UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

GRADO EN INGENERÍA INFORMÁTICA

ANÁLISIS Y EXTRACCIÓN DE CONOCIMIENTOS DE LAS REDES SOCIALES USANDO FCA

ANALYSIS AND KNOWLEDGE EXTRACTION FROM SOCIAL NETWORKS USING FCA

Realizado por
JEAN-PAUL BEAUDRY LOPERA

Tutorizado por
ÁNGEL MORA BONILLA
DOMINGO LÓPEZ ROGRÍGUEZ

Departamento
MATEMÁTICA APLICADA
UNIVERSIDAD DE MÁLAGA

MÁLAGA, SEPTIEMBRE 2021

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA

INFORMÁTICA

GRADUADO EN INGENIERÍA INFORMÁTICA

ANÁLISIS Y EXTRACCIÓN DE CONOCIMIENTOS DE LAS REDES SOCIALES USANDO FCA

ANALYSIS AND KNOWLEDGE EXTRACTION FROM SOCIAL NETWORKS USING FCA

Realizado por
JEAN-PAUL BEAUDRY LOPERA

Tutorizado por
ÁNGEL MORA BONILLA
DOMINGO LÓPEZ RODRÍGUEZ

Departamento
MATEMÁTICA APLICADA

UNIVERSIDAD DE MÁLAGA
MÁLAGA, SEPTIEMBRE DE 2021

Fecha defensa: septiembre de 2021

Abstract

In this final year dissertation we present the development of a web application that facilitates the collection of data from certain social networks to perform analysis of these networks in order to extract information that is present in the data obtained.

The main tool to achieve this goal is to make use of Formal Concept Analysis or FCA, which is a mathematical theory that makes use of lattice theory and logic to discover knowledge from a dataset, this tool is able to find information in a similar way or even more efficient than other famous techniques such as Association Rules.

Also, FCA will be used with other tools already known in data science in order to obtain as much information as possible from a dataset.

Keywords: *Formal Concept Analysis, Social Network Analysis, Data Mining, R (Programming language)*

Resumen

En este trabajo fin de grado se presenta el desarrollo de una aplicación web que facilita la obtención de datos de ciertas redes sociales para realizar análisis de dichas redes con la finalidad de extraer información que se encuentra presente en el conjunto de datos que se obtiene.

La principal herramienta para conseguir este objetivo es hacer uso de Análisis de Conceptos Formales o AFC, la cual es una teoría matemática basada en la teoría de retículos y de la lógica para descubrir conocimiento de un conjunto de datos, esta herramienta es capaz de encontrar información de forma similar o incluso de una forma más eficiente que otras técnicas más conocidas como pueden ser las Reglas de Asociación.

Así mismo, se utilizará AFC con otras herramientas ya conocidas en la ciencia de datos para poder obtener el máximo grado de información posible de un conjunto de datos.

Palabras claves: *Análisis de Conceptos Formales, Análisis de Redes Sociales, Minería de Datos, R (Lenguaje de programación)*

Índice

Índice de Ilustraciones

1 Introducción

1.1. Motivación

En los últimos años, las redes sociales incrementan notablemente su presencia en nuestras vidas. La huella de nuestra vida en los datos almacenados en ellas crece día a día. Facilitan la realización de ciertas labores las cuales antes se hacían de otra manera, tales como la de conocer nuevas personas o la búsqueda de trabajo. Nuestros hobbies, nuestros gustos, nuestras redes de amigos, nuestras ideas políticas quedan reflejadas en ellas. Buscar trabajo o conocer a nuevas personas son solo algunas de las muchas posibilidades que ofrecen. En cualquier caso, todo queda registrado en esa huella que crece.

Este crecimiento, en cuanto al uso cada vez mayor de las redes sociales, ha ido generando cada vez más y más datos. En la referencia [28]. se puede apreciar que desde los inicios de las redes sociales hasta el año 2019 se ha incrementado el número de usuarios activos a más de 2 billones, lo que produce cantidades ingentes de datos relativos a la actividad de cada persona en las redes sociales.

Dada esta gran cantidad de información se precisan de nuevos métodos que nos permiten extraer el conocimiento oculto en dichos datos. Actualmente existen múltiples algoritmos o herramientas para la extracción y análisis de los datos en redes sociales. Nuestro objetivo es la utilización de técnicas formales para la extracción de patrones que revelen información oculta que se encuentra presente en los datos y que no se ha descubierto.

Este trabajo fin de grado propone desarrollar una herramienta que nos permita extraer, analizar y presentar conocimiento oculto en los datos extraídos de redes sociales con una interfaz amigable al usuario en la que representar la información extraída.

1.2. Objetivos

El objetivo principal de este proyecto es desarrollar un herramienta para el análisis de las redes sociales en la cual se permita extraer conocimiento oculto en los datos textuales almacenados de dichas redes. Esta aplicación servirá para facilitar el estudio de las redes sociales con lo cual permitirá ofrecer una ayuda a problemas relacionadas con estas redes, tales como es la segmentación de usuarios, la identificación de posibles clientes, análisis de mercados, estudios sociológicos, etc.

Para cumplir con este objetivo podemos desglosarlo de la siguiente forma:

- Diseño de procesos ETC (Extracción, Transformación y Carga), de manera que podamos obtener los datos deseados de las redes social, realizar la manipulación necesaria e importarlos en una base de datos relacional para su posterior uso.

- Análisis de los datos obtenidos usando técnicas clásicas de la ciencia de datos y extracción de información haciendo uso de FCA[23].

- Diseño de una Interfaz web donde pueda obtener nuevos datos de las redes sociales en cuestión, que sea posible tratar con la información obtenida y que represente de forma visual los resultados obtenidos del análisis realizado.

1.3. Metodología

La metodología a usar en este proyecto es la metodología Agile, ya que es una de las más usadas actualmente en el sector del desarrollo que permite dar un enfoque iterativo en el que evaluar de forma constante los resultados.
Bajo esta metodología se hará uso del método Scrum, la cual se utiliza para desarrollar y abordar problemas complejos de una manera eficiente y sencilla.

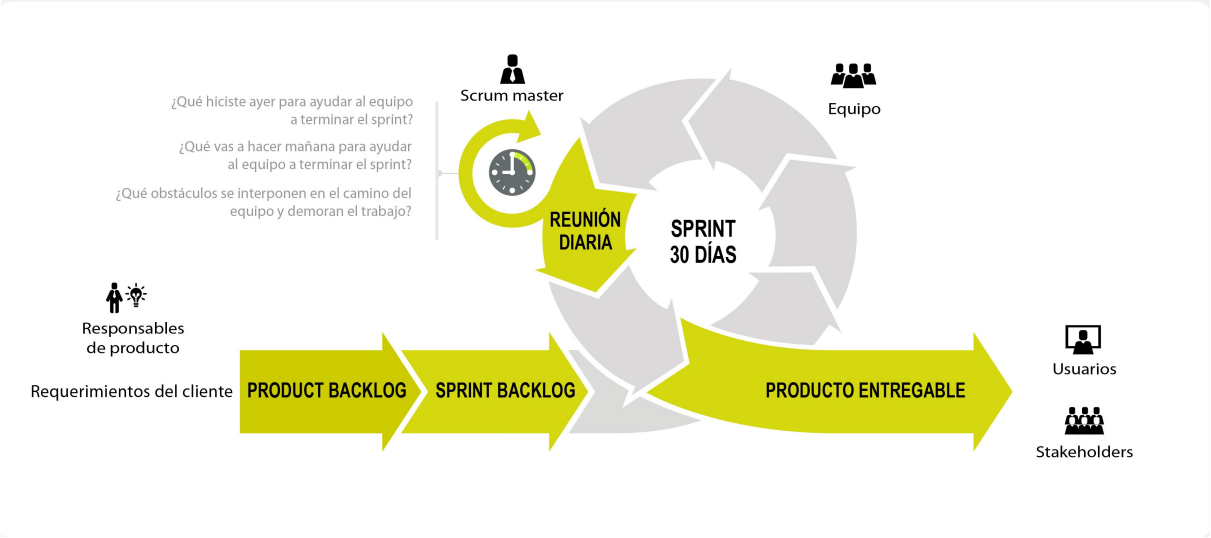

Figura 1.1: Metodología Scrum

Por como funciona el proceso Scrum es el motivo por el cual se implementa este proyecto, dado que se ejecutan ciclos de intervalos cortos de 2 a 4 semanas en la cual en cada iteración es preciso proporcionar un resultado completo, consiguiendo que finalmente se incremente de forma iterativa el producto final con el mínimo esfuerzo cuando el cliente lo solicite.

En este proyecto los roles con el método Scrum serían los siguientes:

- **Scrum Master**: Es el responsable de que las técnicas Scrum sean comprendidas y aplicadas en la organización. Es el mánager de Scrum, un líder que se encarga de eliminar impedimentos o inconvenientes que tenga el equipo dentro de un sprint. Este rol será compartido entre el tutor y co-tutor del trabajo fin de grado, Ángel Mora y Domingo López Rodríguez.

- **Desarolladores**: Son los responsables de realizar y entregar el producto en cada sprint. En este rol el encargado es el presente autor del trabajo fin de grado, Jean-Paul Beaudry.

1.4. Estructura del documento

La memoria de este proyecto se estructura de la siguiente manera:

- **Introducción**: En este apartado se establece la motivación y los objetivos que este proyecto pretende aportar, así mismo se detalla la metodología usada.
- **Catálogo de Requisitos**: En este apartado se detallan los diferentes tipos de requisitos necesarios para el funcionamiento de la aplicación, así mismo los casos de uso para mostrar más en detalle el proceso general a seguir en la ejecución de la aplicación.
- **Arquitectura y Entorno Tecnológico**: En este capítulo viene detallado la arquitectura necesaria para que el proyecto funcione en cualquier ordenador independientemente del sistema operativo.
- **Análisis de Conceptos Formales**: Este capítulo viene descrito la base de proyecto en el cual se utiliza FCA para la extracción de conocimiento de datos, se define las definiciones matemáticas necesarias para poder describir el funcionamiento de esta estructura matemática.
- **Análisis y extracción de la información**: Este capítulo describe el sistema encargado de realizar los procesos ETC (Extracción, Transformación y Carga) de los datos de las redes sociales para que se puedan manipular y tratar para su posterior uso. Así mismo también se describe como se analiza y extrae la información de las redes social usando FCA.
- **Implementación de la aplicación**: Este capítulo viene descrito la implementación de la aplicación web Shiny, bajo el cual se mostrará al usuario una interfaz amigable para obtener datos de las redes sociales y mostrar la información de interés mediante visualizaciones.
- **Conclusiones y Líneas Futuras**: Este capítulo recoge las conclusiones finales del proyecto, donde se comenta se ofrece una valoración general sobre el proyecto y se define como mejorar el proyecto en sí.

- **Referencias**: En este capítulo viene incluido un listado con todas las fuentes bibliográficas consultadas en este trabajo fin de grado.
- **Apéndices**: En este capítulo se incluyen tres apéndices, donde viene incluido el manual de instalación y el manual de usuario.

2

Catálogo de Requisitos

En este apartado vamos a detallar los requisitos imprescindibles que son necesarios para la implementación de la aplicación.

Existen tres tipos de requisitos:

- **Funcionales**. Describe el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.
- **No Funcionales**. Detalla los criterios que se pueden utilizar para juzgar el funcionamiento de un sistema, en lugar de comportamientos específicos.
- **De Información**. Describe la información que debe almacenar y gestionar el sistema para dar soporte a los procesos de negocio.

2.1. Requisitos Funcionales

Un requisito funcional puede abarcar desde la declaración abstracta de alto nivel de los requisitos de la aplicación hasta especificaciones detalladas de requisitos funcionales matemáticos. Para cada requisito funcional se establece un código formado especificado de la siguiente forma: **FR-XX**, donde la parte correspondiente a *XX* referencia a un código numérico formado por dos números.

- **FR-01: Obtención de datos mediante la API**
  - *El sistema debe ofrecer la posibilidad de obtener nuevos datos referentes a las publicaciones mediante la API que ofrece la red social.*

- **FR-02: Normalización de datos**

  - *Se debe seguir una normalización de los datos para que estos tengan el formato adecuado y sean consistentes.*

- **FR-03: Integración de los datos**

  - *El sistema integrará los datos obtenidos en el servidor de base de datos para que estos puedan ser usados posteriormente.*

- **FR-04: Lectura de los datos del servidor de base de datos**

  - *Será posible realizar consultas personalizadas por ciertos parámetros para la obtención de los datos localizados en el servidor de base de datos.*

- **FR-05: Análisis de los datos usando diversas técnicas**

  - *El sistema realizará varios tipos de análisis para la extracción de conocimiento de un conjunto de datos, entre las cuales se aplicará el método FCA para identificar esta información oculta.*

- **FR-06: Visualización de gráficos**

  - *Una vez realizado el análisis de los datos se imprimirá por pantalla gráficos en los cuales se mostrará de forma visual la información más relevante del conjunto de datos.*

- **FR-07: Selección de datos para la búsqueda de información**

  - *El sistema permitirá al usuario seleccionar y definir ciertos parámetros para la búsqueda de información en la aplicación.*

- **FR-08: Exportación de datos**

  - *El sistema permitirá la exportación de datos en diferentes apartados de la aplicación.*

2.2. Requisitos No Funcionales

Los requisitos no funcionales deben verse como aquellas características que permite valorar la calidad y el correcto desarrollo del proyecto. Para cada requisito no funcional se establece un código formado especificado de la siguiente forma: **NFR-XX**, donde la parte correspondiente a *XX* referencia a un código numérico formado por dos números.

- **NFR-01: Rendimiento**
  - *El sistema no debe tardar más de tres segundos en mostrar los resultados de una búsqueda.*
- **NFR-02: Escalabilidad**
  - *La base de datos deberá de disponer de un pool de conexiones configurables para que la aplicación sea escalable en función de los recursos hardware y software disponibles.*
- **NFR-03: Disponibilidad**
  - *El sitio web de la aplicación será accesible empleando cualquier navegador web. Además, todas las funcionalidades de la aplicación deberán ser accesibles a través de la interfaz de usuario.*
- **NFR-04: Seguridad**
  - *Todas las comunicaciones externas entre los servidores de datos, la aplicación y el cliente del sistema deben estar cifradas utilizando certificados SSL. Así mismo, garantizamos que el servidor esté en la nube con lo cual evitamos los problemas de seguridad que puedan haber si fuese un servidor local.*
- **NFR-05: Mantenibilidad**
  - *El código fuente que se implemente en el lenguaje de programación pertinente seguirá las reglas de estilo del mismo.*
- **NFR-06: Integridad de los datos**

- *Los datos se mantendrán correctos y completos tras ser modificados con sentencias INSERT, DELETE o UPDATE.*

- **NFR-07: Usabilidad**

  - *La aplicación cuenta con un diseño que se adapta al tamaño de pantalla de cualquier dispositivo.*

2.3. Requisitos de Información

Los requisitos de información sirven para determinar estructurar e identificar los datos. Para cada requisito de información se establece un código formado especificado de la siguiente forma: **FIR-XX**, donde la parte correspondiente a *XX* referencia a un código numérico formado por dos números.

- **FIR-01: Subreddit**

  - *Identificador Subreddit, Nombre, Número aproximado de subscriptores.*

- **FIR-02: Usuario**

  - *Identificador Usuario, Nombre, Descripción pública, Fecha Creación, Fecha añadida, Karma total, Karma de los comentarios, EsEmpleado, EsModerador, EsOro.*

- **FIR-03: Publicación**

  - *Identificador Publicación, Título, Descripción, Relación de Votos, Número de votos, Número total de recompensas, Puntuación, Fecha Creación, Fecha añadida, Enlace Publicación, Enlace Descripción, Dominio, Identificador Subreddit, Identificador Usuario.*

- **FIR-04: Moneda**

  - *Identificador Moneda, Nombre, Descripción, Precio Moneda, Precio Recompensa.*

- **FIR-05: Comentario**
  - *Identificador Publicación, Identificador Usuario, Estructura, Fecha Comentario, Puntuación, Comentario.*
- **FIR-06: Recompensa**
  - *Identificador Publicación, Identificador Moneda, Cantidad.*

2.4. Casos de uso

Los casos de uso representan una lista de acciones que suelen definir las interacciones entre un rol y un sistema para lograr un objetivo. Estos diagrama sirven para validar la arquitectura del sistema y verificar el sistema en desarrollo.

