.character(genesID[, 1])
  }
  return(genesID)
})

## Carga del archivo de datos del universo de genes
universe <- reactive({
  universeFile <- input$universeFile
  if (!is.null(universeFile)) {
    universe <- read.table(universeFile$datapath)
    universe <- as.character(universe[, 1])
  }
  return(universe)
})

## Carga de los archivos de datos de los términos GO para la función mgoSim
GOTerms1 <- reactive({
  GOFile1 <- input$GO1
  if (!is.null(GOFile1)) {
    GOTerms1 <- read.table(GOFile1$datapath)
    GOTerms1 <- as.character(GOTerms1[, 1])
  }
  return(GOTerms1)
})

GOTerms2 <- reactive({
  GOFile2 <- input$GO2
  if (!is.null(GOFile2)) {
    GOTerms2 <- read.table(GOFile2$datapath)
    GOTerms2 <- as.character(GOTerms2[, 1])
  }
  return(GOTerms2)
})

## Carga de los archivos de datos de NGS
NGSpeaks <- reactive({
  NGSpeakFile <- input$peakFile
  if (!is.null(NGSpeakFile)) {
    NGSpeaks <- read.table(NGSpeakFile$datapath)
    NGSpeaks <- as.character(NGSpeaks[, 1])
  }
  return(NGSpeaks)
})

# GO Analysis
```

```
  ## Output de la función enrichGO
  e_GO <- eventReactive(input$submit1, {
    e_GO <- enrichGO(
      gene = genesID(), OrgDb = input$OrgDb, keyType = input$keyType, ont =
input$ont,
      pvalueCutoff = input$pvalueCutoff, pAdjustMethod = input$pAdjMethod,
      universe = universe(), qvalueCutoff = input$qvalCutoff,
      minGSSize = input$minGSSize, maxGSSize = input$maxGSSize,
      readable = input$readable, pool = input$pool
      )
    })

  ### Resultados de la función enrichGO
  results_enrichGO <- reactive({
    if (!is.null(e_GO())) {
      results_enrichGO <- e_GO@result
    } else {NULL}
  })

  ### Output de la tabla con los resultados de la función enrichGO
  output$enrichGO_results <- renderTable(
    results_enrichGO()
    )

  #### Descarga de la tabla de resultados de enrichGO
  output$download_GOGEA_res <- downloadHandler(
    filename = "GO_GEA_results.txt",
    content = function(file) {
      write.csv(results_enrichGO(), file, sep = ",", row.names = FALSE, col.names
= TRUE)
    }
  )

  ### Output con el diagrama de puntos
  output$dotplot_GOgea <- renderPlot(
    if (!is.null(e_GO())) {
      DOSE::dotplot(e_GO())
    }
  )

  #### Descarga del diagrama de puntos de GO GEA
  output$download_GOGEA_dot <- downloadHandler(
    filename = "GO_GEA_dotplot.txt",
    content = function(file) {
      ggsave(file, DOSE::dotplot(e_GO()))
    }
  )

  ### Output con el mapa de enriquecimiento
  output$emapplot_GOgea <- renderPlot(
```

```
    if (!is.null(e_GO())) {
      DOSE::emapplot(e_GO())
    }
  )

  #### Descarga del mapa de enriquecimiento
  output$download_GOGEA_map <- downloadHandler(
    filename = "GO_GEA_emapplot.txt",
    content = function(file) {
      ggsave(file, DOSE::emapplot(e_GO()))
    }
  )

  ### Output con la representación de las asociaciones
  output$cnetplot_GOgea <- renderPlot(
    if (!is.null(e_GO())) {
      DOSE::cnetplot(e_GO())
    }
  )

  #### Descarga de la representación de asociaciones
  output$download_GOGEA_cnet <- downloadHandler(
    filename = "GO_GEA_cnetplot.txt",
    content = function(file) {
      ggsave(file, DOSE::cnetplot(e_GO()))
    }
  )

  ## Output de la función gseGO
  gse_GO <- eventReactive(input$submit2, {
    gse_GO <- gseGO(
      genesID(), ont = input$ont_gse, OrgDb = input$OrgDb_gse, keyType =
input$keyType_gse,
      exponent = input$exponent, nPerm = input$nPerm, minGSSize =
input$minGSSize_gse,
      maxGSSize = input$maxGSSize_gse, pvalueCutoff =
input$pvalueCutoff_gse,
      pAdjustMethod = input$pAdjMethod_gse, verbose = input$verbose, seed =
input$seed, by = input$by
    )
  })

  ### Resultados de la función gseGO
  results_gseGO <- reactive({
    if (!is.null(gse_GO())) {
      results_gseGO <- gse_GO@result
    } else {NULL}
  })

  ### Output de la tabla con los resultados de la función gseGO
```

```
  output$gseGO_results <- renderTable(
    results_gseGO()
  )