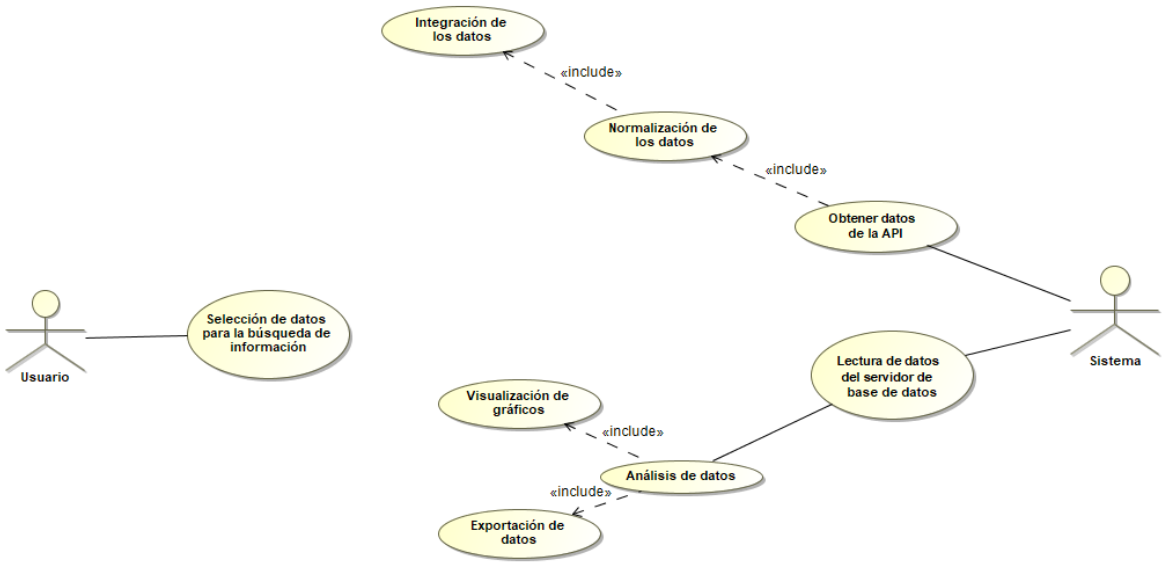

Figura 2.1: Casos de uso

Donde el usuario tiene la posibilidad de seleccionar los parámetros a buscar para la obtención de los datos. Del lado del sistema, este realiza la petición a la API para la obtención de datos y procede a la normalización e integración de los datos. Así mismo, el sistema lee los datos presentes en el servidor de base de datos y realiza un análisis sobre el conjunto de datos obtenido.

3

Arquitectura y Entorno Tecnológico

3.1. Arquitectura del Sistema

3.1.1. Virtualización de Contenedores: Docker

Docker[2] es una plataforma de contenedores de código abierto. Docker permite a los desarrolladores empaquetar aplicaciones en contenedores -componentes ejecutables estandarizados que combinan el código fuente de la aplicación con todas las bibliotecas y dependencias del sistema operativo (SO) necesarias para ejecutar el código en cualquier entorno.

Aunque los desarrolladores pueden crear contenedores sin Docker, esta herramietna hace que sea más fácil, sencillo y seguro crear, desplegar y gestionar contenedores. Es esencialmente un conjunto de herramientas que permite a los desarrolladores construir, desplegar, ejecutar, actualizar y detener contenedores utilizando comandos simples y automatización que ahorra trabajo.

Los contenedores ofrecen todas las ventajas de las máquinas virtuales, como el aislamiento de las aplicaciones, la escalabilidad rentable y la posibilidad de disponer de ellos.

Ventajas que proporciona Docker:

- **Gestión de dependencias**: Permite gestionar las dependencias desde el sistema operativo hasta detalles como las versiones de los paquetes de R y Latex.

- **Reproducibilidad**: Asegura que los análisis realizados son reproducibles.
- **Portabilidad**: Existe una gran portabilidad permitiendo que se pueda trasladar un contenedor a otras máquinas.
- **Espacio en disco**: Ocupa muy poco espacio en comparación con las maquinas virtuales.
- **Creación automática de contenedores**: Docker puede construir automáticamente un contenedor basado en el código fuente de la aplicación.
- **Versionado de contenedores**: Docker puede hacer un seguimiento de las versiones de una imagen de contenedor, retroceder a versiones anteriores y rastrear quién construyó una versión y cómo.
- **Reutilización de contenedores**: Los contenedores existentes pueden utilizarse como imágenes base, básicamente como plantillas para construir nuevos contenedores.
- **Bibliotecas de contenedores compartidas**: Los desarrolladores pueden acceder a un registro de código abierto que contiene miles de contenedores aportados por los distintos usuarios.

3.1.2. Bases de datos

3.1.2.1. MariaDB

MariaDB[3] es un sistema de gestión de bases de datos relacionales de código abierto. Al igual que otras bases de datos relacionales, MariaDB almacena los datos en tablas formadas por filas y columnas. Los usuarios pueden definir, manipular, controlar y consultar los datos mediante el lenguaje de consulta estructurado SQL.

MariaDB esta basado en MySQL, por lo que ambas comparten muchas características y opciones de diseño.

Ventajas de MariaDB con respecto a MySQL:

- MariaDB tiene 12 nuevos motores de almacenamiento mientras que MySQL tiene menos motores.
- MariaDB tiene un grupo de conexiones más grande que soporta hasta más de 200.000 conexiones, mientras que MySQL tiene un grupo de conexiones más pequeño.
- En MariaDB la replicación de los datos es más rápida mientras que en MySQL es más lenta.
- MariaDB es de código abierto, mientras que MySQL utiliza código propietario en su edición Enterprise.
- Comparativamente MariaDB es más rápido que MySQL.

3.1.2.2. phpMyAdmin

phpMyAdmin[4] es una herramienta de código abierto basada en PHP que permite administrar bases de datos MySQL y MariaDB en línea. Ofrece una interfaz web amigable al usuario bajo el cual se puede gestionar todo lo relativo a una base de datos y permite ejecutar consultas de Lenguaje de Consulta Estructurado (SQL).

3.2. Lenguaje de Programación R

R[8] es un lenguaje de programación pensado para la computación estadística y la generación de gráficos.

R ofrece una gran variedad de herramientas (paquetes) y técnicas, las cuales permiten realizar cualquier tipo de análisis estadístico, hacer uso de algoritmos de inteligencia artificial o realizar análisis de datos en conjuntos de datos.

Los puntos fuertes de R son:

- **Análisis de datos**: R fue escrito por estadísticos para estadísticos, por lo que está diseñado ante todo como un lenguaje para el análisis estadístico y de datos. Gran parte de la investigación de vanguardia en aprendizaje automático se realiza en R, y

cada semana se añaden paquetes a CRAN que implementan estos nuevos métodos. Además, muchos modelos en R pueden exportarse a otros lenguajes de programación como C, C++, Python, etc.

- **Visualización de datos**: aunque el paquete básico de gráficos de R es completo y potente, las bibliotecas adicionales como ggplot2 y lattice hacen que R sea el lenguaje de referencia para los enfoques de visualización de datos más potentes.

3.3. Herramientas y Entornos de Desarrollo

3.3.1. RStudio

RStudio[7] es un entorno de desarrollo integrado que permite interactuar con R más fácilmente. RStudio es en realidad un complemento del lenguaje de programación R, en el cual toma el software R y le añade una interfaz gráfica muy fácil de usar.

3.3.2. Overleaf

Overleaf[9] es una herramienta de escritura y publicación colaborativa en línea de látex que hace que todo el proceso de escritura, edición y publicación de documentos científicos sea mucho más rápido y sencillo.

4

Análisis de Conceptos Formales

Análisis de Conceptos Formales o FCA[23] es una teoría matemática basada en la teoría de retículos y en la lógica, está orientada para el análisis de datos ya que permite encontrar información oculta la cual no se puede obtener con otros métodos de aprendizaje computacional.

4.1. Definiciones previas

Antes de dar la definición formal de FCA, es necesario recordar algunos conceptos de Álgebra.

Para las definiciones que aparecen a continuación solo consideramos los conjuntos finitos de objetos.

Definición 4.1.1 *Dados dos conjuntos A y B, llamamos* ***relación binaria*** *a un subconjunto R del producto cartesiano $A \times B$. Por tanto, los elementos de R serán pares (a, b) donde $a \in A, b \in B$.*

También se puede escribir de la forma aRb en vez de $(a, b) \in R$.

Si $A = B$ entonces $R \subseteq A \times A$ es llamado una relación binaria sobre el conjunto A.

Definición 4.1.2 *Se dice que R una relación binaria R sobre un conjunto A es una* ***relación de orden****, si satisface las siguientes condiciones para todos los elementos $a, b, c \in A$:*

- *aRa (reflexividad)*

- aRb *y* $bRa \Longrightarrow a = b$ *(simetría)*
- aRb *y* $bRc \Longrightarrow aRc$ *(transitividad)*

Si dados $a, b \in A$, siempre tenemos bien aRb ($a \leq b$) o bRa ($b \leq a$) entonces se denomina relación de orden **total**. En caso contrario, se denomina **parcial**.

Se usa el símbolo $\leq$ para el orden parcial, y en caso de $a \neq b$ y $a \leq b$ se escribe $a < b$. Se lee $a \leq b$ como "a es menor o igual a b".

Definición 4.1.3 *Un conjunto parcialmente ordenado o poset es un par* $(P, \leq)$*, donde* P *es un conjunto y* $\leq$ *es una relación de orden parcial sobre* P*.*

Definición 4.1.4 *Dado un conjunto parcialmente ordenado* $(P, \leq)$*, un elemento* a *es* ***anterior*** *a* b*, si* $a \leq b$ *y no un elemento* c *tal que* $a \leq c \leq b$ *. En este caso,* b *es* ***posterior*** *a,* y *se escribe como* $a \prec b$*.*

Todo conjunto parcialmente ordenado finito $(P, \leq)$ se puede dibujar como un diagrama de Hasse. Elementos de P se representan mediante círculos pequeños en el plano. Si $a \prec b$, el círculo correspondiente a a se representa en un punto más alto que el círculo correspondiente a b, y los dos círculos están conectados mediante una línea.

Definición 4.1.5 *Sea* $(P, \leq)$ *un conjunto parcialmente ordenado y* A *un subconjunto de* P*. Una* ***cota inferior*** *de* A *es un elemento* l *de* P *con* $l \leq A$ *para todo* $a \in A$*. Una* ***cota superior*** *de* A *se define de forma análoga. Se denomina ínfimo de* A *a la mayor de estas cotas inferiores y se denota como* $\inf A$ *o* $\bigwedge A$*. De forma análoga, se denomina supremo de* A *a la menor de estas cotas superiores y se denota como* $\sup A$ *o* $\bigvee A$*.*

Para $A = \{a, b\}$ también se puede escribir $a \wedge b$ para $\inf A$ y $a \vee b$ para $\sup A$.

Definición 4.1.6 *Un conjunto parcialmente ordenado* $\boldsymbol{L} = (L, \leq)$ *es un retículo, si para cada par de elementos* a *y* b *en* L *el supremo* $a \vee b$ *y el ínfimo* $a \wedge b$ *siempre existe.* $\boldsymbol{L}$ *es un retículo* ***completo*** *si todo subconjunto* X *tiene supremo e ínfimo. Para cada retículo completo* $\boldsymbol{L}$ *existe un único elemento supremo* $\bigvee L$*, llamado el* ***elemento identidad*** *del retículo, denotado como* $\boldsymbol{1}_L$*. De forma análoga, el ínfimo* $\boldsymbol{0}_L$ *se llama el* ***elemento cero****.*

A continuación, detallamos la base para el funcionamiento de FCA y los contextos formales.

Definición 4.1.7 *Sea $\varphi : P \to Q$ y $\psi : P \to Q$ sean funciones entre dos conjuntos parcialmente ordenados $(P, \leq)$ y $(Q, \leq)$. Esta pareja de funciones se llama una* ***conexión de Galois*** *entre conjuntos ordenados si:*

- $p_1 \leq p_2 \Longrightarrow \varphi(p_1) \geq \varphi(p_2)$
- $p_1 \leq p_2 \Longrightarrow \psi(p_1) \geq \psi(q_2)$
- $p \leq \psi\varphi p \Longrightarrow q \leq \varphi\psi q$

Definición 4.1.8 *Un retículo de concepto de un contexto formal $\mathbb{K} = (G, M, I)$ es una estructura definida como $\langle \mathbb{K}, \leq \rangle$, dónde $\langle \mathbb{K}, \leq \rangle$ es una colección de todos los conceptos formales.*

Definición 4.1.9 *Un* ***concepto formal*** *$\mathbb{K} = (G, M, I)$ consiste en dos conjuntos G y M y una relación I entre G y M. Los elementos de G se llaman* ***objetos*** *y los elementos de M se llaman* ***atributos*** *del contexto. La notación gIm o $(g, m) \in I$ significa que el objeto g tiene el atributo m.*

Teorema 4.1.1 *El conjunto de todos los conceptos formales de un contexto $\mathbb{K}$ junto con relación de orden I forman un retículo completo, llamado el* ***retículo de conceptos*** *de $\mathbb{K}$ y se denota por $\underline{\mathfrak{B}}(\mathbb{K})$.*

Definición 4.1.10 *Para $A \subseteq G$, sea*

$$A' := \{m \in M | (g, m) \in I \text{ para todo } g \in A\}$$

y, para $B \subseteq M$, sea

$$B' := \{g \in G | (g, m) \in I \text{ para todo } m \in B\}$$

Estos operadores se llaman **operadores de derivación** para $\mathbb{K} = (G, M, I)$

Teorema 4.1.2 *Sea (G, M, I) un contexto formal, los subconjuntos $A, A_1, A_2 \subseteq G$ y $B \subseteq M$ tenemos que:*

- $A_1 \subseteq A_2 \Longleftrightarrow A_2^{'} \subseteq A_1^{'}$
- $A \subseteq A^{''}$
- $A^{'} = A^{'''} (por\ tanto, A^{''''} = A^{''})$
- $(A_1 \cup A_2)^{'} = A_1^{'} \cap A_2^{'}$
- $A \subseteq B^{'} \Longleftrightarrow B \subseteq A^{'} \Longleftrightarrow A \times B \subseteq I$

Para el subconjunto de atributos también se aplica propiedades parecidas.

Definición 4.1.11 *Un* ***operador de cierre*** *sobre el conjunto G es función $\varphi : 2^G \longrightarrow 2^G$ con las siguientes propiedades:*

- $\varphi\varphi X = \varphi X$ *(**idempotencia**)*
- $X \subseteq \varphi X$ *(**extensión**)*
- $X \subseteq Y \Longrightarrow \varphi X \subseteq \varphi Y$ *(**monotonicidad**)*

Para un operador de cierre φ el conjunto φX se llama el ***cierre*** *de X.*

Un subconjunto $X \subseteq G$ se dice que esta ***cerrado*** *si $\varphi X = X$*

Definición 4.1.12 *Un* ***contexto formal*** *(A, B) con $A \subseteq G$, $B \subseteq M$ tales que $A' = B, B' = A$. En particular, B es un conjunto cerrado para el operador de cierre resultado de componer los operadores de derivación.*

Definición 4.1.13 *Por cada par de conceptos formales (A_1, B_1) y (A_2, B_2) de un contexto formal su* ***mayor subconcepto común*** *se define como:*

$$(A_1, B_1) \wedge (A_2, B_2) = (A_1 \cap A_2, (B_1 \cup B_2)^{''}).$$

El ***menor superconcepto común*** *de (A_1, B_1) y (A_2, B_2) viene dado por*

$$(A_1, B_1) \vee (A_2, B_2) = ((A_1 \cup A_2)^{''}, B_1 \cap B_2).$$

Se puede llamar supremo a "menor subconcepto común" e ínfimo a "mayor subconcepto común".

Definición 4.1.14 *Un subconjunto $X \subseteq L$ de un retículo $(L, \leq)$ se llama* ***supremamente denso*** *si cualquier elemento del retículo $v \in L$ se puede representar como*

$$\upsilon = \bigvee\{x \in X | x \leq \upsilon\}$$

de forma análoga para subconjuntos ***infimamente denso***

$$\upsilon = \bigwedge\{x \in X | x \leq \upsilon\}$$

4.2. Definición formal de Análisis de Conceptos Formales

Teorema 4.2.1 ***Teorema Básico de Análisis de Conceptos Formales****.*
El retículo de conceptos $\underline{\mathfrak{B}}(G, M, I)$ *es un retículo completo. Para conjuntos arbitrarios de conceptos formales*

$$\{(A_j, B_J) | j \in J\} \subseteq \underline{\mathfrak{B}}(G, M, I)$$

sus ínfimos y supremos vienen dados de la siguiente forma:

$$\bigwedge_{j \in J}(A_j, B_j) = (\bigcap_{j \in J} A_j, (\bigcup_{j \in J} B_j)'') \ ,$$
$$\bigvee_{j \in J}(A_j, B_j) = ((\bigcup_{j \in J} A_j)'', \bigcap_{j \in J} B_j) \ .$$

Un retículo completo L *es isomorfo al retículo* $\underline{\mathfrak{B}}(G, M, I) \iff$ *si existe una función biyectiva* $\gamma : G \to V$ *y* $\mu : M \to V$ *tal que* $\gamma(G)$ *es supramemente denso en* $\mathbf{L}$*,* $\mu(M)$ *es infimamente denso en* $\mathbf{L}$*, y* $gIm \iff \gamma g \leq \mu m$ *para todo* $g \in G, m \in M$*. En particular,* $\mathbf{L}$ *es isomorfa a* $\underline{\mathfrak{B}}(L, L, \leq)$*.*

Definición 4.2.1 *Una expresión de la forma* $A \to B$ *con* $A, B \in M$ *es una* ***implicación*** *en el contexto formal* $K = (G, M, I)$ *si* $B \subseteq A''$*, es decir, si cada objeto que posea los atributos del conjunto* A*, también posee los atributos del conjunto* B*.*

4.3. Ejemplo

Sea la siguiente tabla un contexto formal (G, M, I), donde G es el conjunto de los objetos $\{o1, o2, o3, o4, o5, o6\}$, M es el conjunto de los atributos $\{a1, a2, a3, a4, a5\}$ y $I \subseteq G \times M$ es la relación binaria entre G y M.

	a1	a2	a3	a4	a5
o1		x	x		x
o2	x		x	x	
o3	x	x	x	x	
o4	x			x	
o5	x	x	x	x	
o6	x		x	x	

Tabla 4.1: Objetos / Atributos

Usando los operadores de derivación podemos calcular las imágenes de conjuntos de objetos y atributos, algunos ejemplos:

- $\{o2\}' = \{a1, a3, a4\}$
- $\{a3\}' = \{a1, a2, a3, a5, a6\}$
- $\{o3, o5\}' = \{a1, a2, a3, a4\}$
- $\{a3, a4\}' = \{o2, o3, o5, o6\}$

Los operadores de derivación pueden ser iterados, es decir, si partimos de un conjunto $A \subseteq G$, es posible obtener A' que es un subconjunto de M. Si aplicamos el segundo operador de derivación, obtenemos $(A')'$ o A'', que es un conjunto de objetos. Si seguimos con el proceso de iteración, podemos obtener A''', A'''' y así sucesivamente.

- $\{o3\}'' = \{a1, a2, a3, a4\}' = \{o3, o5\}$
- $\{o1, o3, o5\}'' = \{a2, a3\}' = \{o1, o3, o5\}$
- $\{a3, a4\}'' = \{o2, o3, o5, o6\}' = \{a1, a3, a4\}$
- $\{a3\}'' = \{o1, o2, o3, o5, o6\}' = \{a3\}$

Con esto se construye el siguiente retículo de conceptos:

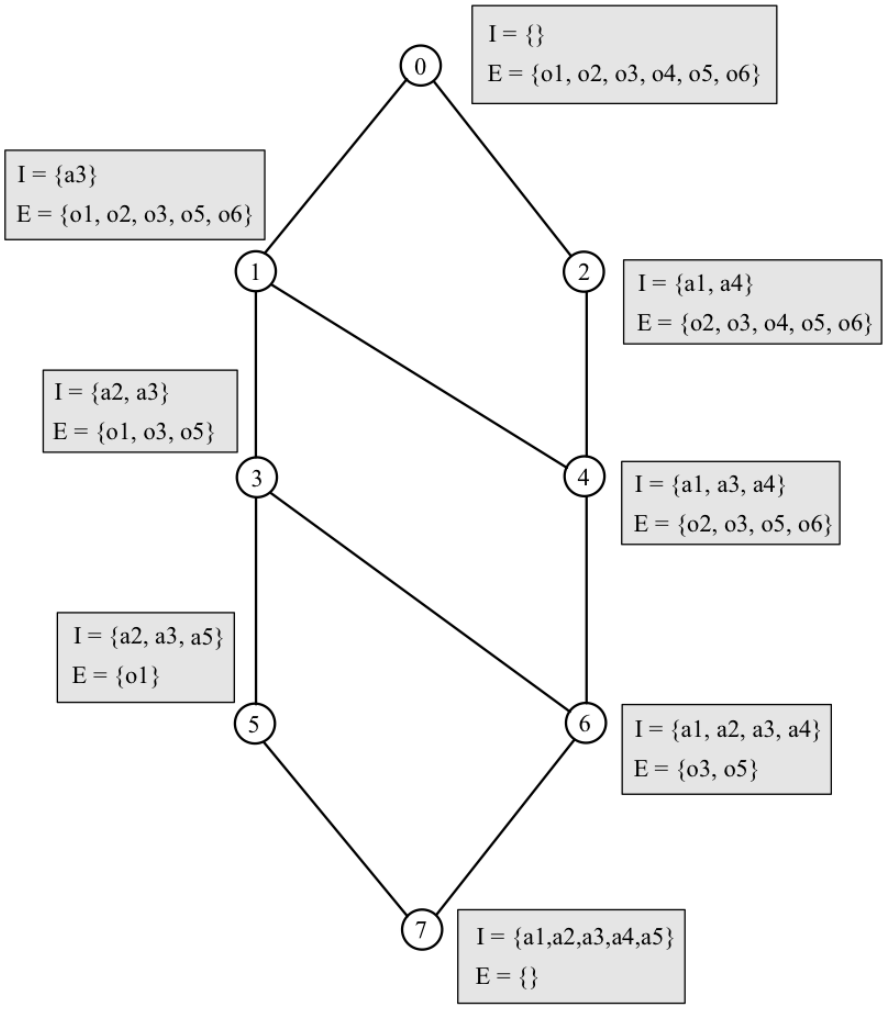

Figura 4.1: Retículo de conceptos del Ejemplo 1

Donde I es el conjunto de intenciones y E es el conjunto de extensiones.

4.4. Ejemplo con fcaR

fcaR[16] Es un paquete de R que permite aplicar el método de Análisis de Conceptos Formales Difuso desde R, este paquete permite trabajar con contextos formales, extraer su retículo de conceptos y generar implicaciones a partir de ella.

Si abrimos el entorno de desarrollo RStudio y introducimos la siguiente matriz

```
> x <- matrix(c(0, 1, 1, 0, 1,
              1, 0, 1, 1, 0,
              1, 1, 1, 1, 0,
```

```
                1, 0, 0, 1, 0,
                1, 1, 1, 1, 0,
                1, 0, 1, 1, 0),
              nrow = 6,
              ncol = 5,
              dimnames = list(c("o1","o2","o3","o4","o5","o6"), c("a1","a2","
    a3","a4","a5")))
```

Podemos generar un contexto formal de dicha matriz con los siguientes comandos:

```
> fc <- FormalContext$new(x)
> fc$find_concepts()
> fc$concepts$plot()
```

Al ejecutar el último comando se puede apreciar que se obtiene el mismo resultado que la figura anterior:

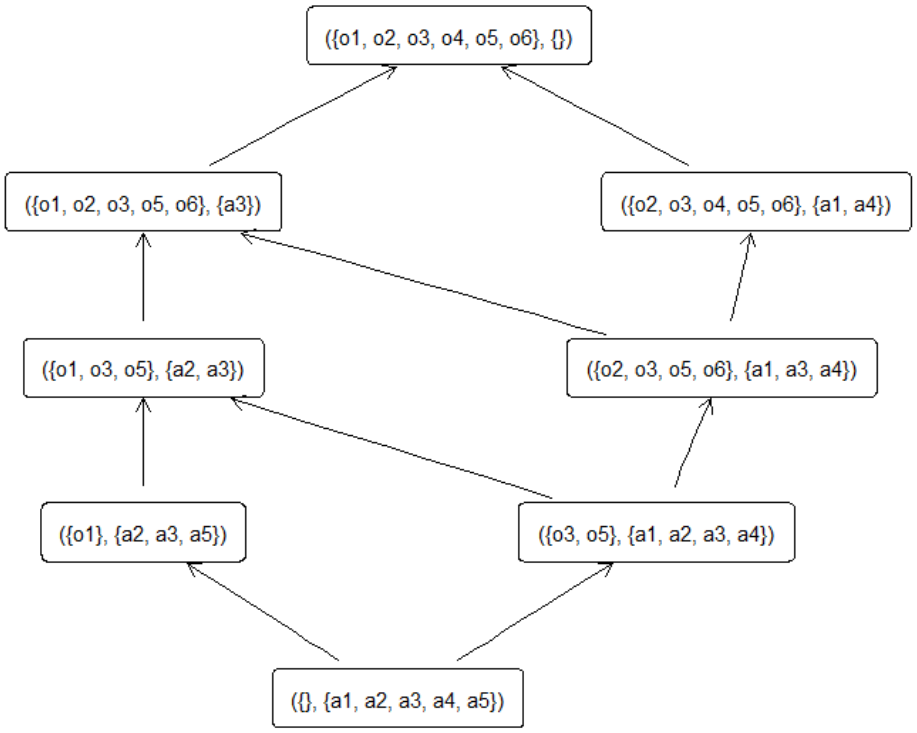

Figura 4.2: Retículo de conceptos generado por fcaR del Ejemplo 1

Una vez generado el contexto formal con sus correspondientes conceptos es posible generar implicaciones del retículo de conceptos, es decir, al generar las implicaciones podemos encontrar relaciones entre los datos, si se ejecuta los siguientes comandos bajo el contexto formal generado anteriormente:

```
> fc$find_implications()
> fc$implications$print()
```

Se obtienen 4 implicaciones

```
Implication set with 4 implications.
Rule 1: {a5} -> {a2, a3}
Rule 2: {a4} -> {a1}
Rule 3: {a2} -> {a3}
Rule 4: {a1} -> {a4}
```

donde nos indica que:

- Si en un objeto se da el atributo $a5$, entonces también deben aparecer los atributos $\{a2, a3\}$.
- Si en un objeto se da el atributo $a4$, entonces también debe aparecer el atributo $\{a1\}$.
- Si en un objeto se da el atributo $a2$, entonces también debe aparecer el atributo $\{a3\}$.
- Si en un objeto se da el atributo $a1$, entonces también debe aparecer el atributo $\{a4\}$.

5

Análisis y extracción de la información

5.1. Extracción

En este apartado vamos a comentar como extraemos la información haciendo uso de procesos ETC.

Extracción, Transformación y Carga o ETC es un proceso de integración de datos desde múltiples fuentes con la finalidad de construir un almacén de datos.

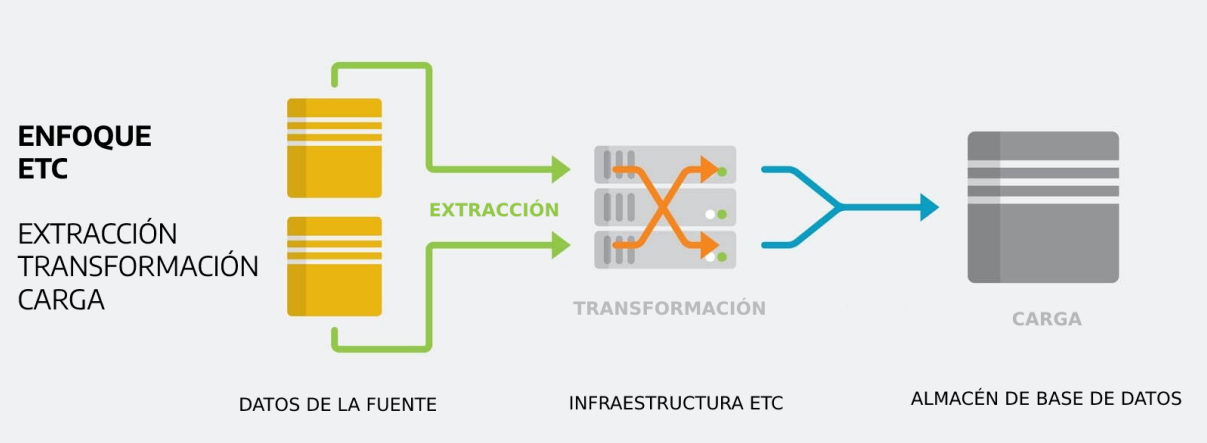

Figura 5.1: Proceso ETC

Este proceso se divide en tres fases diferentes:

- **Extracción**: Esta fase se encarga de obtener los datos deseados desde los múltiples fuentes.

- **Transformación**: Fase en el cual se transforma los datos obtenidos para mantener una consistencia entre los datos y que sea más fácil de manejar.
- **Carga**: Fase en la que se carga los datos transformados en un servidor o almacén de base de datos.

En este caso para el proyecto la extracción de los datos se hará realizando peticiones HTTP de tipo GET a la API de las redes sociales.

5.1.1. Reddit

Reddit[10] es una plataforma en la cual se combina contenidos web, noticias sociales, foros y una red social con una gran multitud de usuarios de diferentes partes del mundo.

La forma en la que se destacan las publicaciones es mediante el sistema de votos, en los cuales cualquier usuario registrado puede dar su voto positivo o negativo a cada publicación , esto permite ver rápidamente aquellas publicaciones que son repudiadas por la comunidad (publicación que tiene una gran cantidad de votos negativos) o que son tendencia en el presente día o las publicaciones con mayor número de votos positivos en un periodo de tiempo (publicaciones que han sido tendencia en el periodo de tiempo especificado).

Esto tiene como resultado que Reddit sea uno de los sitios dónde se publican la gran mayoría de memes y sensaciones virales de todo Internet.

A continuación detallamos las peticiones que se utilizan para la obtención de datos de esta red social:

- GET **/user/username/about**
  - **Descripción:** Permite obtener información sobre un usuario que no ha sido eliminado o baneado.
  - **Parámetros:**:
    - *username*: donde se especifica el nombre del usuario a buscar en cuestión.

- GET **[/r/subreddit]/comments/article**
  - **Descripción:** Permite obtener los comentarios de un artículo.
  - **Parámetros:**:
    - *depth*: especifica la profundidad máxima de los subhilos.
    - *limit*: limita el número de comentarios a solicitar, el mínimo de elementos que se pueden obtener es 25 y el máximo 100.
    - *sort*: especifica como se ordena los comentarios, es una opción de la lista (confidence, top, new, controversial, old, random, qa, live).
    - *sr_detail*: expande los subreddits.
    - *article*: código identificador único de cada artículo.
    - *subreddit*: nombre identificador único del subreddit.
- GET **[/r/subreddit]/hot**
  - **Descripción:** Permite obtener las publicaciones que son tendencia a la hora de realizar la petición.
  - **Parámetros:**:
    - *limit*: limita el número de publicaciones a solicitar, el mínimo de elementos que se pueden obtener es 25 y el máximo 100.
    - *show*: con la opción "all" muestra todas las publicaciones.
    - *sr_detail*: expande los subreddits.
    - *subreddit*: nombre identificador único del subreddit.
- GET **[/r/subreddit]/new**
  - **Descripción:** Permite obtener las publicaciones que son nuevos.
  - **Parámetros:**:
    - *limit*: limita el número de publicaciones a solicitar, el mínimo de elementos que se pueden obtener es 25 y el máximo 100.
    - *show*: con la opción "all" muestra todas las publicaciones.
    - *sr_detail*: expande los subreddits.

- *subreddit*: nombre identificador único del subreddit.

- GET **[/r/subreddit]/rising**

  - **Descripción:** Permite obtener las publicaciones en los cuales hay mucha actividad reciente con respecto a los comentarios o votos a la hora de realizar la petición.
  - **Parámetros:**:
    - *limit*: limita el número de publicaciones a solicitar, el mínimo de elementos que se pueden obtener es 25 y el máximo 100.
    - *show*: con la opción "all" muestra todas las publicaciones.
    - *sr_detail*: expande los subreddits.
    - *subreddit*: nombre identificador único del subreddit.

- GET **[/r/subreddit]/top**

  - **Descripción:** Permite obtener las publicaciones en los cuales han tenido los mayores votos durante un periodo de tiempo.
  - **Parámetros:**:
    - *t*: limita las publicaciones a un plazo de tiempo determinado, permite como opción lo siguiente: hour, day, week, month, year, all.
    - *limit*: limita el número de publicaciones a solicitar, el mínimo de elementos que se pueden obtener es 25 y el máximo 100.
    - *show*: con la opción "all" muestra todas las publicaciones.
    - *sr_detail*: expande los subreddits.
    - *subreddit*: nombre identificador único del subreddit.

- GET **[/r/subreddit]/sort**

  → **[/r/subreddit]/top**

  → **[/r/subreddit]/controversial**

  - **Descripción:** Permite obtener ordenar y obtener las publicaciones en un cierto orden.

- **Parámetros:**:
  - *t*: limita las publicaciones a un plazo de tiempo determinado, permite como opción lo siguiente: hour, day, week, month, year, all.
  - *limit*: limita el número de publicaciones a solicitar, el mínimo de elementos que se pueden obtener es 25 y el máximo 100.
  - *show*: con la opción "all" muestra todas las publicaciones.
  - *sr_detail*: expande los subreddits.
  - *subreddit*: nombre identificador único del subreddit.

NOTA: Remarcar que no es necesario especificar el subreddit en cuestión, es decir, en las peticiones GET si no se quiere delimitar la búsqueda a un subreddit se quita el siguiente texto de la petición "[/r/subreddit]".