  #### Descarga de la tabla de resultados de gseGO
  output$download_GOGSEA_res <- downloadHandler(
    filename = "GO_GSEA_results.txt",
    content = function(file) {
      write.csv(results_gseGO(), file, sep = ",", row.names = FALSE, col.names =
TRUE)
    }
  )

  ### Output con el diagrama de puntos
  output$dotplot_GOgsea <- renderPlot(
    if (!is.null(gse_GO())) {
      DOSE::dotplot(gse_GO())
    }
  )

  #### Descarga del diagrama de puntos de GO GSEA
  output$download_GOGSEA_dot <- downloadHandler(
    filename = "GO_GSEA_dotplot.txt",
    content = function(file) {
      ggsave(file, DOSE::dotplot(gse_GO()))
    }
  )

  ### Output con el mapa de enriquecimiento
  output$emapplot_GOgsea <- renderPlot(
    if (!is.null(gse_GO())) {
      DOSE::emapplot(gse_GO())
    }
  )

  #### Descarga del mapa de enriquecimiento
  output$download_GOGSEA_map <- downloadHandler(
    filename = "GO_GSEA_emapplot.txt",
    content = function(file) {
      ggsave(file, DOSE::emapplot(gse_GO()))
    }
  )

  ### Output con la representación de las asociaciones
  output$cnetplot_GOgsea <- renderPlot(
    if (!is.null(gse_GO())) {
      DOSE::cnetplot(gse_GO())
    }
  )
```

```
  #### Descarga de la representación de las asociaciones
  output$download_GOGSEA_cnet <- downloadHandler(
    filename = "GO_GSEA_cnetplot.txt",
    content = function(file) {
      ggsave(file, DOSE::cnetplot(gse_GO()))
    }
  )

  ### Output con la puntuación de GSEA y asociación de fenotipo
  output$gseaplot_GOgsea <- renderPlot(
    if (!is.null(gse_GO())) {
      DOSE::gseaplot(gse_GO())
    }
  )

  #### Descarga de la puntuación de GSEA y asociación de fenotipo
  output$download_GOGSEA_fen <- downloadHandler(
    filename = "GO_GSEA_gseaplot.txt",
    content = function(file) {
      ggsave(file, DOSE::gseaplot(gse_GO()))
    }
  )

  ## Output de la función mgoSim
  ssa_GO <- eventReactive(input$submit3, {
    ssa_GO <- mgoSim(
      GO1 = GOTerms1(), GO2 = GOTerms2(), semData = (hsGO <-
godata('org.Hs.eg.db', ont = "MF")),
      measure = input$measure, combine = input$combine
    )
  })

  ### Resultados de la función mgoSim
  results_mgoSimGO <- reactive({
    if (!is.null(ssa_GO())) {
      results_mgoSimGO <- ssa_GO@result
    } else {NULL}
  })

  ### Output de la tabla con los resultados de la función mgoSim
  output$mgoSim_results <- renderTable(
    results_mgoSimGO()
  )

  #### Descarga de la tabla de resultados de gseGO
  output$download_GOSSA_res <- downloadHandler(
    filename = "GO_SSA_results.txt",
    content = function(file) {
      write.csv(results_mgoSimGO(), file, sep = ",", row.names = FALSE,
col.names = TRUE)
```

```
  }
)

# KEGG Analysis
## Output de la función enrichKEGG
e_KEGG <- eventReactive(input$submit4, {
  e_KEGG <- enrichKEGG(
    gene = genesID(), organism = input$organism, keyType = input$keyType,
    pvalueCutoff = input$pvalueCutoff, pAdjustMethod = input$pAdjMethod,
    universe = universe(), minGSSize = input$minGSSize,
    maxGSSize = input$maxGSSize, qvalueCutoff = input$qvalCutoff
  )
})

### Resultados de la función enrichKEGG
results_enrichKEGG <- reactive({
  if (!is.null(e_KEGG())) {
    results_enrichKEGG <- e_KEGG@result
  } else {NULL}
})

### Output de la tabla con los resultados de la función enrichKEGG
output$enrichKEGG_results <- renderPrint(
  results_enrichKEGG()
)

#### Descarga de la tabla de resultados de enrichKEGG
output$download_KEGGGEA_res <- downloadHandler(
  filename = "KEGG_GEA_results.txt",
  content = function(file) {
    write.csv(results_enrichKEGG(), file, sep = ",", row.names = FALSE,
col.names = TRUE)
  }
)

### Output con el diagrama de puntos
output$dotplot_KEGGgea <- renderPlot(
  if (!is.null(e_KEGG())) {
    DOSE::dotplot(e_KEGG())
  }
)

#### Descarga del diagrama de puntos
output$download_KEGGGEA_dot <- downloadHandler(
  filename = "KEGG_GEA_dotplot.txt",
  content = function(file) {
    ggsave(file, DOSE::dotplot(e_KEGG()))
  }
)
```

```
  ### Output con el mapa de enriquecimiento
  output$emapplot_KEGGgea <- renderPlot(
    if (!is.null(e_KEGG())) {
      DOSE::emapplot(e_KEGG())
    }
  )