Con la información detallada de cada petición GET ya se puede realizar las peticiones a la API de Reddit para ir obteniendo la información que el usuario solicita.
Por ejemplo, se desea realizar la siguiente petición "*Quiero obtener las 5 publicaciones con mayor votos en el subreddit WorldNews*", la petición quedaría de la siguiente forma:

```
1 https://www.reddit.com/r/WorldNews/top?limit=5
```

Para obtener los datos de una forma estructurada la cual se puede procesar por la aplicación de una forma sencilla posteriormente, es necesario añadir el parámetro ".json" a la petición GET que se realice. Esto solicita a la API de Reddit que devuelva el resultado de una petición en un fichero JSON, permitiendo solicitar los datos deseados de forma eficiente, ya que se devuelve el resultado con una estructura y el tiempo de respuesta para obtener dicho fichero es muy pequeño.
Por ejemplo, para el caso anterior se quedaría de la siguiente forma:

```
1 https://www.reddit.com/r/WorldNews/top.json?limit=5
```

Para este proyecto se utiliza el paquete de R *RedditExtractoR* para obtener los comentarios de una publicación y se utiliza el paquete de R *jsonlite* para realizar peticiones GET para obtener el resto de la información correspondiente a las publicaciones o usuarios.

De la información obtenida se procesa y se guarda la correspondiente información según el tipo:

- **Publicaciones:** identificador de la publicación, identificador del subreddit, identificador del autor, título, ratio de votos, total de los premios recibidos, puntuación, fecha creada, fecha añadida, enlace a la publicación, enlace a una web externa si la publicación lo tiene, versión reducida del enlace a la web externa.
- **Comentarios:** identificador de la publicación, identificador del usuario, estructura del comentario, fecha del comentario, puntuación del comentario, texto del comentario.
- **Monedas:** identificador de la moneda, nombre, descripción, precio, recompensa.
- **Premios:** identificador de la publicación, identificador de la moneda, cantidad.
- **Usuarios:** identificador del usuario, nombre, fecha creada, fecha añadida, karma total, karma de los comentarios, descripción pública.

Algunos procesos que se realizan para mantener una consistencia en los datos pueden ser los siguientes:

- Quitar caracteres no alfa-numéricos en los campos de texto.
- Cambiar el formato de todas las fechas al formato iso.
- Obtener los identificadores de cada usuario o publicación y modificarlos para su correcta relación.
- Comprobar que no se haya insertado una publicación anteriormente, si es el caso no se tratan los datos ni se insertan en el servidor de base de datos.

Una vez realizada la petición y se hayan tratado los datos de forma correcta, esta información se inserta en el servidor de base de datos, donde esta sirve para almacenar toda la información correspondiente a cualquier petición que se realice a la API de Reddit, esto con el tiempo se va convirtiendo en una base de datos con un tamaño considerado, pero permite obtener mejores resultados a la hora de extraer información, dado que cuantos más datos existan en el servidor de bases de datos, más y mejores relaciones entre los datos se pueden encontrar y obtener.

La base de datos correspondiente a la red social Reddit tiene la siguiente estructura.

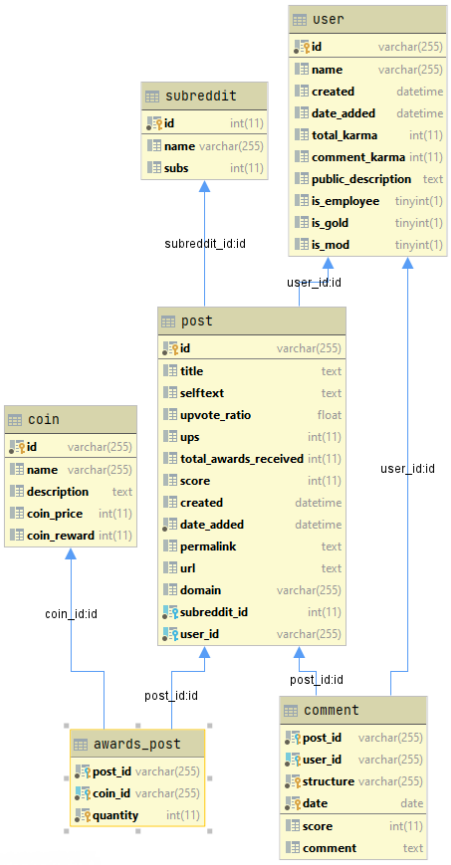

Figura 5.2: Estructura de la base de datos sna_reddit

5.2. Análisis

Se parte de un conjunto de datos con un número de registros entre 1000 y 10000.

```
> data <- db.fetch_all("SELECT post.*, subreddit.name as subreddit, user.
    name as username
          FROM post JOIN subreddit ON post.subreddit_id = subreddit.id
    JOIN user ON post.user_id = user.id
          WHERE date(post.created) >= '2021-01-14' AND date(post.created)
     <= '2021-09-12'
          ORDER BY post.created DESC LIMIT 10000 ;")
```

Normalizamos los datos usando las siguientes funciones:

```
> data_df <- data %>% select(upvote_ratio, total_awards_received, score,
    domain, subreddit)
> data_df[sapply(data_df, is.integer)] <- lapply(data_df[sapply(data_df, is
    .integer)], as.numeric)
> data_df[sapply(data_df, is.character)] <- lapply(data_df[sapply(data_df,
    is.character)], as.factor)
> data_df[is.na(data_df)] <- 0
> data_df <- discretizeDF(data_df, default = list(method = "interval",
    breaks = 3, labels = c("Low", "Medium", "High")))
```

Y creamos una matriz binaria en base a estos datos:

```
> create_binary_matrix <- function(cols, data){
    length_cols <- length(colnames(data))
    length_rows <- length(rownames(data))
    output <- data.frame(matrix(ncol = length(cols), nrow = 0))
    colnames(output) <- cols

    for(i in 1:length_rows){
        for(k in 1:length_cols){
```

```
            colname <- paste0(colnames(data)[k], "-", data[i, k])
            other <- paste0(colnames(data)[k], "-other")
            if(colname %in% cols){
                output[i, colname] <- 1
            } else{
                output[i, other] <- 1
            }
        }
    }
    output[is.na(output)] <- 0
    output[sapply(output, is.numeric)] <- lapply(output[sapply(output, is.
    numeric)], as.factor)
    return(output)
}

> top_n_domains <- data_df %>%
  select(domain) %>%
  group_by(domain) %>%
  summarise(n = n()) %>%
  filter(n > quantile(n, SHINY_FREQ_PERCENTAGE)) %>%
  add_row(domain = "other", n=0) %>%
  mutate_if(is.character,
        str_replace_all, pattern = "self", replacement = "subreddit")

> top_n_subreddits <- data_df %>%
  select(subreddit) %>%
  group_by(subreddit) %>%
  summarise(n = n()) %>%
  filter(n > quantile(n, SHINY_FREQ_PERCENTAGE)) %>%
  add_row(subreddit = "other", n = 0)

> cols <- c(paste("upvote_ratio", names(table(data_df$upvote_ratio)), sep =
    "-"),
      paste("total_awards_received", names(table(data_df$total_awards_
    received)), sep = "-"),
      paste("score", names(table(data_df$score)), sep = "-"),
      paste("domain", top_n_domains$domain, sep = "-"),
```

```
        paste("subreddit", top_n_subreddits$subreddit, sep = "-"))

> data_matrix <- create_binary_matrix(cols, data_df); data_matrix[1:5, 1:5]
```

	upvote_ratio-Low	upvote_ratio-Medium	upvote_ratio-High	total_awards_received-Low	total_awards_received-Medium
1	0	1	0	1	0
2	0	0	1	1	0
3	0	0	1	1	0
4	0	0	1	1	0
5	0	0	1	1	0

Dado que crear un contexto formal con tantas filas consume muchos recursos de memoria del servidor o equipo, es necesario buscar una alternativa para poder reducir esta carga. Se precisa buscar otra solución en la cual se puede trabajar con un número razonable de reglas sin un exceso en el consumo de memoria.

La solución propuesta es crear un objeto de tipo *transactions* de la matriz binaria que se ha creado en el punto anterior.

```
> transactions <- as(data_matrix, "transactions")
```

Se aplica el algoritmo apriori con unos parametros $support = 0{,}5$ y $confidence = 1$ y con esto conseguimos un número determinado de reglas:

```
> rules <- apriori(transactions, parameter = list(support = 0.5, confidence
    = 1, maxlen = 5));
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
    maxlen target   ext
          1     0.1     1 none FALSE            TRUE       5     0.5       1
       5  rules TRUE

```

```
Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 5000

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[265 item(s), 10000 transaction(s)] done [0.07s].
sorting and recoding items ... [137 item(s)] done [0.02s].
creating transaction tree ... done [0.01s].
checking subsets of size 1 2 3 4 done [39.08s].
writing ... [3957333 rule(s)] done [1.26s].
creating S4 object  ... done [1.33s].
```

Una vez ejecutado el algoritmo *Apriori* para reducir el número de reglas podemos trabajar con las reglas redundantes o las reglas significantes:

```
> rules.non_redundant <- rules[!is.redundant(rules)]; rules.non_redundant
set of 50 rules

> rules.significant <- rules[is.significant(rules, transactions)];
set of 82851 rules
```

En este caso vemos que obtenemos muy pocos reglas no redundantes y por tanto se trabajaría con estas reglas. Ahora el siguiente paso es convertir estas reglas a una matriz dispersa y posteriormente convertir esta matriz en un objeto de tipo *transactions*

```
> transactions_reduced <- transactions(as(items(rules.non_redundant), "
    ngCMatrix")); transactions_reduced
transactions in sparse format with
 50 transactions (rows) and
 265 items (columns)
```

Una vez hecho procedemos a crear un contexto formal con estas transacciones ya reducidas.

```
> fc <- FormalContext$new(transactions_reduced)
```

Una vez creado el contexto procedemos al análisis de la información del conjunto de datos.

Primero, es posible calcular los conceptos de este conjunto de datos, la cual hace uso del algoritmo difuso *NextClosure* [27]:

```
> fc$find_concepts()

> fc$concepts$size()
[1] 67

> head(fc$concepts)
A set of 6 concepts:
1: ({1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
    21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
    39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}, {})
2: ({14, 15, 16, 17, 33, 34, 35, 36, 37, 47, 48, 49}, {subreddit-other=0})
3: ({33, 34, 40, 41, 42, 43, 44, 45, 47, 48, 50}, {subreddit-worldnews=0})
4: ({33, 34, 47, 48}, {subreddit-worldnews=0, subreddit-other=0})
5: ({49, 50}, {subreddit-todayilearned=0})
6: ({40, 41, 42, 43}, {subreddit-news=0, subreddit-worldnews=0})
```

Una vez hecho podemos calcular cierres, primero lo realizamos con el *extent* que se trata del conjunto con los objetos.

```
> S <- Set$new(attributes = fc$objects)
> S$assign(attributes = "1", values = 1); S
{1}

> fc$intent(S)
```

```
{domain-subreddit.unpopularopinion=0}
```

Después realizamos lo mismo pero esta vez con el *intent* que se trata del conjunto con los atributos:

```
> S <- Set$new(attributes = fc$attributes)
> S$assign(attributes = "subreddit-worldnews=0", values = 1); S
{subreddit-worldnews=0}

> fc$extent(S)
{33, 34, 40, 41, 42, 43, 44, 45, 47, 48, 50}
```

Es posible seleccionar un sub-grafo en la cual se cumple que el soporte sea mayor que un valor, en este caso 0.2.

```

> fc$concepts$support()
 [1] 1.00 0.24 0.22 0.08 0.04 0.08 0.04 0.10 0.02 0.02 0.02 0.02 0.02 0.02
    0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.08
    0.02 0.02 0.06
[32] 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.08 0.10 0.04 0.04 0.02 0.12
    0.02 0.02 0.02 0.02 0.02 0.12 0.02 0.02 0.02 0.02 0.04 0.10 0.02 0.08
    0.02 0.06 0.02
[63] 0.06 0.02 0.04 0.02 0.00

> idx <- which(fc$concepts$support() > 0.2)
> sublattice <- fc$concepts$sublattice(idx); sublattice
A set of 4 concepts:
1: ({1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
     21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
     39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}, {})
2: ({14, 15, 16, 17, 33, 34, 35, 36, 37, 47, 48, 49}, {subreddit-other=0})
3: ({33, 34, 40, 41, 42, 43, 44, 45, 47, 48, 50}, {subreddit-worldnews=0})
4: ({33, 34, 47, 48}, {subreddit-worldnews=0, subreddit-other=0})
```

También en base a un concepto en concreto podemos obtener más información correspondiente al grafo como pueden ser los infimos, supremos, superconceptos, subconceptos.

```
> concept <- fc$concepts[31]; concept
A set of 1 concepts:
1: ({40, 44, 45}, {domain-france24.com=0, subreddit-worldnews=0})

> fc$concepts$superconcepts(concept)
A set of 3 concepts:
1: ({1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
     21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
     39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}, {})
2: ({33, 34, 40, 41, 42, 43, 44, 45, 47, 48, 50}, {subreddit-worldnews=0})
3: ({40, 44, 45}, {domain-france24.com=0, subreddit-worldnews=0})

> fc$concepts$subconcepts(concept)
A set of 5 concepts:
1: ({40, 44, 45}, {domain-france24.com=0, subreddit-worldnews=0})
2: ({40}, {domain-france24.com=0, subreddit-news=0, subreddit-worldnews=0})
3: ({44}, {upvote_ratio-High=1, domain-france24.com=0, subreddit-worldnews
   =0})
4: ({45}, {upvote_ratio-Medium=0, domain-france24.com=0, subreddit-
   worldnews=0})
5: ({}, {upvote_ratio-Low=0, upvote_ratio-Low=1, upvote_ratio-Medium=0,
   upvote_ratio-Medium=1, ....})

> concepts <- fc$concepts[1:3]

> fc$concepts$supremum(concepts)
({1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
   21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
   39, 40, 41, 42,
  43, 44, 45, 46, 47, 48, 49, 50}, {})

```

```
> fc$concepts$infimum(concepts)
({33, 34, 47, 48}, {subreddit-worldnews=0, subreddit-other=0})

> head(fc$concepts$join_irreducibles())
A set of 6 concepts:
1: ({21}, {domain-other=0, subreddit-lifeprotips=0})
2: ({18}, {domain-other=0, subreddit-askreddit=0})
3: ({19}, {domain-other=0, subreddit-abrathatfits=0})
4: ({20}, {domain-other=0, subreddit-3amjokes=0})
5: ({36}, {domain-youtu.be=0, subreddit-ableton=0, subreddit-other=0})
6: ({22}, {domain-v.redd.it=0, subreddit-publicfreakout=0})
```

Es posible reducir el contexto formal, permitiendo que se puedan obtener menos conceptos o implicaciones.

Antes de reducir el contexto formal tenemos los siguientes tamaños:

```
> fc$find_concepts()

> fc$concepts$size()
[1] 67

> fc$find_implications()

> fc$implications$cardinality()
[1] 527
```

Después de haber reducido el contexto formal se observa que se obtienen menos implicaciones

```
> fc$reduce()

> fc$find_concepts()

> fc$concepts$size()
```

```
[1] 67

> fc$find_implications()

> fc$implications$cardinality()
[1] 300
```

Haciendo uso del algoritmo *NextClosure* [27] es posible obtener implicaciones tal como se utiliza para computar los conceptos:

```
> head(fc$implications)
Implication set with 6 implications.
Rule 1: {[domain-youtu.be=0, subreddit-ableton=0]} -> {subreddit-other=0}
Rule 2: {[domain-theguardian.com=0, subreddit-science=0]} -> {domain-other
    =0}
Rule 3: {[domain-bloomberg.com=0, subreddit-technology=0]} -> {subreddit-
    worldnews=0, subreddit-other=0}
Rule 4: {subreddit-other=0, [domain-v.redd.it=0, subreddit-publicfreakout
    =0]} -> subreddit-lifeprotips=0, subreddit-news=0, subreddit-
    todayilearned=0, subreddit-worldnews=0, [domain-bloomberg.com=0,
    subreddit-technology=0], [domain-theguardian.com=0, subreddit-science
    =0], [domain-youtu.be=0, subreddit-ableton=0], ...}
Rule 5: {subreddit-worldnews=0, [domain-v.redd.it=0, subreddit-
    publicfreakout=0]} -> {upvote_ratio-Low=0, upvote_ratio-Medium=0, upvote
    _ratio-High=1, total_awards_received-Low=1, total_awards_received-Medium
    =0, total_awards_received-High=0, score-Low=1, score-Medium=0, score-
    High=0, domain-abc.net.au=0,
  domain-aljazeera.com=0, domain-cbc.ca=0, ...}
Rule 6: {subreddit-worldnews=0, subreddit-other=0, [domain-youtu.be=0,
    subreddit-ableton=0]} -> subreddit-
  leopardsatemyface=0, subreddit-lifeprotips=0, subreddit-news=0, subreddit
    -todayilearned=0, [domain-bloomberg.com=0, subreddit-technology=0], [
    domain-theguardian.com=0, subreddit-science=0], [domain-v.redd.it=0,
    subreddit-publicfreakout=0], ...}
```

También es posible eliminar redundancia aplicando técnicas de simplificación lógica:

```
> fc$implications$apply_rules(c("reduction", "composition", "generalization
    ", "simplification"))
Processing batch

—> Reduction: from 300 to 300 in 0 secs.
—> Composition: from 300 to 300 in 0 secs.
—> Generalization: from 300 to 300 in 0 secs.
—> Simplification: from 300 to 300 in 0.14 secs.
Batch took 0.14 secs.
```

Por último, es posible exportar las implicaciones computadas de un contexto formal a reglas de asociación:

```
> rules <- fc$implications$to_arules(); rules
set of 300 rules
```

Con esto podemos ordenar por los distintos parámetros disponibles bajo el paquete *arules*, por ejemplo en este caso ordenamos por los parámetros *lift* y *support*:

```
> lift <- arules::sort(rules, by="lift")
> support <- arules::sort(rules, by="support")

> lift.df <- data.frame(inspect(head(lift, 10)))[c('lhs', 'rhs', 'support',
    'confidence', 'lift')]; lift.df
```

	lhs	rhs	support	confidence	lift
1	{domain-twitter.com=0}	{subreddit-leopardsatemyface=0,subreddit-other=0}	0.02040816	1	24.50000
2	{upvote_ratio-High=1,subreddit-worldnews=0}	{domain-france24.com=0}	0.02040816	1	16.33333
3	{upvote_ratio-Medium=0,subreddit-worldnews=0}	{domain-france24.com=0	0.02040816	1	16.33333
4	{[domain-bloomberg.com=0, subreddit-technology=0]}	{subreddit-worldnews=0,subreddit-other=0	0.02040816	1	12.25000
5	{subreddit-cursedcomments=0}	{domain-i.redd.it=0}	0.02040816	1	12.25000
6	{subreddit-antiwork=0}	{domain-i.redd.it=0}	0.02040816	1	12.25000
7	{domain-euronews.com=0}	{subreddit-worldnews=0,subreddit-other=0}	0.02040816	1	12.25000
8	{domain-dailymail.co.uk=0}	{subreddit-worldnews=0,subreddit-other=0}	0.02040816	1	12.25000
9	{domain-cbc.ca=0}	{subreddit-news=0,subreddit-worldnews=0}	0.02040816	1	12.25000
10	{domain-aljazeera.com=0}	{subreddit-news=0,subreddit-worldnews=0}	0.02040816	1	12.25000

```
> support.df <- data.frame(inspect(head(support, 10)))[c('lhs', 'rhs', '
    support', 'confidence', 'lift')]; support.df
```

	lhs	rhs	support	confidence	lift
1	{subreddit-news=0}	{subreddit-worldnews=0}	0.02040816	1	4.454545
2	{domain-france24.com=0}	{subreddit-worldnews=0}	0.02040816	1	4.454545
3	{subreddit-leopardsatemyface=0}	{subreddit-other=0}	0.02040816	1	4.083333
4	{score-High=0}	{score-Low=1}	0.02040816	1	9.800000
5	{[domain-youtu.be=0, subreddit-ableton=0]}	{subreddit-other=0}	0.02040816	1	4.083333
6	{[domain-theguardian.com=0, subreddit-science=0]}	{domain-other=0}	0.02040816	1	9.800000
7	{[domain-bloomberg.com=0, subreddit-technology=0]}	{subreddit-worldnews=0,subreddit-other=0}	0.02040816	1	12.25000
8	{subreddit-lifeprotips=0}	{domain-other=0}	0.02040816	1	9.800000
9	{subreddit-cursedcomments=0}	{domain-i.redd.it=0}	0.02040816	1	12.25000
10	{subreddit-askreddit=0}	{domain-other=0}	0.02040816	1	9.800000

6

Implementación de la aplicación

6.1. Extracción de datos

En este apartado se comenta sobre la implementación del apartado de la extracción de datos en la aplicación.

Esta implementación se encuentra distribuida en varios ficheros, las cuales se mencionan a continuación:

- **award.R**. Fichero que engloba todos los métodos para el tratamiento de las recompensas y monedas que se encuentran asociadas a las publicaciones y comentarios de los usuarios.

- **comment.R**. Fichero que engloba el tratamiento de los comentarios de una publicación.

- **post.R**. Fichero principal donde se realiza todo el proceso de extracción e inserción de los datos.

- **subreddit.R**. Fichero en la cual viene implementado el tratamiento de los datos referentes a los subreddits.

- **user.R**. Fichero en la cual se trata los datos referentes a los autores de las publicaciones y los usuarios asociados a los comentarios de cada publicación.

6.1.1. Subreddits

Si se parte de una aplicación en la cual no hay información presente en la base de datos, se puede importar una lista predefinida de subreddits que se encuentra en la carpeta *data*.

Así mismo, en este fichero existen dos métodos para la obtención de un número de publicaciones de todos los subreddits existentes en la base de datos o de una lista de subreddits más reducida.

NOTA: En ambos casos la obtención de los datos esta limitado a la conexión de internet del usuario y del límite de conexiones que establece Reddit, por lo tanto en función del número de subreddits que se especifique o existan en la base de datos puede tardar de 5 minutos a 1 hora.

6.1.2. Publicaciones

Se parte de la base que se necesita una URL de una publicación, una vez obtenido una URL se realiza una petición GET para obtener la información de esta publicación en el formato JSON.

Después, se empieza a normalizar la información obtenida del conjunto de datos en la cual se realizan las siguientes acciones:

- Si la publicación no tiene una descripción se añade una descripción vacía.
- Si el usuario no existe, el usuario ha sido eliminado, suspendido o dado de baja por cualquier motivo, se asigna un usuario por defecto.
- Se quita todos los caracteres no alfa-numéricos del título y de la descripción.
- De la URL se quitan las comillas simples y se convierte las barras dobles por una única.
- Se obtiene el identificador único del autor de la publicación para que en un paso posterior se obtenga la información relacionada con este usuario. Si este identificador no se encuentra se asigna un identificador genérico.

- En base al subreddit se extrae el identificador único al cual está asignado en la base de datos

- Se obtiene aquellas publicaciones que no se han insertado en la base de datos y se ejecuta una función para que nos indique si por cada publicación este se ha insertado en la base de datos, en base a si se han insertado todos o falta alguna publicación por insertar se realiza la *Acción 1 - Insertar todos* o *Acción 2 - Insertar las publicaciones no existentes en el servidor de base de datos.*

    - *Acción 1 - Insertar todos*

        - Por cada publicación del conjunto de datos se obtiene la información correspondiente al autor y se inserta en la base de datos. En un punto posterior entraremos más en detalle sobre el tratamiento de los usuarios.
        - Se inserta todas las publicaciones en la base de datos haciendo uso de la inserción múltiple que ofrece SQL.
        - Se obtiene las recompensas y monedas asociadas por cada publicación.
        - Si la opción de insertar los comentarios se encuentra activo (por defecto, esta opción se encuentra habilitado) se obtienen y se insertan en la base de datos.

    - *Acción 2 - Insertar las publicaciones no existentes en el servidor de base de datos*

        - Del conjunto de datos obtenemos aquellas publicaciones que no han sido insertadas.
        - Se realiza el mismo procedimiento que en el apartado *Acción 1.*

Una vez finalizado todo el proceso, se devuelve el conjunto de datos ya tratado para que sea manipulado e impreso por la aplicación web.

6.1.3. Comentarios

En este sub-apartado comentamos el proceso para el tratamiento de los comentarios de una publicación, en la cual recibe un conjunto de datos y por cada publicación presente

en este conjunto se obtienen los comentarios.

Este proceso es el siguiente:

- Por cada publicación se realiza lo siguiente:
    - Se obtienen los comentarios de una publicación en la cual realiza una petición GET para obtener la información en el formato JSON (está predefinido para obtener los 50 comentarios con más puntos, al establecer un número de comentarios más alto implica más lentitud a la hora de obtener y procesar los comentarios), una vez se obtiene el conjunto inicial se realiza un tratamiento y manipulación para conservar los datos de más interés referente a cada comentario realizado.
    - Se eliminan aquellas columnas del conjunto de datos que no son necesarias para la aplicación.
    - Se eliminan aquellos comentarios realizados por el usuario *AutoModerator* o un usuario eliminado, ya que en el primer caso los comentarios son predefinidos y no son de interés, y en el segundo caso el comentario ha sido eliminado por algún motivo y tampoco es de interés conservar este dato.
    - De los comentarios restantes asociados a la publicación se realiza una eliminación de todo carácter no alfa-numérico en el campo referente a lo escrito por el usuario.
    - Se normaliza la fecha del comentario a un formato estándar y se obtiene el identificador único de la publicación.
    - De aquellos comentarios se comprueba si la publicación dispone de una descripción y se actualiza con la descripción de la publicación que se encuentra localizado en el conjunto de datos. Esto se debe a que se puede dar el caso en el que se crea la publicación sin descripción y posteriormente cuando se obtiene los comentarios asociados a esta publicación, la descripción de la publicación ha sido actualizado por el autor del mismo.

- Por cada usuario presente en el conjunto de datos se obtiene la información referente a el mismo y se inserta en el servidor de base de datos.
- Por último, se inserta en el servidor de base de datos estos comentarios asociados a una publicación.

6.1.4. Recompensas

En este sub-apartado comentamos el proceso para el tratamiento de las recompensas asociadas a una publicación, se recibe un conjunto de datos con todas las publicaciones.

El proceso para este tratamiento es el siguiente:

- Se comprueba si en todas las publicaciones hay al menos 1 o más recompensas asociadas, en caso afirmativo se procede a realizar lo siguiente por cada publicación:
  - Referente a la moneda se obtiene los identificadores de la misma, el nombre, la descripción, el precio y la recompensa asociada.
  - Referente a las recompensas asociadas a cada publicación se obtiene los identificadores de la publicación, de la moneda y la cantidad.
  - Una vez obtenidos estos datos se guardan en un conjunto diferente y procede con su inserción en el servidor de base de datos aquellas monedas y recompensas asociadas a la publicación que no se encuentran presentes.

6.1.5. Usuarios

En este sub-apartado comentamos el proceso para el tratamiento de los usuarios asociadas a una publicación la cual puede haber sido como autor o escritor de un comentario asociado a una publicación.

El proceso para el tratamiento de los usuarios es el siguiente:

- Se comprueba si el usuario existe, una vez hecho esto se comprueba si se corresponde con los usuarios *AutoModerator* o un usuario eliminado, en ambos casos se devuelve un conjunto con unos datos predefinidos.

- Una vez pasado estas comprobaciones, se realiza una petición GET para obtener la información, se comprueba si se obtiene un resultado, en caso contrario significa que el usuario ha sido eliminado del sistema.

- Se comprueba si el usuario ha sido suspendido, en caso afirmativo, se devuelve unos datos predefinidos.

- Si el usuario tiene establecido una descripción en su perfil de usuario se elimina todos carácter no alfa-numérico.

- Llegado a este punto, el usuario es un usuario válido y se encuentra activo, por tanto se procesa la información relacionada con el usuario y se inserta en el servidor de base de datos.

6.2. Aplicación Web Shiny

En este apartado se comenta sobre la implementación de la aplicación web realizada en shiny.

La aplicación se estructura para ofrecer dos finalidades:

- La extracción de datos mediante la aplicación y visualización de los datos obtenidos

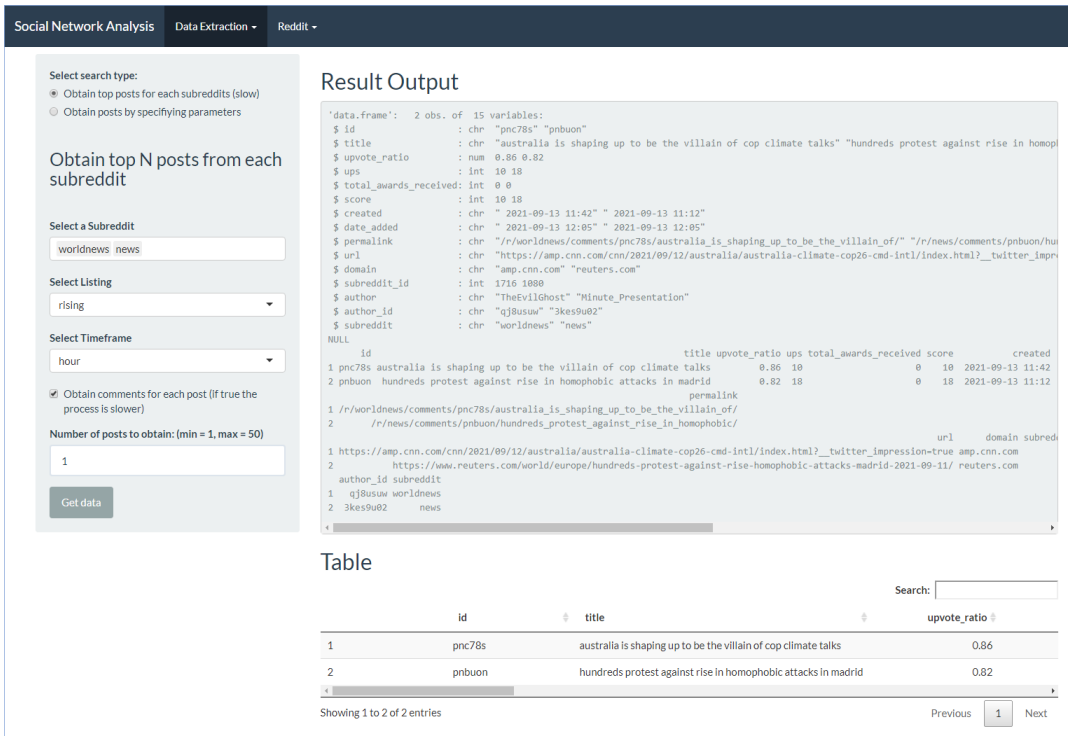

Figura 6.1: Aplicación Shiny - Extracción datos

- La obtención de los datos del servidor de base de datos, visualizar estos datos y la aplicación de diversas técnicas para extraer información de estos datos.

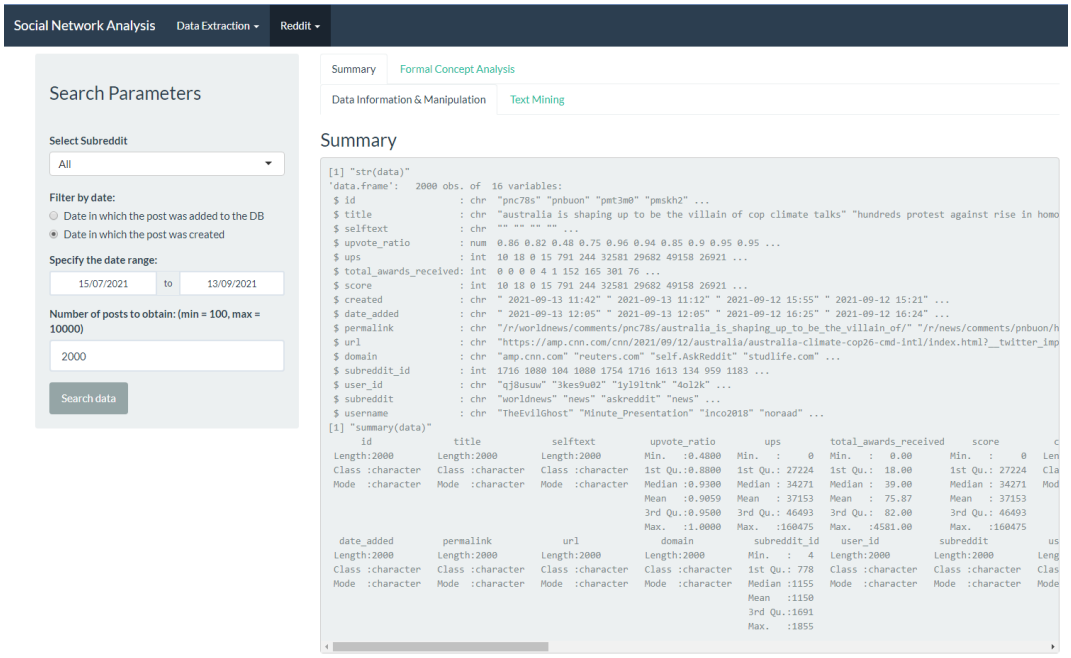

Figura 6.2: Aplicación Shiny - Análisis de los datos

Para tener una aplicación web más modular y en la cual se intenta no repetir código, se han realizado las siguientes acciones:

- Separar la lógica del apartado UI y del apartado del servidor en dos ficheros diferentes, esto se realiza dado que contienen muchas líneas de código para una separación sobre la funcionalidad que realiza la misma.
- La implementación de módulos shiny para los apartados UI y de servidor, permitiendo reutilizar implementaciones ya existentes y tener una modularización de la aplicación más eficiente.