  #### Descarga del mapa de enriquecimiento
  output$download_KEGGGEA_map <- downloadHandler(
    filename = "KEGG_GEA_emapplot.txt",
    content = function(file) {
      ggsave(file, DOSE::emapplot(e_KEGG()))
    }
  )

  ### Output con la representación de las asociaciones
  output$cnetplot_KEGGgea <- renderPlot(
    if (!is.null(e_KEGG())) {
      DOSE::cnetplot(e_KEGG())
    }
  )

  #### Descarga de la representación de las asociaciones
  output$download_KEGGGEA_cnet <- downloadHandler(
    filename = "KEGG_GEA_cnetplot.txt",
    content = function(file) {
      ggsave(file, DOSE::cnetplot(e_KEGG()))
    }
  )

  ## Output de la función gseKEGG
  gse_KEGG <- eventReactive(input$submit6, {
    gse_KEGG <- gseKEGG(
      genesID(), organism = input$organism_gse, keyType =
input$keyType_KEGGgse,
      exponent = input$exponent_gse, nPerm = input$nPerm_gse, minGSSize =
input$minGSSize_KEGGgse,
      maxGSSize = input$maxGSSize_KEGGgse, pvalueCutoff =
input$pvalueCutoff_KEGGgse,
      pAdjustMethod = input$pAdjMethod_KEGGgse, verbose =
input$verbose_gse,
      use_internal_data = input$use_internal_data, seed = input$seed_gse, by =
input$by_gse
    )
  })

  ### Resultados de la función gseKEGG
  results_gseKEGG <- reactive({
    if (!is.null(gse_KEGG())) {
      results_gseKEGG <- gse_KEGG@result
```

```
  } else {NULL}
 })

 ### Output de la tabla con los resultados de la función gseKEGG
 output$gseKEGG_results <- renderTable(
  results_gseKEGG()
 )

 #### Descarga de la tabla de resultados de gseKEGG
 output$download_KEGGGSEA_res <- downloadHandler(
  filename = "KEGG_GSEA_results.txt",
  content = function(file) {
   write.csv(results_gseKEGG(), file, sep = ",", row.names = FALSE,
col.names = TRUE)
  }
 )

 ### Output con el diagrama de puntos
 output$dotplot_KEGGgsea <- renderPlot(
  if (!is.null(gse_KEGG())) {
   DOSE::dotplot(gse_KEGG())
  }
 )

 #### Descarga del diagrama de puntos
 output$download_KEGGGSEA_dot <- downloadHandler(
  filename = "KEGG_GSEA_dotplot.txt",
  content = function(file) {
   ggsave(file, DOSE::dotplot(gse_KEGG()))
  }
 )

 ### Output con el mapa de enriquecimiento
 output$emapplot_KEGGgsea <- renderPlot(
  if (!is.null(gse_KEGG())) {
   DOSE::emapplot(gse_KEGG())
  }
 )

 #### Descarga del mapa de enriquecimiento
 output$download_KEGGGSEA_map <- downloadHandler(
  filename = "KEGG_GSEA_emapplot.txt",
  content = function(file) {
   ggsave(file, DOSE::emapplot(gse_KEGG()))
  }
 )

 ### Output con la representación de las asociaciones
 output$cnetplot_KEGGgsea <- renderPlot(
  if (!is.null(gse_KEGG())) {
```

```
      DOSE::cnetplot(gse_KEGG())
    }
  )

  #### Descarga de la representación de las asociaciones
  output$download_KEGGGSEA_cnet <- downloadHandler(
    filename = "KEGG_GSEA_cnetplot.txt",
    content = function(file) {
      ggsave(file, DOSE::cnetplot(gse_KEGG()))
    }
  )

  ### Output con la puntuación de GSEA y asociación de fenotipo
  output$gseaplot_KEGGgsea <- renderPlot(
    if (!is.null(gse_KEGG())) {
      DOSE::gseaplot(gse_KEGG())
    }
  )

  #### Descarga de la puntuación de GSEA y asociación de fenotipo
  output$download_KEGGGSEA_fen <- downloadHandler(
    filename = "KEGG_GSEA_gseaplot.txt",
    content = function(file) {
      ggsave(file, DOSE::gseaplot(gse_KEGG()))
    }
  )

  # NGS Analysis
  ## Output de la función seq2gene
  s2g <- eventReactive(input$submit7, {
    s2g <- seq2gene(
      seq = NGSpeaks(), tssRegion = c(input$tssRegion_down,
input$tssRegion_up),
      flankDistance = input$flankDistance, TxDb = input$TxDb, sameStrand =
input$sameStrand
    )
  })

  output$downloadGeneFile <- downloadHandler(
    filename = "geneIDs.txt",
    content = function(file) {
      write.table(s2g(), file, sep = "", row.names = FALSE, col.names = FALSE)
    }
  )

}

# Run the application
shinyApp(ui = ui, server = server)
```