Esta implementación se encuentra distribuida en varios ficheros y carpetas, las cuales se mencionan a continuación:

- **src/database/standard.R**. Fichero que implementa una entidad que conecta con el servidor de base de datos.
- **src/modules/server**. Carpeta que contiene los módulos diseñados para el apartado de servidor Shiny
  - **data.R**. Se encarga de ejecutar la lógica del apartado *Reddit* en cualquiera de sus opciones.
  - **etl.R**. Se encarga de ejecutar la lógica del apartado *Data Extraction*.
  - **fca.R**. Se encarga de ejecutar todo el proceso de análisis del sub-apartado de *Formal Concept Analysis*.
  - **tm.R**. Se encarga de ejecutar la lógica del sub-apartado de *Text Mining*.
- **src/modules/ui**. Carpeta que contiene los módulos para el apartado gráfico del servidor Shiny.
  - **data.R**. Fichero que contiene la configuración de los elementos que se muestran en el apartado *Reddit* en cualquiera de sus opciones.
  - **etl.R**. Fichero que contiene la configuración de los elementos que se muestran en el apartado *Data Extraction*.

    - **fca.R**. Fichero que contiene la configuración de los elementos que se muestran en el sub-apartado de *Formal Concept Analysis*.
    - **main.R**. Fichero en el cual se configura la plantilla general para toda la aplicación.
    - **sidebar.R**. Fichero que contiene la configuración de los elementos de la barra lateral la cual se muestra de forma distinta según en el apartado.

- **src/constants.R**. Fichero que contiene una serie de constantes necesarias para el funcionamiento de la aplicación.
- **src/functions.R**. Fichero que contiene una serie de funciones generales que son usadas en múltiples apartados de la aplicación.
- **src/includes.R**. Fichero que contiene la inclusión y compilación de todos los ficheros necesarios.
- **src/load.R**. Fichero que contiene la instalación y carga de todos los paquetes de R que son necesarios.
- **app.R**. Fichero que lanza la aplicación web.
- **globals.R**. Fichero que precarga una serie de datos y que es común a toda la aplicación web.
- **server.R**. Fichero que carga toda la lógica de la parte del servidor de la aplicación web.
- **ui.R**. Fichero que carga la parte gráfica de la aplicación web.

A continuación, se comenta los diferentes módulos usados en los apartados de la aplicación web.

En este apartado se hace uso de los módulos de UI: *main, sidebar y etl*; Del lado del servidor se hace uso de los módulos: *etl*.
El fin de esta pantalla es disponer de la posibilidad de obtener información mediante la API y visualizar los datos que se obtienen para una previsualización rápida.

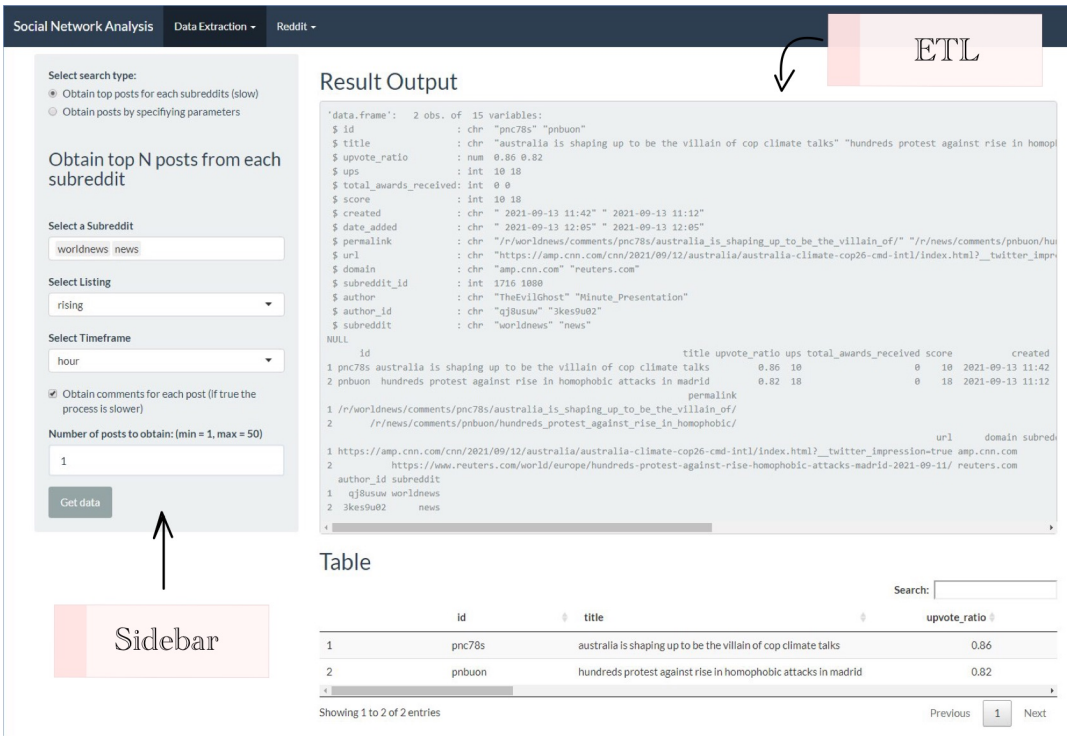

Figura 6.3: Aplicación Shiny - Módulos en Extracción datos

En este apartado se hace uso de los módulos de UI: *main, sidebar y data*; Del lado del servidor se hace uso de los módulos: *data*.

El fin de esta pantalla es disponer de la posibilidad de obtener la información que se encuentra presente en el servidor de base de datos y realizar un tratamiento de los mismos para un estudio posterior del conjunto de datos obtenido.

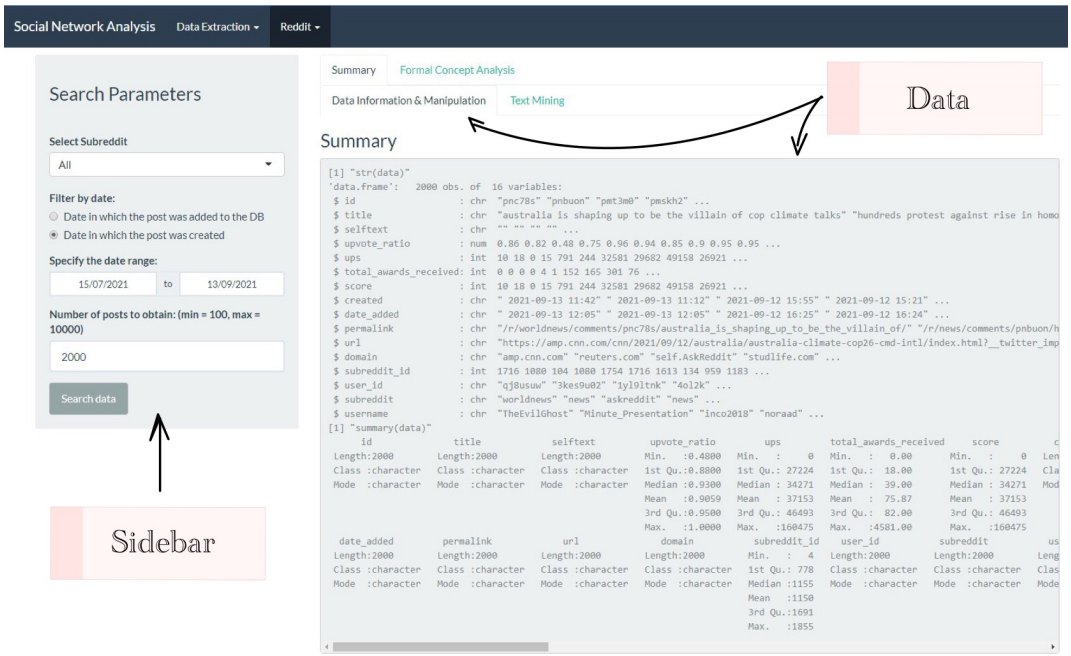

Figura 6.4: Aplicación Shiny - Módulos en Visualización datos

En este apartado se hace uso de los módulos de UI: *main, sidebar y tm*; Del lado del servidor se hace uso de los módulos: *tm*.

El fin de esta pantalla es mostrar que información relevante se puede obtener de las publicaciones referente al texto presente en el título o descripción.

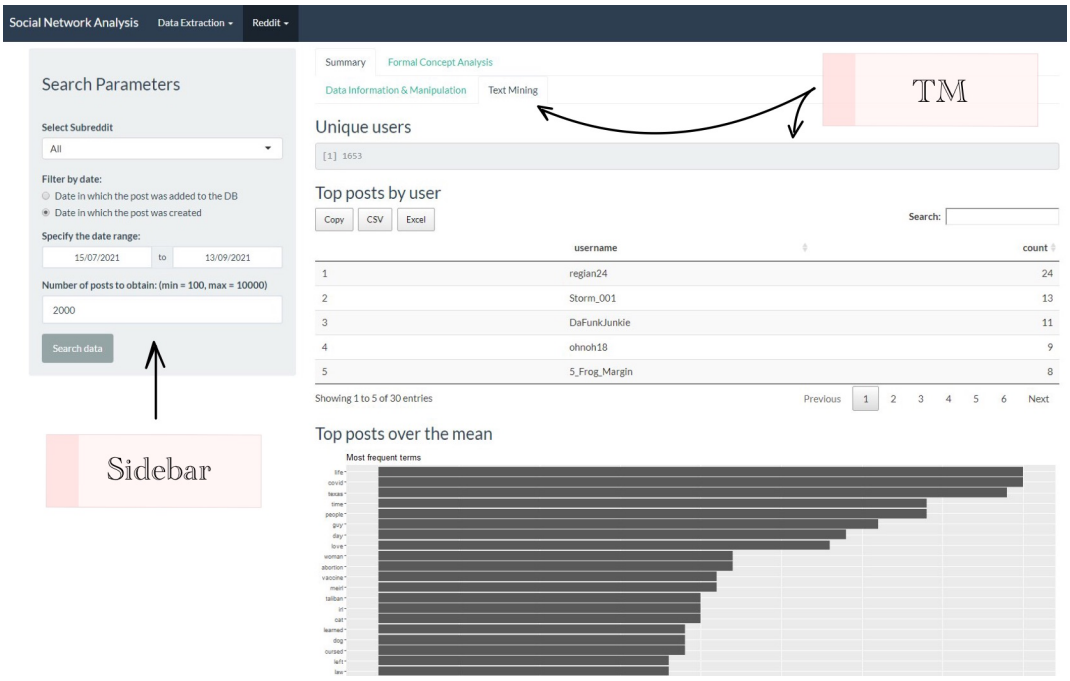

Figura 6.5: Aplicación Shiny - Módulos en Minería de datos

En este apartado se hace uso de los módulos de UI: *main, sidebar y fca*; Del lado del servidor se hace uso de los módulos: *fca*.

El fin de esta pantalla es poder realizar un análisis del conjunto de datos en el cual se aplican diversas operaciones para poder ir extrayendo la información mediante el uso de *FCA*.

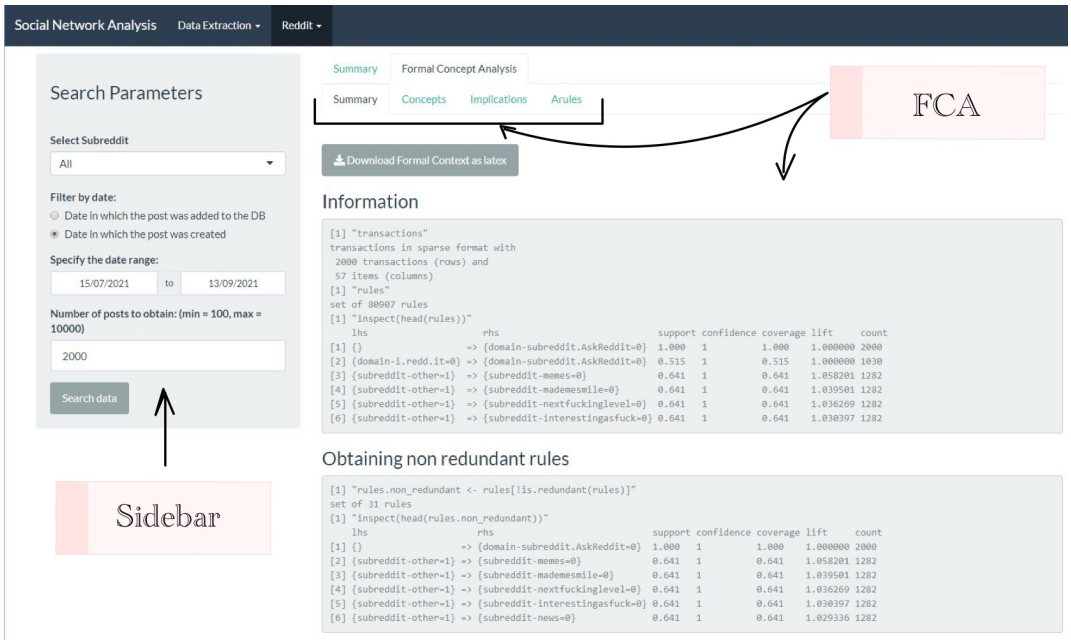

Figura 6.6: Aplicación Shiny - Módulos en Análisis de Conceptos Formales

7

Conclusiones y Líneas Futuras

7.1. Dificultades encontradas durante el proyecto

Durante la realización de este proyecto se han encontrado algunas dificultades que han hecho que se retrase el desarrollo del proyecto en sí.

La escasez de información referente a la configuración e instalación de un servidor Shiny en un contenedor Docker donde se requiere que esté aplicación se conecte a un servidor de base de datos externo, la cual a la hora de compilar la imagen personalizada la aplicación no era capaz de conectarse debido a la ausencia de librerías del sistema operativo. La solución a esto se trata de instalar y compilar varias librerías referentes a servidores de bases de datos en el mismo contenedor para que cuando el paquete de *R* realice la llamada esta sea comprendida por el sistema operativo.

Uno de los principales dificultades durante este proyecto se ha encontrado en el apartado de análisis de los datos, en el cual se descargaba un conjunto de datos de un tamaño considerable con más de 20,000 registros y a la hora de aplicar el análisis diseñado este hacía que se colgará la aplicación debido al gran consumo de memoria, ya que un conjunto de dicho tamaño puede generar un contexto formal con más de 500,000 reglas y para esto necesita una gran cantidad de memoria. Para solucionar este problema se ha tratado de reducir los datos con los que se trabaja pero esto no tuvo éxito ya que se seguía obteniendo un conjunto de datos relativamente grande. La solución encontrada, fue en base al conjunto de datos, aplicarle el algoritmo Apriori con unos parámetros para poder

filtrar y reducir las reglas que se obtuvieran, con esto obtener las reglas no redundantes y crear el contexto formal en base a estas reglas. Consiguiendo que se reduzca el tiempo de ejecución necesaria para poder realizar el análisis.

Otro inconveniente encontrada es relativa al entorno de desarrollo RStudio, en la cual cada vez que se lanza la aplicación y posteriormente esta se finaliza, este entorno no realiza una limpieza de los objetos en memoria permitiendo que se vaya acumulando el consumo de memoria de la aplicación hasta llegar al limite de memoria del equipo en cuestión. La solución encontrada para este problema es limpiar los objetos de la zona de trabajo y reiniciar la sesión de *R*. Con esto se reinicializa el proceso de la sesión *R* y permitiendo liberar memoria del equipo de desarrollo.

7.2. Conclusiones

Este presente trabajo fin de grado tenía como objetivos de este proyecto la realización de una aplicación web para la obtención de datos mediante la API de Reddit, la obtención de datos para un análisis de este conjunto utilizando técnicas ya conocidas y otra menos conocida como es FCA con la finalidad de ofrecer una ayuda más en el análisis de un conjunto de datos para poder extraer información oculta que se encuentra presente en un conjunto de datos.

Remarcar que durante este proyecto ha habido un alto aprendizaje del lenguaje de programación *R* bajo el cual toda la aplicación está implementado, aunque ya se disponía de ciertos conocimientos con este lenguaje, se ha aprendido a trabajar con micro-servicios de tipo rest bajo R y la implementación de aplicaciones modulares de Shiny. Otro punto a destacar, es que haciendo uso de esta aplicación se podrá usar para analizar e investigar otros problemas relevantes relacionados con las redes sociales como pueden ser la segmentación de usuarios, la identificación de posibles clientes, análisis de mercados, estudios sociológicos, etc... Así mismo, este proyecto tiene principal finalidad que pueda servir de ayuda a cualquier investigador o persona que esté interesada en el área del lenguaje de programación *R* y FCA.

Por último, destacar que el desarrollo de este proyecto ha sido satisfactoria, en el cual se ha desarrollado todo lo propuesto y se han cumplido los plazos de entrega de cada apartado de este proyecto. Habiendo realizado una distribución de las tareas a realizar y los objetivos necesarios para cada apartado ha facilitado el desarrollo del proyecto y memoria.

7.3. Líneas Futuras

La principal mejora que se puede incluir en este proyecto es añadir más redes sociales a la aplicación para el estudio y análisis de los mismos, como pueden ser Facebook, LinkedIn, Discord u otros. Añadir estas redes incrementa la complejidad de la aplicación ya que por cada red social es necesario mirar si disponen de una API pública, si tienen establecido algún limite para realizar peticiones y qué datos se pueden obtener. De estos datos hay que estudiar la estructura que tiene y si es posible crear relaciones entre ellos ya que si no existen no será posible aplicar las técnicas de FCA para el conjunto de datos.

Otra mejora es facilitar la importación de un conjunto de datos en el cual se quiere descubrir la información oculta que se encuentra presente en dicho conjunto, para esto en la aplicación se habilitaría una opción para realizar dicha importación. Dependiendo del tamaño del conjunto de datos la importación puede tardar más o menos, no obstante una vez importado se ejecutará el proceso de normalización de datos para que estos tengan valores consistentes y validos. Posteriormente, una vez normalizado se aplicarán las técnicas de extracción de información y FCA sobre el conjunto de datos, pudiendo ver los resultados que se obtienen.

Otro punto que sería interesante implementar, es la posibilidad de crear y gestionar tareas programadas para que ejecuten el proceso de extracción de datos con una serie de parámetros para que estos sean ejecutados de forma periódica con lo cual permitiría despreocuparnos de la inserción de nuevos datos en el servidor de base de datos.

Por último, otro punto interesante a incluir en este presente proyecto es la inclusión

de técnicas de inteligencia artificial las cuales se pueden combinar con FCA, consiguiendo extraer mucha más información la cual puede ser útil para la realización de estudios más densos de una red social o tema o la implementación de aplicaciones como detección de spam, clasificación de usuarios, análisis de tendencias, etc...

Referencias

[1] *Documentación R*. URL: https://cran.r-project.org/.

[2] *Docker*. URL: https://docs.docker.com/.

[3] *MariaDB*. URL: https://mariadb.org/documentation/.

[4] *phpMyAdmin*. URL: https://www.phpmyadmin.net/docs/.

[5] *mariadb - contenedor docker*. URL: https://hub.docker.com/_/mariadb.

[6] *phpmyadmin - contenedor docker*. URL: https://hub.docker.com/r/phpmyadmin/phpmyadmin/.

[7] *RStudio*. URL: https://www.rstudio.com/.

[8] *Lenguaje de Programación R*. URL: https://www.r-project.org/.

[9] *Overleaf*. URL: https://www.overleaf.com/.

[10] *Reddit*. URL: https://www.reddit.com/.

[11] *Twitter*. URL: https://twitter.com/.

[12] *Tidyverse*. URL: https://www.tidyverse.org/.

[13] *Reddit API*. URL: https://www.reddit.com/dev/api/.

[14] *Twitter API*. URL: https://developer.twitter.com/en/docs.

[15] *Shiny*. URL: https://shiny.rstudio.com/.

[16] *fcaR*. URL: https://cran.r-project.org/web/packages/fcaR/fcaR.pdf.

[17] *twitteR*. URL: https://cran.r-project.org/web/packages/twitteR/twitteR.pdf.

[18] *RedditExtractoR*. URL: https://cran.r-project.org/web/packages/RedditExtractoR/RedditExtractoR.pdf.

[19] *data.table*. URL: https://cran.r-project.org/web/packages/data.table/data.table.pdf.

[20] *RMariaDB*. URL: https://cran.r-project.org/web/packages/RMariaDB/RMariaDB.pdf.

[21] *Bootstrap*. URL: https://getbootstrap.com/docs/5.0/getting-started/introduction/.

[22] Pablo Cordero y col. «A conversational recommender system for diagnosis using fuzzy rules». En: *Expert Systems with Applications* 154 (2020). DOI: 10.1016/j.eswa.2020.113449.

[23] Bernhard Ganter, Gerd Stumme y Rudolf Wille. *Formal Concept Analysis*. Lecture Notes in Artificial Intelligence. Springer-Verlag Berlin Heidelberg, 2005. ISBN: 978-3-540-31881-1.

[24] Dmitry Ignatov. «Introduction to Formal Concept Analysis and Its Applications in Information Retrieval and Related Fields». En: *RuSSIR* 505 (2014). DOI: 10.1007/978-3-319-25485-2_3.

[25] Paula R. C. Silva y col. «Formal Concept Analysis Applied to Professional Social Networks Analysis». En: *Proceedings of the 19th International Conference on Enterprise Information Systems* 3 (2020). DOI: 10.5220/0006333401230134.

[26] Pablo Cordero y col. «Knowledge discovery in social networks by using a logic-based treatment of implications». En: *Knowledge-Based Systems* 87 (2015). DOI: 10.1016/j.knosys.2015.07.018.

[27] Radim Bělohlávek. «Concept lattices and order in fuzzy logic». En: *Ann. Pure Appl. Logic*. 2004. Cap. 128, págs. 277-298.

[28] Esteban Ortiz-Orpina. *The rise of social media*. URL: https://ourworldindata.org/rise-of-social-media.

[29] Stanislav Krajči. «Social Network Analysis and Formal Concept Analysis». En: *Social Networks: A Framework of Computational Intelligence*. Ed. por Witold Pedrycz y Shyi-Ming Chen. Springer International Publishing, 2014. Cap. 3, págs. 41-50.

[30] Julia Silge y David Robinson. *Welcome to Text Mining with R*. URL: https://www.tidytextmining.com/.

Apéndice A
Glosario

agile Enfoque iterativo a la gestión de proyectos y desarrollo de software. 10

api Interfaz de Programación de Aplicaciones. 15, 19, 36, 39, 41, 61, 68, 69, 85

apriori Algoritmo para la minería de datos para la búsqueda de un conjunto de elementos frecuentes. 44, 45, 67

c Lenguaje de programación de propósito general. 24

c++ Lenguaje de programación de propósito general basado en C. 24

cran Repositorio de paquetes R. 24

csv Fichero que contiene valores separados por comas. 90

diagrma hasse Representación gráfica simplificada de un conjunto parcialmente ordenado finito. 26

docker Plataforma de virtualización que permite crear, probar y desplegar aplicaciones rápidamente . 5, 21, 22, 67, 78, 82

dockerfile Fichero de configuración para una imagen modificada para un contenedor. 82

etc Proceso de Extracción, Transformación y Carga. 7, 10, 35

extent Conjunto con los atributos comunes. 46

fca Análisis de Conceptos Formales. 2, 10, 12, 16, 25, 27, 64, 68, 69, 70

get Método de petición HTTP en la cual se solicita una representación de un recurso específico. 36, 37, 38, 39, 40, 54, 56, 58

ggplot2 Paquete de R para la creación de gráficos atractivos. 24

github Proveedor que ofrece una aplicación online para el control de versiones de un proyecto. 77

http Protocolo usado para la transferencia de datos en Internet. 36

intent Conjunto de objetos que posee todos los atributos en el conjunto. 47

iso Organización Internacional de Normalización. 40

json Formato estándar para la representación de datos basado en la sintaxis de JavaScript. 39, 54, 56

lattice Paquete de R para la creación de gráficos de grafos. 24

lift Medida la cual indica el grado de importancia de una regla. 91

látex Sistema de composición de textos que ofrece una alta calidad tipográfica. 7, 24, 91

mariadb Servidor de base de datos basado en MySQL. 22, 23, 78, 79

mysql Servidor de base de datos. 22, 23

overleaf Herramienta online para la escritura de documentos en látex. 24

php Lenguaje de programación orientado al desarrollo web. 23

phpmyadmin Gestor de base de datos. 23, 78, 79, 81

powershell Consola del sistema operativo Windows. 78

python Lenguaje de programación interpretado. 24

r Lenguaje de programación orientado al análisis estadístico. 2, 3, 23, 24, 31, 40, 68

reddit Red social. 6, 36, 39, 41, 54, 60, 68, 85, 88

rest Interfaz de programación de aplicaciones que permite interactuar con micro-servicios que confirman el estilo REST. 68

rstudio Entorno de desarrollo para R. 24, 31, 68

scrum Metodología Ágil que facilita el desarrollo software. 7, 10, 11

shiny Paquete de R para crear páginas web interactivas. 7, 12, 58, 59, 60, 62, 63, 64, 65, 67, 68, 82, 83, 86, 87, 89, 90, 91

sql Lenguaje de Consulta Estructurado. 23, 55, 79

subreddit Foro en la cual se publican temas. 6, 37, 38, 39, 40, 53, 54, 55, 86, 87, 89

support Medida que indica que tan frecuente aparece un elemento en un conjunto de datos. 91

ui Interfaz de Usuario. 60, 61, 62, 63, 64

url Localizador de Recursos Uniforme. 54

Apéndice B
Manual de Instalación

En este apéndice se describe todo el proceso de instalación de la aplicación bajo cualquier ordenador y sistema operativo.

Antes de entrar en los detalles es necesario remarcar, que se precisa de un ordenador con unas características con una configuración mínima que se detallan a continuación para que se pueda ejecutar la aplicación sin problemas:

1. Procesador: Intel Core i5 o AMD Ryzen 5
2. Memoria RAM: 8 GB
3. Disco SSD

El siguiente paso es descargar el código fuente del repositorio ubicado en Github, abrimos la página del repositorio y aparecerá una pantalla similar a la siguiente:

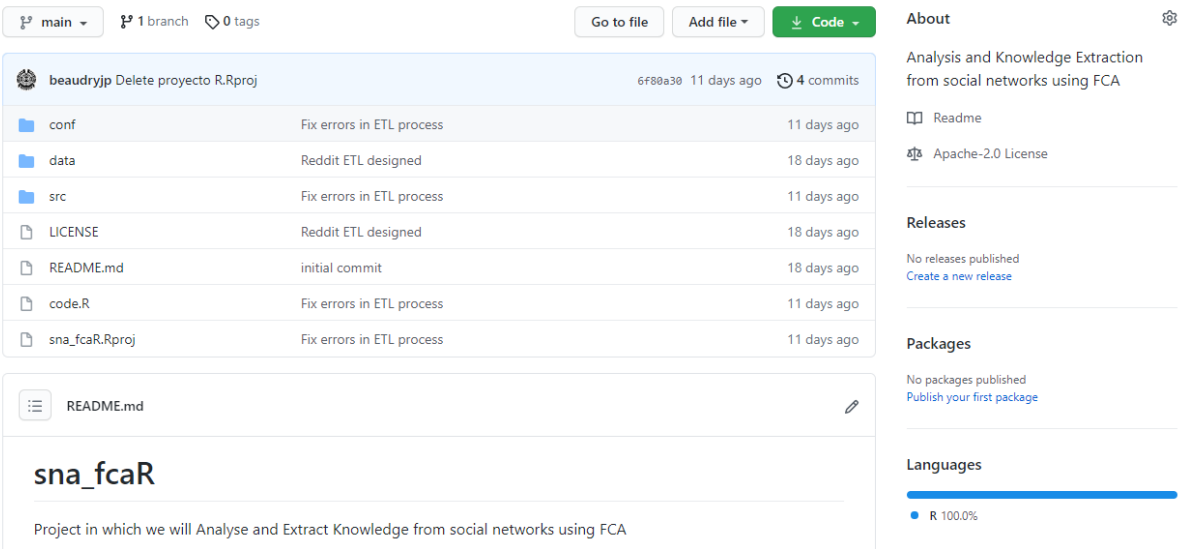

Figura B.1: Repositorio del código fuente de la aplicación

En dicha pantalla hay que pulsar sobre el botón *CODE* y posteriormente *Download ZIP*, esto descargará un fichero comprimido con todo el código fuente del proyecto el cual puede ser abierto por las aplicaciones nativas de descompresión de ficheros de cualquier sistema operativo.

Para el siguiente paso es necesario disponer de un servidor de base de datos ya existente o instalar uno nuevo, para este presente trabajo se utiliza la herramienta Docker[2], el cual gestionará dos contenedores, una maquina virtual con el servidor de base de datos MariaDB[3] y una maquina virtual con el gestor de base de datos phpMyAdmin[4] ya preconfigurada.

NOTA: Si la instalación se hace bajo el sistema operativo Windows, es necesario disponer de la versión 10 del mismo y tener instalado el Subsistema de Linux para Windows. En el siguiente enlace viene detallado los pasos a seguir para la instalación de dicho subsistema.

Se recomienda instalar Docker con las opciones que aparecen por defecto, una vez instalado la aplicación Docker, es necesario abrir una ventana de terminal (En Linux/-MacOS: Terminal, En Windows: PowerShell) y ejecutar los siguientes comandos en el siguiente orden:

- Para el contenedor MariaDB con los parámetros:
  - *-p*: El puerto interno y externo en el que corre el servicio MariaDB.
  - *-name*: El nombre del contenedor.
  - *-e*: Establecemos como variable de entorno la contraseña del usuario **root**, remarcar que es recomendable cambiar la contraseña a una más segura si se pone la aplicación en producción.
  - *-d*: Indicar que el contenedor funcione en segundo plano.

```
1 docker run -p 3306:3306  --name mariadb -e MYSQL_ROOT_PASSWORD=root123 -d
    mariadb/server
```

- Para el contenedor phpMyAdmin con los parámetros:

  * *-p*: El puerto interno y externo en el que corre el servicio phpMyAdmin.
  * *-name*: El nombre del contenedor.
  * *–link*: Enlazamiento de este contenedor con el contenedor del servidor de base de datos MariaDB.
  * *-d*: Indicar que el contenedor funcione en segundo plano.

```
1 docker run ––name my–own–phpmyadmin –d ––link mariadb:db –p 8081:80
    phpmyadmin/phpmyadmin
```

Una vez ejecutados estos dos comandos, tendremos dos contenedores con dos servicios diferentes en funcionamiento y en segundo plano. Para acceder al gestor de base datos hay que abrir un navegador y acceder a la siguiente dirección 127.0.0.1:8081, pinchar en la pestaña *SQL* y pegar el contenido copiado anteriormente.

En este momento es necesario crear la base de datos en la cual se alojará los datos relativos a la aplicación, este proceso se puede realizar mediante el gestor de base de datos *phpMyAdmin* en la pestaña *Databases*.

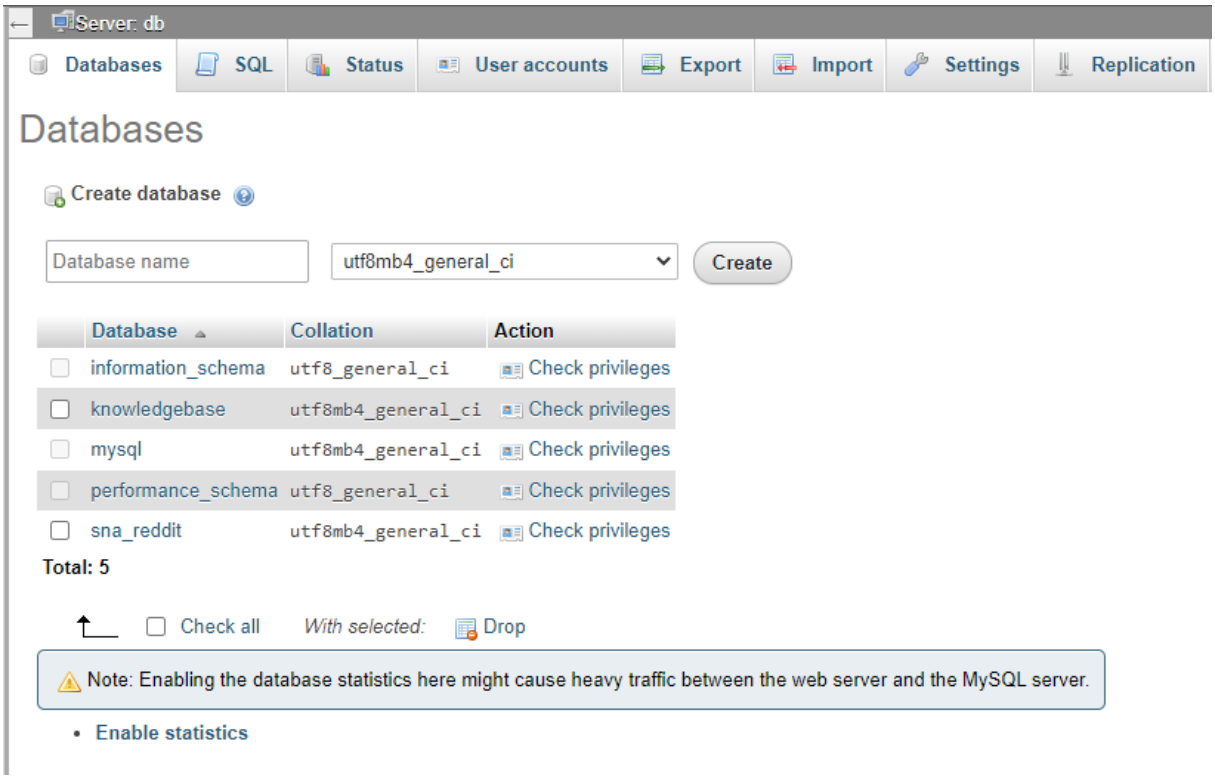

Figura B.2: Gestor de Base de datos - Creación nueva base de datos

No obstante también es posible realizarlo mediante un comando, el cual se especifica a continuación:

```
docker exec -it mariadb mysql -uroot -proot123 -e "create database sna_
    reddit"
```

Para este presente proyecto se ha creado un usuario para el servidor de base de datos llamado *david*, tal como se ha realizado en el punto anterior también es posible realizarlo por el gestor de base de datos *phpMyAdmin* en la pestaña *Privileges*.

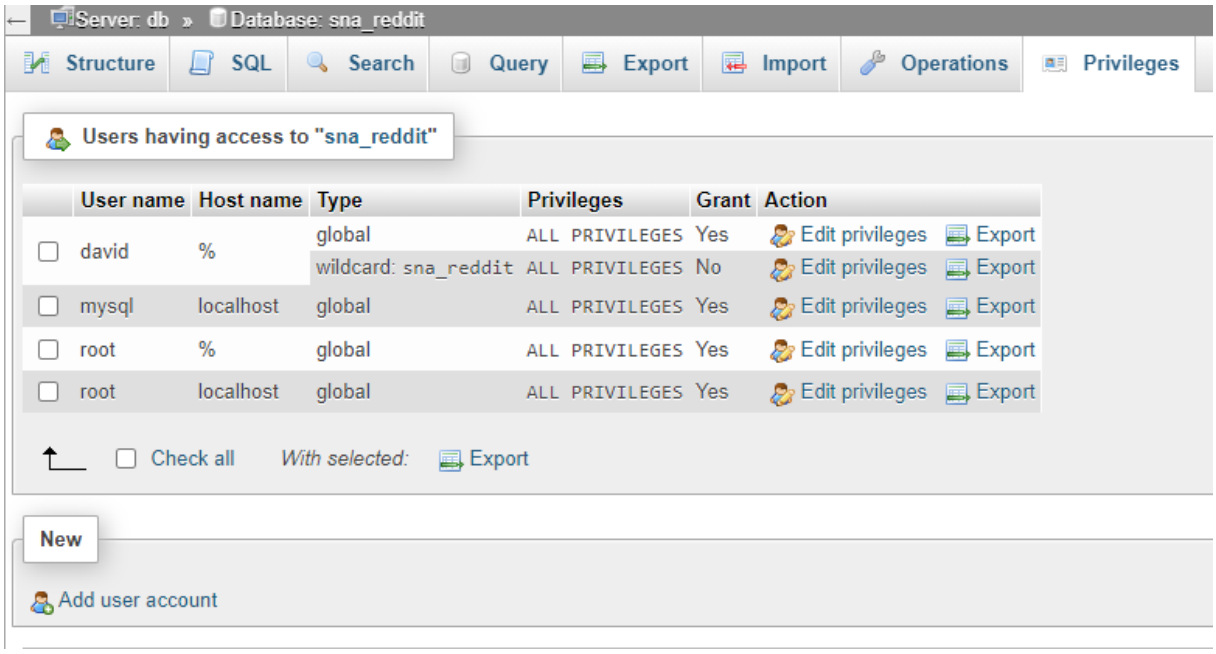

Figura B.3: Gestor de Base de datos - Creación nuevo usuario

Igualmente es posible realizarlo mediante un comando para la consola, el cual se especifica a continuación:

```
1 docker exec -it mariadb mysql -uroot -proot123 -e "CREATE USER 'david'@'%'
    IDENTIFIED BY 'p1YFHaVBpDmmurLU';"
2
3 docker exec -it mariadb mysql -uroot -proot123 -e "GRANT ALL PRIVILEGES on
    sna_reddit.* TO 'david'@'%' IDENTIFIED BY 'p1YFHaVBpDmmurLU'; FLUSH
    PRIVILEGES;"
```

Para importar la copia de seguridad de la base de datos en el servidor es recomendable realizarlo mediante línea de comandos debido al gran tamaño que tiene el fichero, no obstante esta operación se puede realizar mediante la aplicación web *phpMyAdmin*. Para realizarlo mediante la consola es necesario abrir una y ejecutar el siguiente comando:

```
1 docker exec -i mariadb mysql -uroot -proot123 sna_reddit < "sna_reddit.sql"
```

Por último, comentar que es posible encadenar la ejecución de todos estos comandos en uno solo para una mayor facilidad en la mantenimiento y administración de este servidor de base de datos:

```
docker run -p 3306:3306 --network r-db --name mariadb -e MYSQL_ROOT_PASSWORD=root123 -d mariadb/server && sleep 5 && \
docker exec -it mariadb mysql -uroot -proot123 -e "create database sna_reddit" && sleep 5 && \
docker exec -it mariadb mysql -uroot -proot123 -e "CREATE USER 'david'@'%' IDENTIFIED BY 'p1YFHaVBpDmmurLU'; " && sleep 5 &&
docker exec -it mariadb mysql -uroot -proot123 -e "GRANT ALL PRIVILEGES on sna_reddit.* TO 'david'@'%' IDENTIFIED BY 'p1YFHaVBpDmmurLU'; FLUSH PRIVILEGES;" && sleep 5 && \
docker exec -i mariadb mysql -uroot -proot123 sna_reddit < "sna_reddit.sql" && sleep 5 && \
docker run --name mariadb-phpmyadmin --network r-db -d --link mariadb:db -p 8081:80 phpmyadmin/phpmyadmin
```

Donde hay que especificar la contraseña root correcta en todas las operaciones que se realizan por la consola referente al servidor de base de datos.

Ahora creamos una imagen modificada de la aplicación web Shiny que incluye los paquetes necesarios para que la aplicación funcione, esta configuración viene definido en el fichero *Dockerfile* que se encuentra ubicado en la carpeta data.

Debemos crear una nueva carpeta en la cual tenemos la siguiente estructura:

- app/ (carpeta dónde estará ubicada el código fuente de la aplicación)
- Dockerfile (fichero con la configuración Docker)

Una vez se tiene esta estructura es necesario abrir una consola y ejecutar el siguiente comando:

```
docker build --tag custom-shiny .
```

Al finalizar la ejecución de este comando aparecerá si se ha ejecutado de forma correcta o no, en caso de que no ha habido problemas procedemos a lanzar el siguiente comando para crear un contendedor basado en esta nueva imagen creada:

```
docker run -d --name rshiny -p 3838:3838 custom-shiny
```

Con esto ya es posible acceder a la aplicación Shiny mediante un navegador web mediante el enlace 127.0.0.1:3838.

Con respecto al funcionamiento de los contenedores, es posible visualizar si están funcionando de forma correcta con el siguiente comando:

```
docker ps -a
```

Para ver el consumo actual de cada contenedor, es necesario ejecutar el siguiente comando:

```
docker stats
```

Apéndice C
Manual de Usuario

En este apéndice se describe en detalle todo el proceso para poder manejar la aplicación por cualquier usuario.

Esta aplicación ofrece dos funcionalidades:

1. Extraer información de la red social usando la API.

2. Trabajar con los datos guardados en la base de datos.

Estas dos opciones vienen separadas en el menú superior de la aplicación, la cual la primera opción se corresponde con *Data Extraction* y la segunda opción con *Reddit*.

C.1. Extracción de datos

Para este apartado pinchamos en el menú en *Data Extraction* →*Reddit*, una vez pinchado en este enlace en el apartado izquierdo aparecerá una barra lateral con campos a modificar como el siguiente:

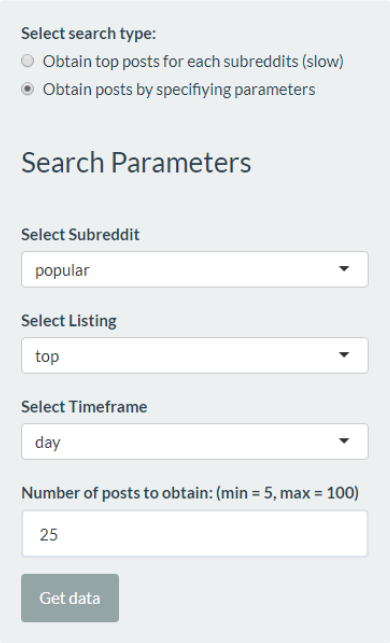

Figura C.1: Aplicación Shiny - Barra lateral de la Opción 1 del apartado Extracción de los datos

En esta pantalla viene seleccionado por defecto la opción *Obtain posts by specifying parameters*, esta opción nos permite obtener un número de publicaciones de un subreddit en concreto o de una etiqueta llamada popular donde engloba las publicaciones con mas controversia en este momento.

Con esta opción seleccionada es posible seleccionar los siguientes campos:

- **Select Subreddit**. Permite indicar el subreddit del cual obtener las publicaciones. Por defecto, viene marcada la opción *popular* pero es posible especificar un subreddit en concreto.

- **Select Listing**. Permite especificar en qué orden se buscan las publicaciones. Por defecto, viene marcada la opción *top* pero es posible especificar otro tipos de ordenes como *new* o *rising*.

- **Select Timeframe**. Permite especificar el plazo de tiempo en el cual se deben obtener las publicaciones. Por defecto, viene marcada la opción *día* pero es posible seleccionar otros plazos como pueden ser la *hora*, *semana*, *mes*, *año* o *todos*.

- **Number of posts to obtain**. Permite especificar el número de publicaciones a obtener. Por defecto, viene establecido 25 publicaciones el cual es un número razonable ya que suele tardar unos 30 segundos en ejecutarse, cuanto más alto sea este valor más tarda la operación en finalizar.

Si se selecciona la otra opción disponible *Obtain top posts for each subreddit* se obtiene una barra lateral con unas opciones similares a la opción anterior

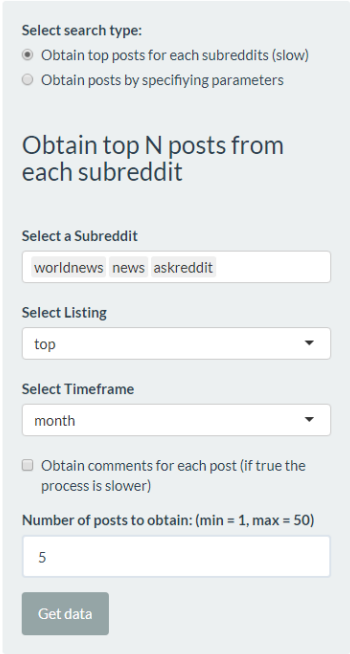

Figura C.2: Aplicación Shiny - Barra lateral de la Opción 2 del apartado Extracción de los datos

Con esta opción es posible establecer los siguientes campos:

- **Select Subreddit**. Permite indicar el subreddit o subreddits del cual obtener las publicaciones, también existe la opción para seleccionar todos los subreddits. Por defecto, viene marcada la opción *worldnews*, *news* y *askreddit* pero es posible añadir o quitar elementos de esta lista.

- **Select Listing**. Permite especificar en qué orden se buscan las publicaciones. Por defecto, viene marcada la opción *top* pero es posible especificar otro tipos de ordenes como *new* o *rising*.

- **Select Timeframe**. Permite especificar el plazo de tiempo en el cual se deben obtener las publicaciones. Por defecto, viene marcada la opción *día* pero es posible seleccionar otros plazos como pueden ser la *hora*, *semana*, *mes*, *año* o *todos*.

- **Obtain comments for each post**. Permite especificar si obtiene los comentarios de cada publicación. Si se activa esta opción el proceso de obtención de los datos es más lento.

- **Number of posts to obtain**. Permite especificar el número de publicaciones a obtener. Por defecto, viene establecido 25 publicaciones el cual es un número razonable ya que suele tardar unos 30 segundos en ejecutarse, cuanto más alto sea este valor más tarda la operación en finalizar.

Una vez seleccionado los valores deseados en ambos casos se pincha en el botón *Get data* y a la derecha de la barra lateral aparecerá el resultado obtenido.

Este resultado se divide en dos apartados, uno mostrando el resultado obtenido y el tipo de dato de cada columna, y otro apartado donde se muestra una tabla donde se puede visualizar de forma más amigable al usuario los datos.

C.2. Trabajo con los datos

Para este apartado pinchamos en el menú en *Reddit*, una vez pinchado en este menú tenemos tres opciones disponibles *Posts*, *Awards* y *Comments*. Para cada opción se muestra una barra lateral como el siguiente:

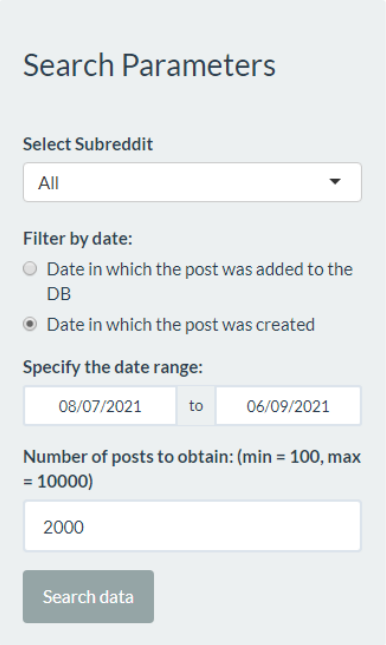

Figura C.3: Aplicación Shiny - Barra lateral del apartado Análisis de los datos

En el caso de *Awards* y *Comments* aparece un selector para seleccionar qué orden se debe aplicar para recoger los datos de la base de datos.

En esta barra lateral es posible establecer los siguientes campos:

- **Select Subreddit**. Permite indicar el subreddit del cual obtener las publicaciones. Por defecto, viene marcada la opción *popular* pero es posible especificar un subreddit en concreto. Esta opción solo esta visible en el apartado de *Posts*.

- **Select Order**. Permite indicar en qué orden se recogen los datos del servidor. Esta opción solo está visible en los apartados *Awards* y *Comments*.

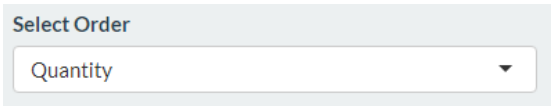

Figura C.4: Aplicación Shiny - Selector de Orden

- **Filter by date**. Permite especificar si se obtiene las publicaciones ordenadas por su fecha de creación o fecha en la que se insertó en el servidor de base de datos.

- **Specify the date range**. Permite especificar en qué rango de fechas se deben obtener los registros. Por defecto, viene establecido como fecha de fin la fecha actual y fecha de inicio 60 días anteriores a la fecha actual.

- **Number of posts to obtain**. Permite especificar el número de publicaciones a obtener.

Una vez seleccionado los valores deseados en ambos casos se pincha en el botón *Get data* y a la derecha de la barra lateral aparecerá el resultado obtenido.

En el resultado obtenido aparecerá un sub menú como aparece en la siguiente imagen:

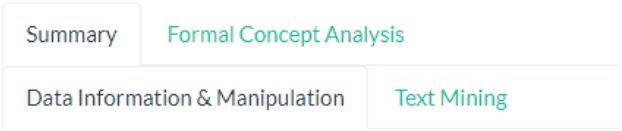

Figura C.5: Aplicación Shiny - Sub Menú del apartado de Análisis de los datos

C.2.1. Resumen

En la pestaña *Data Information & Manipulation* obtenemos diversa información sobre los datos obtenidos del servidor de base de datos.

En esta misma pantalla se ofrece la posibilidad de descargar la matriz binaria completa en un fichero CSV mediante el botón:

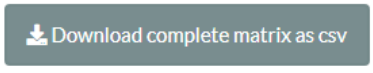

Figura C.6: Aplicación Shiny - Descargar matriz binaria

En la pestaña *Text Mining* se obtiene más información sobre los términos que aparecen en las publicaciones.

C.2.2. Análisis de Conceptos Formales

En este apartado del sub menú se divide en cuatros sub apartados tales como aparece en la siguiente imagen:

Figura C.7: Aplicación Shiny - Sub Menú del apartado de Análisis de Conceptos Formales

En la pestaña *Summary* se muestra información sobre el número de transacciones que hay en el conjunto de datos y a partir de las reglas no redundantes crea el contexto formal.

En este misma pantalla se ofrece la posibilidad de descargar el contexto formal en látex mediante el botón:

Figura C.8: Aplicación Shiny - Descargar en látex

En la pestaña *Concepts* se obtienen los conceptos para el contexto formal y se trabaja sobre estos conceptos calculados.

En la pestaña *Implications* se calculan las implicaciones para el contexto formal y se trabaja sobre estas, en la cual se intenta realizar una recomendación.

En la pestaña *Arules* se exporta las implicaciones a reglas de asociación y se muestra por pantalla diversas gráficas en las cuales se visualizan las reglas no redundantes y las reglas significantes, así mismo se ordenan las reglas por las propiedades *lift* y *support* para su posterior visualización.

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA