



# Universidad Austral de Chile

Facultad de Ciencias de la Ingeniería  
Escuela de Ingeniería Civil en Informática

## **PIP INSTALL PYSOPHIA2: UNA LIBRERÍA PYTHON PARA EL ANÁLISIS DE MEDIOS DE PRENSA INTERNACIONALES**

Proyecto para optar al título de  
**Ingeniero Civil en Informática**

PROFESOR PATROCINANTE:  
MATTHIEU VERNIER  
DOCTOR EN INFORMÁTICA

PROFESOR INFORMANTE:  
PABLO HUIJSE  
DOCTOR EN INGENIERÍA ELÉCTRICA

PROFESOR INFORMANTE:  
JUAN PABLO SALAZAR  
DOCTOR EN INGENIERÍA

**FELIPE ANDRES SALDIAS TEILLIER**

VALDIVIA - CHILE

2021

## **AGRADECIMIENTOS**

Quiero agradecer principalmente a mis padres, German Saldias Luengo y Maritza Teillier Romero por nunca bajar los brazos, creer en mí y apoyarme en esta carrera de largo aliento y siempre estar ahí cuando los necesité y sobre todo por darme las oportunidades y libertades para elegir mi camino en esta vida. Este título se lo dedico a ellos.

Me gustaría agradecer también al profesor Matthieu Vernier por ser mi profesor Patrocinante y a los profesores Pablo Huijse y Juan Pablo Salazar por tomar el rol de profesores informantes.

También quiero agradecer a todas las personas que me he cruzado en este camino, desde mi primera carrera que entré a estudiar, hasta el fin de este proceso que terminó con cambio de universidad y carrera, en especial a los cabros de viña que conocí en los últimos años, en los cuales me vi reflejado y de cierta manera vi en ellos que había opciones y caminos posibles, incluso cuando se toma una mala decisión en un inicio.

Agradecer también a todos los profesores que me enseñaron en cada una de las materias cursadas, en este punto, mención especial a Sebastián Etchegaray, Pablo Huijse y Cristobal Navarro por permitirme ejercer mi primer trabajo como informático en Motivus cuando me encontraba finalizando cuarto año, en ese año y medio aprendí muchísimo y me permitió saciar mi sed de crear y relacionarme con clientes reales.

Y por último, como dice Snoop Dogg (I Wanna Thank Me), quiero agradecerme a mí, por creer en mí, por todo el trabajo duro, por nunca abandonar y dar la lucha hasta el final, todas las desveladas haciendo trabajos valieron la pena.

## INDICE DE CONTENIDOS

|   |     |
|---|-----|
| INDICE DE CONTENIDOS.....   | I   |
| INDICE DE FIGURAS.....  | III |
| RESUMEN.....  | IV  |
| ABSTRACT .....  | V   |
| 1. INTRODUCCIÓN.....  | 1   |
| 1.1 Contexto: Sophia2, un grupo de investigación interdisciplinar para analizar el pluralismo en los medios desde la Lingüística computacional y el Machine Learning. 1 |     |
| 1.2 Origen del proyecto: tener un programa cliente que permita analizar los datos del proyecto Sophia2.....   | 1   |
| 1.3 Requisitos funcionales: ¿Qué debe permitir hacer la librería PySophia2? .....   | 3   |
| 1.4 Requisitos no funcionales: ¿Cuáles son los criterios de calidad de la librería? .....   | 5   |
| 1.5 Resumen de los objetivos del proyecto de título.....  | 6   |
| 2. MARCO TEÓRICO .....  | 7   |
| 2.1 ¿Cuáles son las librerías Python más similares a pysophia2? .....   | 7   |
| 2.2 ¿Cuáles son las buenas prácticas para desplegar una librería?.....  | 7   |
| 2.3 ¿Cuáles son las buenas prácticas para desarrollar una librería usable? .....  | 8   |
| 3. FUNCIONAMIENTO GENERAL Y BIBLIOTECAS PYTHON UTILIZADAS.....  | 10  |
| 3.1 Archivos y directorio .....   | 10  |
| 3.1 Conexiones a DB's.....  | 11  |
| 3.2 Librerías Python utilizadas .....   | 12  |
| 3.2.1 Pandas .....  | 12  |
| 3.2.2 Matplotlib .....  | 13  |
| 3.2.3 Spacy.....  | 13  |
| 3.2.4 MariaDB.....  | 13  |
| 3.2.5 ElasticSearch.....  | 13  |
| 4. HISTORIA 1: CONSTRUCCIÓN DE UN DATASET DE NOTICIAS A PARTIR DE DETERMINADOS PARÁMETROS .....   | 14  |
| 4.1 Requerimientos del problema.....  | 14  |
| 4.2 Diseño de la solución .....   | 14  |

|   |    |
|---|----|
| 4.3 Implementación .....  | 15 |
| 4.3.1 Métodos .....   | 16 |
| 5. HISTORIA 2: VISUALIZAR LA EVOLUCIÓN DE LA FRECUENCIA DE USO DE PALABRAS.....                               | 20 |
| 5.1 Requerimientos del problema.....  | 20 |
| 5.2 Diseño de la solución .....   | 20 |
| 5.3 Implementación .....  | 22 |
| 5.3.1 Métodos.....  | 22 |
| 6. HISTORIA 3: EXTRAER MÉTRICAS SOBRE LAS FUENTES DE INFORMACIÓN Y PERIODISTAS. ....                          | 26 |
| 6.1 Requerimientos del problema.....  | 27 |
| 6.2 Diseño de la solución .....   | 27 |
| 6.3 Implementación.....   | 27 |
| 6.3.1 Métodos.....  | 27 |
| 7. RESULTADOS .....   | 30 |
| 7.1 Documentación .....   | 31 |
| 7.2 Notebook de ejemplo .....   | 31 |
| 7.2.1 Construcción de un dataset de noticias a partir de determinados parámetros .....                        | 33 |
| 7.2.2 Visualizar la evolución de la frecuencia de uso de palabras.....  | 39 |
| 7.2.3 Extraer métricas sobre las fuentes de información y periodistas .....                                   | 43 |
| 8. CONCLUSIONES Y PERSPECTIVAS.....   | 47 |
| 8.1 La librería pysophia2 cumple con los requisitos funcionales identificados al inicio del proyecto .....    | 47 |
| 8.2 La librería pysophia2 cumple con los requisitos no funcionales identificados al inicio del proyecto ..... | 47 |
| 8.3 ¡La librería pysophia2 ya se está utilizando! .....   | 47 |
| 8.4 Recomendaciones y perspectivas sobre la mantención y la evolución de pysophia2.....                       | 48 |
| 9. REFERENCIAS .....  | 49 |

## INDICE DE FIGURAS

|  |    |
|--|----|
| Figura 1: Modelo Entidad-Relación de la base de datos de Sophia2 .....   | 2  |
| Figura 2: Vista de Google NGramViewer analizando la frecuencia de las palabras<br>"Frankenstein", "Sherlock Holmes" y "Albert Einsten" en un dataset conformado por<br>libros..... | 4  |
| Figura 3: Submódulos pysophia2.....  | 10 |
| Figura 4: Archivos del proyecto pysophia2.....   | 11 |
| Figura 5: Conexiones a DB utilizadas por pysophia2. ....   | 11 |
| Figura 6: Librerías Python usadas en pysophia2. ....   | 12 |
| Figura 7: DataFrame entregado al usuario al solicitar un set de noticias. ....   | 15 |
| Figura 8: Documentación para la función login(). ....  | 31 |
| Figura 9: Índice del Jupyter Notebook de ejemplo. ....   | 32 |
| Figura 10: Ejemplo de uso de la función login.....   | 33 |
| Figura 11: Ejemplo de uso de la función list_countries.....  | 33 |
| Figura 12: Ejemplo de uso de la función list_media_outlets. ....   | 34 |
| Figura 13: Ejemplo de uso de la función list_media_outlets.....  | 34 |
| Figura 14: Ejemplo de uso de la función stats_countries. ....  | 35 |
| Figura 15: Ejemplo de uso de la función stats_countries_by_date. ....  | 36 |
| Figura 16: Ejemplo de uso de la función stats_media_outlets.....   | 36 |
| Figura 17: Ejemplo de uso de la función stats_media_outlet. ....   | 37 |
| Figura 18: Ejemplo de uso de la función get_dataset sin keywords y con listado de<br>keywords y operador and.....  | 37 |
| Figura 19: Ejemplo de uso de la función get_dataset para listado de keywords con<br>operador or y por frase completa .....   | 38 |
| Figura 20: Ejemplo de uso de la función calculateDist. ....  | 39 |
| Figura 21: Ejemplo de uso de la función view generando un PySophia2NgramViewer.<br>.....   | 40 |
| Figura 22: Ejemplo de uso de la función freq para generar un gráfico con las primeras<br>50 palabras con mayor frecuencia. ....  | 41 |
| Figura 23: Ejemplo de uso de la función freq para graficar la frecuencia de una lista de<br>palabras.....  | 42 |
| Figura 24: Ejemplo de uso de la función popularity para una lista de personajes en un<br>periodo. ....   | 43 |
| Figura 25: Ejemplo de uso de la función list_sources. ....   | 44 |
| Figura 26: Ejemplo de uso de la función mentions.....  | 44 |
| Figura 27: Ejemplo de uso de la función top_mentions para obtener el top 10 de<br>personajes más mencionados en noticias entre el año 2020 y 2021.....                             | 45 |
| Figura 28: Ejemplo de uso de la función gender para chile en un periodo. ....  | 46 |

## RESUMEN

El presente trabajo se enmarca en el proyecto Fondecyt de Iniciación n° 11190714 titulado “Sophia2: métodos basados en lingüística computacional y *machine learning* para analizar el pluralismo en los medios de prensa” desarrollado en la Facultad de Ciencias de la Ingeniería (FCI) de la UACH entre diciembre 2019 y marzo 2023. Uno de los objetivos del equipo Sophia2 es construir herramientas computacionales que ayuden a los científicos en comunicación y ciencias sociales con nociones básicas en programación. Bajo este propósito, el objetivo general de este trabajo de título es el desarrollo de una biblioteca de Python llamada pysophia2 que facilita el análisis de noticias recopiladas desde diversos medios de prensa por el equipo de investigación.

Para el desarrollo de esta biblioteca se definieron tres grandes historias de usuario que fueron abordadas:

Historia 1: “COMO investigador en comunicación y ciencia social PUEDO construir un dataset de noticias de prensa a partir de ciertos parámetros (países, fechas, medios y palabras claves) PARA utilizarlo como recurso para mi investigación.”

Historia 2: “COMO investigador en comunicación y ciencia social PUEDO visualizar la evolución de la frecuencia de uso de un grupo de palabras PARA analizar la evolución del discurso de uno o varios medios de prensa.”

Historia 3: “COMO investigador en comunicación y ciencia social PUEDO extraer métricas sobre las fuentes de información PARA analizar algunos sesgos en el discurso de uno o varios medios de prensa.”

Como resultado se obtuvo una biblioteca Python totalmente funcional de fácil instalación gracias al repositorio oficial de paquetes de Python, documentada según estándares del lenguaje y que cumple con criterios usabilidad y despleabilidad definidos como requisitos no funcionales necesarios para este producto. En la actualidad, pysophia2 ya se encuentra siendo utilizada en dos proyectos relacionados al análisis de medios de prensa.

## ABSTRACT

This work is part of the Fondecyt Initiation project n° 11190714 entitled "Sophia2: methods based on computational linguistics and machine learning to analyze pluralism in the press media" developed at the Faculty of Engineering Sciences (FCI) of the UACH between December 2019 and March 2023. One of the objectives of the Sophia2 team is to build computational tools that help scientists in communication and social sciences with basic notions in programming. Under this purpose, the general objective of this thesis is the development of a Python library called pysophia2 that facilitates the analysis of news collected from various press media by the research team.

For the development of this library, three major user stories were defined and addressed:

Story 1: "AS a researcher in communication and social science I CAN build a dataset of press news from certain parameters (countries, dates, media and keywords) TO use as a resource for my research."

Story 2: "AS a researcher in communication and social science I CAN visualize the evolution of the frequency of use of a group of words TO analyze the evolution of the discourse of one or several press media."

Story 3: "AS a researcher in communication and social science I CAN extract metrics on information sources TO analyze some biases in the discourse of one or several press media."

As a result, a fully functional Python library was obtained, easy to install due to the official Python package repository, documented according to language standards and meeting usability and deployability criteria defined as non-functional requirements needed for this product. Currently, pysophia2 is already being used in two projects related to media analysis.

# 1. INTRODUCCIÓN

## 1.1 Contexto: Sophia2, un grupo de investigación interdisciplinar para analizar el pluralismo en los medios desde la Lingüística computacional y el Machine Learning

El presente trabajo se enmarca en el proyecto Fondecyt de Iniciación n° 11190714 titulado “Sophia2: métodos basados en lingüística computacional y *machine learning* para analizar el pluralismo en los medios de prensa” desarrollado en la Facultad de Ciencias de la Ingeniería (FCI) de la UACH entre diciembre 2019 y marzo 2023. En este proyecto colaboran un grupo de investigadores y estudiantes de la FCI y de la Facultad de Filosofía y Humanidades de la UACH.

Desde una perspectiva general, el proyecto Sophia2 busca crear indicadores del pluralismo en los medios de prensa basándose en algoritmos de Lingüística Computacional y *Machine Learning*. El grupo de investigadores tiene como visión proporcionar nuevas herramientas que permitan concientizar el pluralismo en los medios y activar el pensamiento crítico de los ciudadanos.

Desde 2016, este grupo desarrolla softwares y protocolos científicos para analizar fenómenos sociales y cognitivos vinculados al consumo y a la producción de noticias de prensa en Chile. En particular, podemos citar el trabajo de tesis de Magíster en Informática realizado por Fabian Ruíz (2020) cuyo objetivo principal consistía en diseñar y probar un método computacional para caracterizar la evolución de los sesgos de género en la prensa chilena basándose sobre el algoritmo *Dynamic Topic Models* (Blei & Lafferty, 2006). Este método fue probado con un caso de estudio sobre 369.860 noticias de la prensa chilena entre 2016 y 2019, y permitió obtener indicadores de sesgos de género similares a los obtenidos por la UNESCO y el *Global Media Monitoring Project* de manera más escalable. Por ejemplo, permitió caracterizar automáticamente que el discurso mediático chileno tiende a presentar las mujeres como actrices o políticas y los hombres de manera más diversa (futbolistas, políticos, expertos, líderes de opinión, líderes artísticos, etc.).

## 1.2 Origen del proyecto: tener un programa cliente que permita analizar los datos del proyecto Sophia2

Desde marzo de 2020 se han estado desarrollando distintos scripts de *crawling*, *scraping* y análisis de datos de medios de prensa organizados dentro de una arquitectura de software compleja, llamada Caleuche. Fabián Ruíz, ex alumno de la carrera, es el principal desarrollador de este software. Caleuche realiza una recopilación de los principales



medios de prensa de cada país del mundo. En su versión actual, Caleuche indexa noticias de prensa publicadas por aproximadamente 300 medios en inglés, español, francés y portugués. Se despliega Caleuche a través de contenedores Docker que manejan la persistencia de los datos, la comunicación entre componentes a través una arquitectura basada en eventos y la gestión de errores. Varios desarrolladores (estudiantes de la carrera de Ingeniería Civil en Informática) están integrando *scrapers* específicos a partir de una lista de medios internacionales. El resultado de este trabajo es una base de datos que permite manejar la información resumida en el modelo Entidad-Relación presentado en la Figura 1.

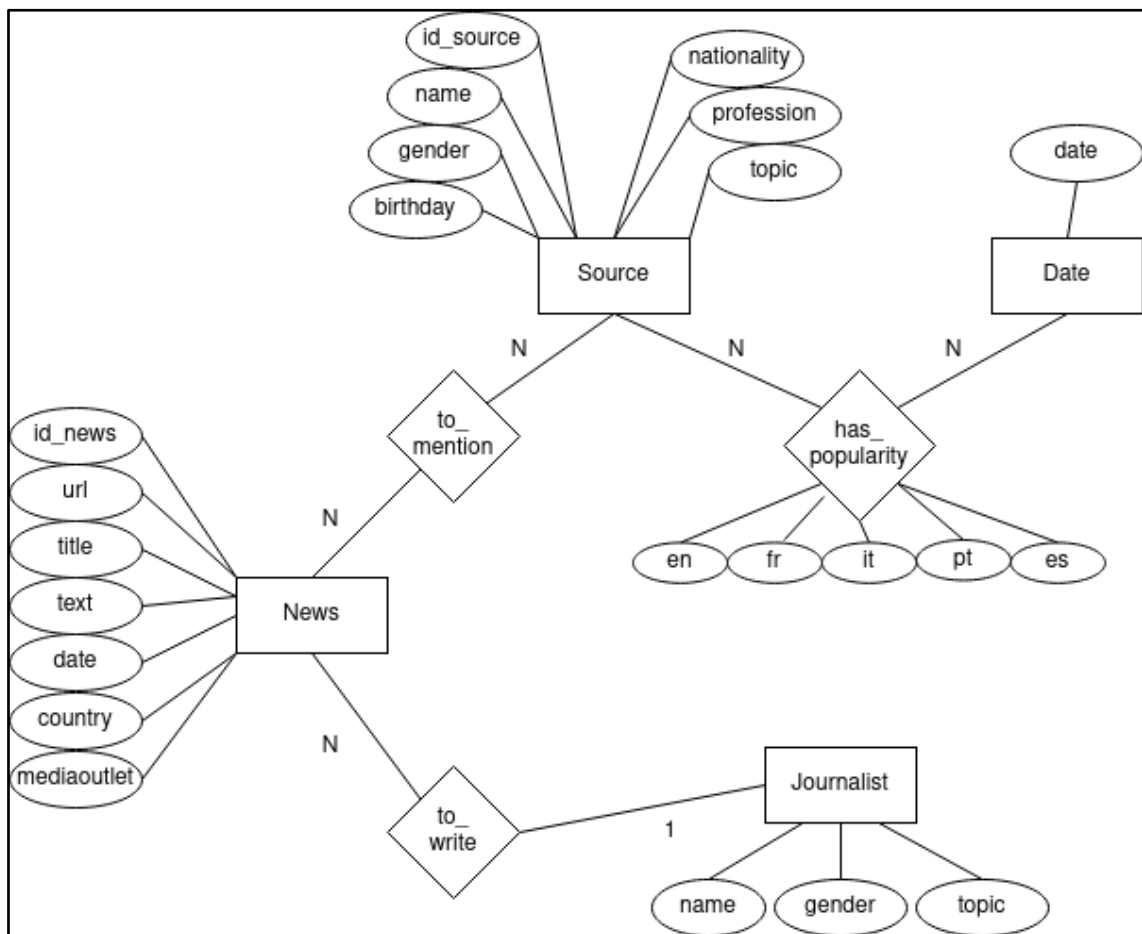


Figura 1: Modelo Entidad-Relación de la base de datos de Sophia2

Uno de los objetivos del equipo Sophia2 es construir herramientas computacionales que ayuden a los científicos en comunicación y ciencias sociales. En general, estos usuarios tienen ciertas competencias en informática y programación. Por ejemplo, pueden programar en R o Python para analizar un conjunto de datos. Sin embargo, no tienen una autonomía completa frente a necesidades técnicas más complejas como por ejemplo extraer estadísticas desde consultas SQL avanzadas o modelos de *Word Embedding*,

(Brownlee, 2017) o entrenar modelos de tópicos, etc. Este proyecto nace con el objetivo de responder a las necesidades de este tipo de usuarios.

### 1.3 Requisitos funcionales: ¿Qué debe permitir hacer la librería PySophia2?

Para responder a las necesidades de nuestros usuarios, surge la oportunidad de construir una librería de software abierta, llamada pysophia2. Los primeros usuarios de esta librería son principalmente los investigadores del grupo Sophia2, estudiantes de Comunicación y Periodismo de pregrado y postgrado de la UACH, y como una posible proyección, la librería podría representar un aporte para investigadores en comunicación y ciencia social de otras partes del mundo. Por ejemplo, la librería pysophia2 podría incluirse en una próxima edición del libro “*Computational Analysis of Communication*” (Wouter Van, Trilling, & Arcila, 2021)

Conversando con el investigador principal de Sophia2, que actúa en este contexto como el “dueño del producto” en la terminología Scrum, se identifican tres historias de usuario iniciales para organizar el desarrollo de la librería. Estas historias de usuario reflejan las funcionalidades prioritarias de pysophia2 que se desarrolla en este trabajo de título.

**Historia 1:** “COMO investigador en comunicación y ciencia social PUEDO construir un dataset de noticias de prensa a partir de ciertos parámetros (países, fechas, medios y palabras claves) PARA utilizarlo como recurso para mi investigación.”

Esta primera historia permite realizar análisis exploratorios y conocer la información almacenada en la base de datos con el objetivo de posteriormente descargar un conjunto de noticias que cumpla con mis requerimientos como investigador.

**Historia 2:** “COMO investigador en comunicación y ciencia social PUEDO visualizar la evolución de la frecuencia de uso de un grupo de palabras PARA analizar la evolución del discurso de uno o varios medios de prensa.”

Esta segunda historia se inspira directamente de la herramienta *Google NGram Viewer* (Google, 2010) (ver Figura 2). Esta herramienta permite visualizar la evolución temporal de la frecuencia para un determinado término basado en su aparición en libros publicados entre el año 1500 y el año 2018. En el caso de pysophia2, se trata de poder hacer el mismo tipo de análisis, pero a partir de datasets de noticias de prensa.

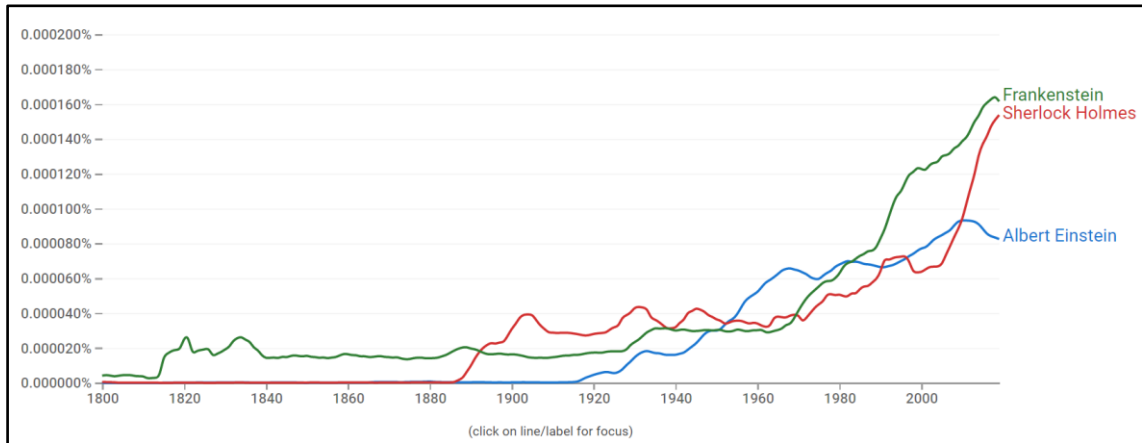


Figura 2: Vista de Google NGramViewer analizando la frecuencia de las palabras “Frankenstein”, “Sherlock Holmes” y “Albert Einsten” en un dataset conformado por libros.

**Historia 3:** “COMO investigador en comunicación y ciencia social PUEDO extraer métricas sobre las fuentes de información PARA analizar algunos sesgos en el discurso de uno o varios medios de prensa.”

Esta tercera y última historia de usuario se inspira de una extensa literatura en ciencias de la comunicación que apunta a analizar los medios de prensa. Frecuentemente en este tipo de trabajo, las preguntas de investigación cuestionan las fuentes de información y los periodistas. Por ejemplo, preguntan la distribución de fuentes femeninas o masculinas, las fuentes de información mencionadas según su nivel de popularidad o autoridad o caracterizar el tipo de discurso según los periodistas. Con pysophia2, se facilita este tipo de análisis ofreciendo acceso a distintas métricas sobre las fuentes de información y periodistas mencionados en medios de prensa. Por ejemplo, pysophia2 permite conocer cuál es la tasa de fuentes femeninas y masculinas en un medio o un conjunto de medios en un periodo determinado. Como otro ejemplo, pysophia2 permite extraer estadísticas sobre las profesiones y la popularidad de las fuentes de información mencionadas.

## 1.4 Requisitos no funcionales: ¿Cuáles son los criterios de calidad de la librería?

Además de los aspectos funcionales, las reuniones con el investigador principal de Sophia permiten identificar dos criterios de calidad esenciales que repercuten directamente en el diseño/implementación de la librería. A continuación, damos sus definiciones según la Organización Internacional para la Estandarización (ISO).

### **Usabilidad:**

La usabilidad se refiere a la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso. (ISO, 1991)

Usabilidad es la eficacia, eficiencia y satisfacción con la que un producto permite alcanzar objetivos específicos a usuarios específicos en un contexto de uso específico (ISO, 2018)

La usabilidad depende no sólo del producto si no también del usuario. Por ello un producto no es en ningún caso intrínsecamente usable, sólo tendrá la capacidad de ser usado en un contexto particular y por usuarios particulares. De manera particular, la usabilidad establece que el software debe ser comprendido, aprendido, usado y atractivo para el usuario por lo que una definición concisa de los métodos, evitar conceptos demasiado abstractos y una buena guía de para aprender a utilizar el software son esenciales para pysophia2.

### **Portabilidad/Desplegabilidad:**

Grado de efectividad y eficiencia con el que un sistema, producto o componente se puede transferir de un hardware, software u otro entorno operativo o de uso a otro. (ISO, 2011) (ISO/IEC 25010, 2011)

En particular, este requerimiento establece que la forma en la que el software llega al usuario debe ser lo más simple posible. En nuestro caso, decidimos publicar la librería `pysophia2` en el repositorio público de librerías para Python: PyPI (PyPI, 2021). Por lo tanto, el usuario puede desplegar la librería con el comando `pip install pysophia2`, lo cual instalará todas las dependencias necesarias.

## **1.5 Resumen de los objetivos del proyecto de título**

El objetivo general del proyecto consiste en desarrollar una librería Python usable y portable, que se conecta a la base de datos de Sophia2 y facilita el análisis de medios de prensa y la investigación en comunicación y ciencia social.

De manera más específica, el proyecto establece los objetivos siguientes:

1. Investigar las herramientas necesarias a utilizar y conceptos relacionados para desarrollar y difundir librerías Python.
2. Diseñar módulos, definir estándares de buenas prácticas a utilizar y establecer prioridades para el desarrollo de la librería asegurando la interoperabilidad con la arquitectura Caleuche.
3. Obtener el producto final desarrollado de manera iterativa según metodologías ágiles, asegurando un software con buena usabilidad por sus usuarios.

## 2. MARCO TEÓRICO

### 2.1 ¿Cuáles son las librerías Python más similares a pysophia2?

El repositorio PyPI (*Python Package Index*), desarrollado por la fundación Python, indexa alrededor de 317.000 librerías Python en 2021. PyPI ayuda a buscar e instalar software desarrollado por la comunidad Python, y ayuda a los autores de librerías en distribuirlas. Posicionamos la librería *pysophia2* comparándola con dos grupos de librerías existentes. Por un lado, existen librerías que permiten recopilar y analizar datos textuales utilizando métodos de *scraping* (por ejemplo: *requests*, *beautifulsoup*), de *machine learning* y de *natural language processing* (*spacy*, *gensim*, *transformers*, *torchtext*). Estas librerías son las más completas para desarrolladores experimentados integrando los últimos avances tecnológicos. Por otro lado, existen distintas iniciativas que buscan facilitar la usabilidad de estas librerías para usuarios menos experimentados en programación. Se puede citar en particular una librería como *newspaper3k* que apunta a facilitar el *scraping* de noticias de prensa encapsulando *requests* y *beautifulsoup*. También esta *textacy* que encapsula *spacy*, o más recientemente *socialscience.ai* que apunta explícitamente a acercar los científicos sociales a técnicas de inteligencia artificial, a través de documentación y tutoriales reproducibles.

*Pysophia2* comparte la misma idea de facilitar el uso de librerías de inteligencia artificial para científicos sociales. La diferencia principal con librerías como de *socialscience.ai* y *textacy* se encuentran en el ámbito de aplicación de *pysophia2*, esta se enfoca específicamente en el contexto del análisis de los datos de medios de prensa.

### 2.2 ¿Cuáles son las buenas prácticas para desplegar una librería?

Las bibliotecas de Python son fácilmente instalables mediante el comando “*pip install library\_name*” cuando estas se encuentran alojadas en PyPI. Es la manera más común de asegurar la disponibilidad del software para todo quien desee utilizarlo sin tener que disponer de complicadas instalaciones. Para alojar una librería en PyPI, la primera etapa consiste en crear una cuenta en su página web, tanto en el repositorio de test (test PyPI, s.f.), como en su versión oficial (PyPI, 2021).

Para publicar una librería en PyPI es necesario contar con dos paquetes de Python (Bienvenu, 2020) indispensables para este propósito “*wheel*”, el cual se encargará de codificar nuestra librería en el formato binario aceptado y basado en el estándar PEP427

(PEP427, 2012), y “*twine*” que facilitará el proceso de publicación en PyPI, ambos disponibles para instalar mediante el comando *pip install*. Una vez instalados estos paquetes es necesario crear en la carpeta raíz el archivo de configuración “*setup.py*” el cual especificará diversos datos de la librería tales como: *name*, *version*, *description*, *url*, *author*, *license*, *language*, paquetes que contiene la librería y los paquetes externos requeridos utilizados por esta.

Para el campo “*version*” se utiliza el estándar ampliamente aceptado llamado versionamiento semántico, el cual consta de asignar la versión siguiendo un estándar *major.minor.patch* en donde cada uno de estos números se refiere a lo siguiente:

- **major:** aumenta cuando existe un cambio en el software que será incompatible con las versiones anteriores de este.
- **minor:** se aumenta para indicar que ha sido añadida alguna nueva funcionalidad o deprecado alguna antigua, pero el código sigue siendo retro compatible.
- **patch:** se utiliza para denotar que se ha arreglado algún bug y este cambio es retro compatible con versiones anteriores.
- 

Una vez que se tiene el archivo de configuración preparado, ha llegado el momento de publicar la librería, para lo cual se seguirán los siguientes pasos:

1. Compilar el código mediante el comando `python setup.py sdist bdist_wheel`.
2. Verificar que la construcción de la compilación fue exitosa mediante el comando `twine check dist/*`.
3. Publicar la librería en el repositorio de test mediante `twine upload --repository-url https://test.pypi.org/legacy/ dist/*`.
4. Si todo lo anterior funcionó correctamente se puede publicar la librería en el repositorio oficial de PyPI utilizando `twine upload dist/*`.

## 2.3 ¿Cuáles son las buenas prácticas para desarrollar una librería usable?

Antes de publicar una librería es necesario enfocarse en escribir un código de calidad, organizar las diferentes funcionalidades de la librería en submódulos que agrupen funciones de ámbitos similares, cuyos nombres son pilares fundamentales para obtener una librería de calidad, tanto en lado de quien la usará, como de quien seguirá desarrollando, agregando funcionalidades o realizando el mantenimiento y

actualizaciones correspondientes necesarias por eventuales incompatibilidades de versiones en las bibliotecas utilizadas en la construcción de la librería. Con el objetivo de asegurar un Código limpio y de fácil comprensión es recomendado utilizar el estándar definido por la guía de estilos de Python (PEP8, 2001).

Otro punto importante y necesario, es la correcta documentación de las funciones mediante el estándar de Python, en donde se debe incluir una descripción para cada función, así como también sus parámetros con su tipo de dato y una breve explicación de su propósito con el objetivo de clarificar totalmente la interfaz disponible de cara al usuario final.



### 3. FUNCIONAMIENTO GENERAL Y BIBLIOTECAS PYTHON UTILIZADAS

#### 3.1 Archivos y directorio

La biblioteca pysophia2 está construida en el lenguaje de programación Python y fue diseñada para ser utilizada en el mismo. La versión final consta de la creación de un paquete de Python llamado `pysophia2.py` utilizado para encapsular los submódulos *db* y *tools*, este último a su vez encapsula un submódulo (ver Figura 3) llamado *ngram*, que se encarga de diferentes funcionalidades mencionadas en la sección ‘1.3’. En la Figura 4 se puede ver todos los archivos y carpetas del directorio del proyecto.

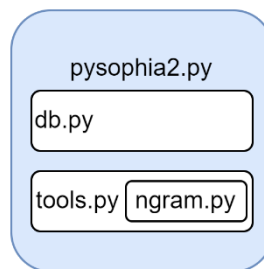


Figura 3: Submódulos *pysophia2*

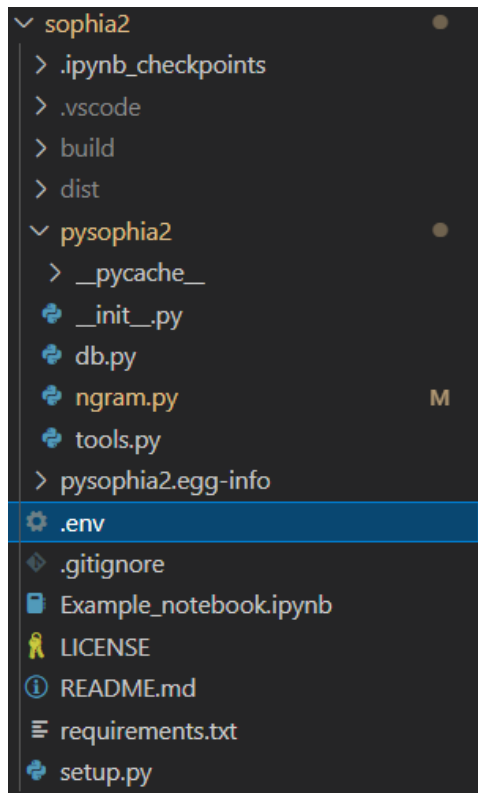


Figura 4: Archivos del proyecto pysophia2.

### 3.1 Conexiones a DB's

Pysophia2 utiliza y maneja dos conexiones simultáneas a bases de datos (DB), por un lado, una conexión a una DB relacional alojada en una instancia de MariaDB, y, por otro lado, una conexión a una DB no relacional: el motor de búsqueda y analítica *Elasticsearch*. Lo anterior puede verse reflejado en la Figura 5.

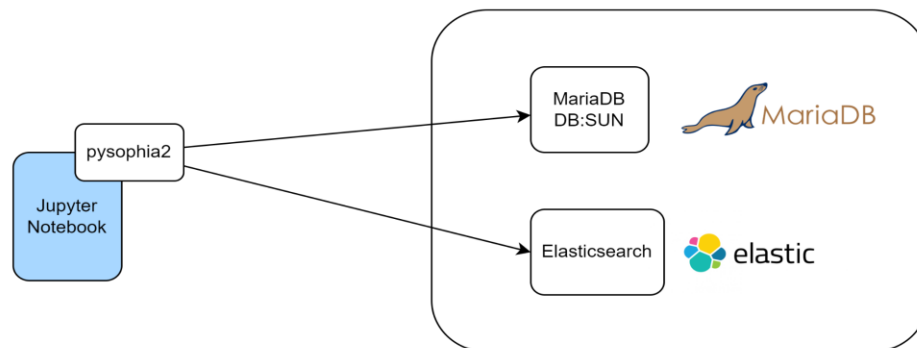


Figura 5: Conexiones a DB utilizadas por pysophia2.

## 3.2 Librerías Python utilizadas

En la construcción de pysophia2 han sido utilizadas múltiples librerías de Python, muchas de ellas comúnmente utilizadas en el área de la inteligencia artificial, procesamiento y visualización de datos (ver Figura 6).



*Figura 6: Librerías Python usadas en pysophia2.*

A continuación, se encuentra una pequeña descripción de cada una de las librerías utilizadas y su uso dentro del proyecto pysophia2.

### 3.2.1 Pandas

Es una herramienta de código abierto potente, flexible y fácil de utilizar para el análisis de datos y la manipulación de estos (Pandas, 2021). Pandas juega un papel muy importante en la representación de las noticias obtenidas de la base de datos, los cuales son almacenados en DataFrames (DataFrame, 2021). Gracias a esta herramienta es posible realizar tareas como: manipulación de los *datasets*, agrupar por fechas, exportar *datasets* a formato csv entre otras

### **3.2.2 Matplotlib**

Es una biblioteca de código abierto enfocada en la generación de visualizaciones de datos que permite crear gráficos de gran calidad (Matplotlib, 2021). En pysophia2, matplotlib es utilizado para visualizar la evolución temporal de la frecuencia para determinados conceptos/palabras claves.

### **3.2.3 Spacy**

Es una biblioteca gratuita de código abierto para realizar tareas avanzadas de procesamiento del lenguaje natural (NLP) en Python diseñada específicamente para su uso en aplicaciones en producción (Spacy, 2021). Spacy es utilizada en pysophia2 para la extracción de conceptos y conceptos compuestos con un alto rendimiento en los cuerpos de las noticias almacenadas en Sun.

### **3.2.4 MariaDB**

Es una biblioteca que contiene un cliente conector SQL para conectarse en tiempo real y realizar consultas a una base de datos alojada en un servidor de MariaDB (MariaDB, 2021). En pysophia2 la biblioteca mariadb es utilizada para establecer conexiones con la base de datos oficial del proyecto sophia2 con el objetivo de calcular estadísticas y recuperar información de la base de datos.

### **3.2.5 ElasticSearch**

Es el cliente oficial de bajo nivel para Python, su objetivo es proporcionar un terreno común para el código relacionado con ElasticSearch, permitiendo recuperar datos desde la base de datos con un variado número de filtros y búsquedas por match de texto (ElasticSearch, 2021). En pysophia2 esta biblioteca es utilizada para recuperar y descargar noticias alojadas en el servidor con gran flexibilidad para búsqueda en textos en tiempos muy superiores a lo que se podría optar con una base de datos relacional común.

## 4. HISTORIA 1: CONSTRUCCIÓN DE UN DATASET DE NOTICIAS A PARTIR DE DETERMINADOS PARÁMETROS

### 4.1 Requerimientos del problema

- El/la científico/a social quiere construir un *dataset* de noticias a utilizar como recurso para su investigación. El *dataset* puede obtenerse mediante distintos parámetros:
  - **país**: el país a considerar.
  - **fechas**: un periodo de tiempo definido por una fecha de inicio y una fecha de fin.
  - **palabra clave**: una palabra que permite obtener las noticias que contienen esta palabra. La palabra puede ser simple (por ejemplo: “contaminación”) o compleja (por ejemplo: “contaminación del aire”).
- El *dataset* obtenido debe poder exportarse a un archivo CSV. El usuario debe poder tener la garantía que no existen noticias duplicadas (que comparten la misma URL) en el *dataset* y eliminar los datos duplicados si existen.
- El usuario debe poder agrupar el resultado de varias consultas en un solo archivo y verificar que no existen datos duplicados.
- El usuario debe poder listar todos los países de los cuales puede obtener noticias y los medios de comunicación en cada uno de estos.
- El usuario puede conocer la cantidad de noticias disponibles para cada uno de los medios de comunicación.

### 4.2 Diseño de la solución

Las funcionalidades para esta historia de usuario han sido encapsuladas en un submódulo de pysophia2 llamado “db” en el cual se establecen dos tipos de conexiones:

- **ElasticSearch**: se utiliza para almacenar una copia de todas las noticias que se han guardado en la base de datos principal. Se ha optado por esta herramienta debido a su capacidad de realizar filtros en el lado del servidor que permite buscar las noticias que contengan un determinado texto en velocidades bastante aceptables, evitando tener que traer todas las noticias para un país, medio y periodo de tiempo y realizar este filtro manualmente

en el computador mediante la librería, previniendo así desperdicio de banda ancha y capacidad computacional. La principal limitante de este método es que solo puede recuperar un máximo de 10.000 noticias por consulta, por lo que en esta versión 0.0.1, si la cantidad de noticias es mayor a 10.000, se sugiere al usuario acotar las fechas utilizadas en la búsqueda.

- **MariaDB:** la base de datos SUN se encuentra alojada en una instancia de servidor de MariaDB, una base de datos de tipo SQL caracterizada por su alta velocidad en consultas debido a su caché en RAM, a la cual se accede mediante credenciales con permiso de solo lectura. Esta conexión es utilizada para obtener estadísticas e información de las noticias almacenadas, como, por ejemplo: países que poseen noticias en ellas, medios de comunicación por cada país, cantidad de noticias entre una determinada fecha, etc.

Todos los requerimientos del problema son resueltos por funciones de manera individual, es decir, existe un grupo de funciones que permite navegar en la base de datos de manera exploratoria, permitiendo conocer estadísticas de las noticias almacenadas en SUN, así como también obtener listados de los países y medios de prensa para cada uno de estos. Una vez que se tiene claro y definidos los parámetros por los cuales se desea realizar la búsqueda, se debe ejecutar una función que retornará un DataFrame que contiene las noticias que cumplen con los parámetros especificados, el cual puede visualizarse en la Figura 7.

|      | id_news   | country | media_outlet | url   | title   | text  | date       | search           |
|------|-----------|---------|--------------|---|---|---|------------|------------------|
| 5    | 5306786.0 | chile   | elciudadano  | https://www.elciudadano.com/medio-ambiente/pre... | Premio Nobel indio dice que el cambio climátic... | El científico indio Rajendra Pachauri, Nobel ...  | 2017-05-29 | cambio climático |
| 10   | 5294966.0 | chile   | elciudadano  | https://www.elciudadano.com/medio-ambiente/bea... | Beatriz Sánchez advierte que Chile es "muy vul... | En el foro "El gran reto de Chile ante el camb... | 2017-09-07 | cambio climático |
| 11   | 5295324.0 | chile   | elciudadano  | https://www.elciudadano.com/medio-ambiente/bea... | Beatriz Sánchez advierte que Chile es "muy vul... | En el foro "El gran reto de Chile ante el camb... | 2017-09-07 | cambio climático |
| 24   | 5295556.0 | chile   | elciudadano  | https://www.elciudadano.com/medio-ambiente/est... | Estos son los países con mayores y menores pro... | Hace dos años, la Universidad de Notre Dame ...   | 2017-09-07 | cambio climático |
| 31   | 5305579.0 | chile   | elciudadano  | https://www.elciudadano.com/medio-ambiente/gob... | Gobierno chileno manifiesta su "profunda decep... | Una «profunda decepción» manifestó el gobiern...  | 2017-06-01 | cambio climático |
| ...  | ...       | ...     | ...          | ...   | ...   | ...   | ...        | ...              |
| 6893 | 6000473.0 | chile   | elciudadano  | https://www.elciudadano.com/justicia/el-genero... | El género de cada uno... ¿y qué?                  | A fines de abril el Ministerio de Educación ...   | 2017-07-08 | cambio climático |

Figura 7: DataFrame entregado al usuario al solicitar un set de noticias.

## 4.3 Implementación

La implementación del módulo *db* posee dos variables globales del tipo diccionario para gestionar los datos y las conexiones, tanto para mariadb, como para elasticsearch, en las cuales se almacenan todos los datos necesarios para establecer la conexión y el objeto utilizado para interactuar con el servidor.

```

sql_vars={
    "__username__": "",
    "__password__": "",
    "__cursor__": None,
    "__host__": "45.79.130.8",
    "__port__": 14096,
    "__database__": "Sun"

elastic_vars = {
    "__ip__" : "45.56.113.162",
    "__port__": 9200,
    "__user__": "elastic",
    "__password__": "uZCxWGE35lD3lhc",
    "__es__": Elasticsearch(
        ["45.56.113.162:9200"],
        http_auth=("elastic",
                   "uZCxWGE35lD3lhc"),
        timeout=60
    )
}

```

### 4.3.1 Métodos

A continuación, se describirán los métodos implementados en la librería para dar cumplimiento a los requerimientos de la primera historia de usuario. Estas funciones utilizarán alguna de las variables para conexiones descritas en el párrafo anterior.

- **login():** este método está disponible en la api de la biblioteca y se encarga de guardar los parámetros de conexión a mariadb en la variable `sql_vars`, posteriormente ejecuta la función `connect_sun()`.

- Parámetros:

- **\_\_username\_\_**: es un parámetro de carácter obligatorio utilizado para el nombre de usuario de mariadb.
  - **\_\_password\_\_**: es un parámetro de carácter obligatorio utilizado para la contraseña del usuario.
  - **\_\_host\_\_**: contiene la ip del servidor mariadb.
  - **\_\_port\_\_**: puerto donde se encuentra la aplicación.
  - **\_\_database\_\_**: nombre de la base de datos.
- **connect\_sun()**: este método es de uso interno y es el encargado de establecer la conexión con la base de datos. No tiene parámetros de entrada y retorna un mensaje que indica si la conexión fue establecida satisfactoriamente o no.
  - **\_\_connection\_validator()**: este método sin parámetros es de uso interno y permite verificar que la conexión SQL fue establecida correctamente, y es el primero en ejecutarse en cada función que accede a la base de datos. En caso de no encontrar una conexión despliega un mensaje al usuario recordando realizar el primer paso de login.
  - **list\_countries()**: este método sin parámetros está disponible en la api de la biblioteca de entrada devuelve una lista de todos los países que tienen noticias dentro de la base de datos de sophia2.
  - **list\_media\_outlets()**: este método está disponible en la api de la biblioteca y retorna una lista con todos los medios de prensa que tienen noticias almacenadas en la base de datos de sophia2.
    - Parámetros:
      - **countries**: es un parámetro de carácter opcional, una lista de países para los cuales se quiere conocer los medios de prensa con noticias. Si no se envía este parámetro se retornan todos los medios de prensa para todos los países en la base de datos.



- **stats\_countries():** este método sin parámetros de entrada está disponible en la api de la biblioteca y retorna estadísticas de la cantidad de noticias para cada país dentro de la base de datos de sophia2.
- **stats\_countries\_by\_date():** este método está disponible en la api de la biblioteca y es similar al anterior, pero es posible determinar un período para el cálculo de las estadísticas.
  - Parámetros:
    - **\_from:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
    - **\_to:** es un parámetro de carácter obligatorio y corresponde a la fecha de término del período en formato YYYY/MM/DD.
- **stats\_media\_outlet():** este método está disponible en la api de la biblioteca y permite obtener las estadísticas de la cantidad de noticias para cada medio de prensa en la base de datos sophia2.
- **get\_dataset():** este método está disponible en la api de la biblioteca y retorna un DataFrame en donde cada fila corresponderá a una noticia de las solicitadas por el usuario para un determinado país, en un determinado período y con la posibilidad de obtener noticias que contengan determinadas palabras clave o una frase.
  - Parámetros:
    - **country:** es un parámetro de carácter obligatorio y corresponde al nombre del país del cual se quieren obtener las noticias.
    - **\_from:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
    - **\_to:** es un parámetro de carácter obligatorio y corresponde a la fecha de término del período en formato YYYY/MM/DD.
    - **keywords:** es un parámetro de carácter opcional. En caso de querer obtener noticias que, en su cuerpo, contengan una o más palabras se debe pasar un *string* con estas palabras separadas por un espacio de la siguiente manera: “PALABRA1 PALABRA2 PALABRA3”.

Para el caso de querer obtener todos los textos que contengan una determinada frase, esta debe ser escrita en un *string*, por ejemplo “esta es una frase”.

- **keywords\_operator:** este es un parámetro de carácter opcional y solo tiene efecto cuando estamos buscando textos que tengan más de una palabra, “PALABRA1 PALABRA2”. Este parámetro puede ser “or” o “and”. Para el primer caso, *or*, se descargan todas las noticias que contengan al menos una de las palabras del parámetro keywords. Para el segundo caso, *and*, es necesario que ambas palabras se encuentren presentes en el texto de una noticia para ser descargadas de la base de datos. Si no se entrega este parámetro, se establecerá el parámetro “or” por defecto.
- **phrase:** este es un parámetro de carácter opcional y funciona como un *flag* que indica si el *string* entregado en el parámetro *keywords* corresponde a una palabra (o lista de palabras) o a una frase completa. Si no se entrega este parámetro, es un *False* por defecto
- **export\_to\_csv():** este método está disponible en la api de la biblioteca y permite exportar un conjunto de noticias retornado por *get\_dataset()* a un formato csv, básicamente es un *wrapper* de la función *to\_csv()* de la librería pandas.
  - Parámetros:
    - **df:** es un parámetro de carácter obligatorio. Contiene el DataFrame con las noticias que desea ser exportado a csv.
    - **name:** es un parámetro de carácter opcional. Se refiere al nombre que tendrá el archivo csv de salida. En caso de no ser pasado el archivo se guardará como “*default.csv*”

## 5. HISTORIA 2: VISUALIZAR LA EVOLUCIÓN DE LA FRECUENCIA DE USO DE PALABRAS.

### 5.1 Requerimientos del problema

- El/la científico/a social quiere saber cómo evoluciona el uso de ciertas palabras en un dataset de noticias.
  - **dataset**: es un conjunto de noticias de prensa en formato DataFrame obtenidas mediante la función `get_dataset()` .
  - **keywords**: las palabras que el usuario quiere monitorear. Las palabras pueden ser simples (“contaminación”) o complejas (“contaminación del aire”).
  - **group\_by**: espacio temporal por el que se quieren agrupar las noticias para visualizar la evolución de las palabras, puede ser “day|month|year”.
- El usuario debe poder generar una visualización para un determinado arreglo de conceptos.
- El usuario debe poder obtener la frecuencia de un arreglo de palabras en un determinado conjunto de noticias. Se debe poder generar un gráfico con la visualización de esta información.
- El usuario debe poder obtener los términos con mayor frecuencia para determinado conjunto de noticias.

### 5.2 Diseño de la solución

La solución para esta funcionalidad se encuentra encapsulada en un submódulo llamado *tools* y, dentro de este, en su propio submódulo llamado *ngram*. La visualización generada por esta herramienta se ha denominado `PySophia2NgramViewer`.

Con el objetivo de optimizar las visualizaciones, y evitar realizar el cálculo de las frecuencias cada vez que se requiere generar una gráfica para el mismo conjunto de noticias, se ha optado por pre calcular y almacenar en una estructura de frecuencias el conteo de palabras para cada periodo solicitado. De esta manera es posible utilizar esta estructura de datos con las frecuencias pre calculadas para generar, en breve tiempo de ejecución, las visualizaciones de `PySophia2NgramViewer` para los términos que se requiera.

Para obtener las visualizaciones requeridas, PySophia2NgramViewer y gráfico de frecuencias, es necesario realizar una serie de pasos descritos a continuación:

- Obtener un diccionario de distribución de frecuencias para los términos en cada uno de los periodos (día, mes o año). Para esto es necesario:
  - Agrupar todas las noticias por el periodo requerido en un diccionario cuya clave contenga la fecha.
  - Obtener los términos simples y compuestos para cada una de las noticias contenidas en este dataset. Para los términos simples se utiliza (Doc, 2021) que permite obtener una secuencia de tokens (Token, 2021) para un texto dado como entrada y de esta manera obtener todos los sustantivos, adjetivos y verbos. En el caso de los términos compuestos se utiliza la herramienta matcher (Matcher, 2021) que permite extraer para un texto dado los conceptos que cumplan con los siguientes patrones:
    - “SUSTANTIVO-de-SUSTANTIVO”,
    - “SUSTANTIVO-del-SUSTANTIVO”,
    - “SUSTANTIVO-ADJETIVO”
  - Calcular la frecuencia para los términos encontrados en cada uno de los periodos y su conjunto de noticias para este subconjunto de noticias, para esto se utiliza el objeto Counter de la librería por defecto de *Python collections*. Esta estructura podría ser utilizada para generar la visualización pysophia2 posteriormente.
- Utilizar este diccionario de frecuencias agrupado por periodo para realizar las visualizaciones requeridas. Se crea un conjunto de puntos para cada uno de los términos solicitados en donde el eje x representa el periodo y el eje y corresponde a la frecuencia de apariciones de este término en los textos de ese periodo.

## 5.3 Implementación

El submódulo ngram tienen una variable global llamada `ngram_vars` en forma de diccionario que almacena los dos componentes principales tal y como se puede ver a continuación:

```
ngram_vars={
    "nlp": "",
    "matcher": ""
}
```

La clave `nlp` es utilizada para almacenar un pipeline configurado con el modelo de lenguaje español descargado de spacy: “`es_core_news_md`” (`es_core_news_md`, 2021) utilizado para la extracción de los términos simples.

La clave `matcher` es utilizada para almacenar el matcher utilizado en la extracción de los términos compuestos por más de una palabra.

Al importar este módulo, se realiza una verificación para conocer si el usuario tiene instalado en su sistema el modelo “`es_core_news_md`”. En caso de no encontrarse, se instalará y se le pedirá al usuario reiniciar el *kernel*. Por otro lado, si el usuario tiene el modelo descargado, se ejecutará una función de configuración encargada de llenar las variables `ngram_vars` tal y como se puede ver en el siguiente código:

```
if not util.is_package("es_core_news_md"):
    subprocess.run("python -m spacy download es_core_news_md")
    print("...instalando el modelo de lenguaje 'es_core_news_md', por favor reiniciar el kernel y vuelva a importar la librería")
else:
    setup()
```

### 5.3.1 Métodos

A continuación, se describirán los métodos implementados en la librería para dar cumplimiento a los requerimientos de la segunda historia de usuario:

- **setup():** este método sin parámetros es de uso interno y se encarga de configurar las herramientas de NLP, cargar el modelo de lenguaje en español y configurar el matcher y sus respectivos patrones para conceptos formados por más de una palabra.
- **get\_keywords():** este método de uso interno se encarga de agregar una nueva columna llamada keywords, al DataFrame de noticias, en la cual se almacenarán, separadas por coma, todos los términos encontrados por el modelo de lenguaje y el matcher.
  - Parámetros:
    - **dataFrame:** parámetro de carácter obligatorio. Corresponde al DataFrame para el cual se requiere encontrar las palabras clave.
- **get\_frequency():** este método de uso interno se encarga de calcular la frecuencia para cada uno de los términos que se encuentran en la columna “keywords” en un determinado dataframe. Se utiliza el método Counter() de la librería collections de Python por ser altamente eficiente.
  - Parámetros:
    - **dataFrame:** parámetro de carácter obligatorio. Corresponde al DataFrame para el cual será calculada la frecuencia de los términos clave.
- **group\_by\_date():** este método de uso interno se encarga de agrupar un dataframe de noticias por un determinado periodo de tiempo establecido por el usuario: día, mes o año. Se utiliza la función groupby de pandas (groupby, 2021) que permite agrupar filas que contengan valores comunes para en ciertas columnas.
  - Parámetros:
    - **dataFrame:** parámetro de carácter obligatorio. Corresponde al DataFrame que será agrupado por periodos.

- **granularity:** parámetro de carácter obligatorio. Corresponde al espacio temporal por el cual serán agrupadas las noticias, este parámetro puede tomar 3 valores, “*day*”, “*month*” o “*year*”.
- **calculateDist():** este método está disponible en la api de la biblioteca y permite calcular una estructura de frecuencias para un conjunto de noticias. Esta estructura es un diccionario en donde cada uno de los elementos tiene el siguiente formato:

```
'periodo1': {
  'total_news': 'cantidad de noticias en el periodo 1',
  'total_elements': 'cantidad de palabras',
  'dictionary':
    {
      'palabra1': 'frecuencia',
      'palabra2': 'frecuencia',
      'palabra3': 'frecuencia'
    }
}
```

- Parámetros:
    - **dataset:** parámetro de carácter obligatorio. corresponde al DataFrame para el cual se requiere calcular la estructura de frecuencias.
    - **group\_by:** parámetro de carácter obligatorio. Corresponde al marco temporal por el cual se agruparán las noticias y calcularán las frecuencias de sus respectivas palabras y términos compuestos.
- **view():** este método está disponible en la api de la biblioteca y permite generar una visualización PySophia2NgramViewer. Para esto es utilizada la estructura de frecuencias para extraer la cantidad de veces que es nombrada una palabra en un determinado periodo, y la librería matplotlib para graficar esta información.
- Parámetros:
    - **world\_distribution:** parámetro de carácter obligatorio. Corresponde a la estructura de frecuencias de un determinado dataset de noticias.

- **words:** parámetro de carácter obligatorio. Corresponde a un arreglo de string, estas pueden ser palabras simples o términos compuestos.
  - **annotate:** parámetro de carácter obligatorio. Es un booleano que activa o desactiva las anotaciones insertas en el gráfico de la cantidad específica para la frecuencia de una palabra.
  - **normalize:** parámetro de carácter obligatorio. Es un booleano que permite generar la visualización en base a la cantidad total de apariciones para una palabra o a la frecuencia relativa de esta palabra con respecto a la cantidad total de noticias del periodo.
- **freq():** este método está disponible en la api de la biblioteca y permite obtener una lista con las palabras más frecuentes en una determinada estructura de frecuencias. También es posible obtener la frecuencia para un conjunto de términos definidos de manera arbitraria.
  - Parámetros:
    - **word\_distribution:** parámetro de carácter obligatorio. Corresponde a la estructura de frecuencias de un determinado dataset de noticias.
    - **top\_n:** parámetro de carácter obligatorio y excluyente con el parámetro words. Acepta un valor numérico entero e indica el número de palabras que se quiere obtener, es decir, las n palabras más frecuentes de la estructura de frecuencias.
    - **words:** parámetro de carácter obligatorio y excluyente con el parámetro top\_n. Corresponde a un arreglo de string, que pueden ser palabras simples o términos compuestos, y se obtendrá la frecuencia para cada una de ellas.



- **graph:** parámetro de carácter opcional. Corresponde a un booleano que activa/desactiva la generación de un gráfico que permite visualizar las frecuencias para las palabras solicitadas.
- **normalize:** parámetro de carácter opcional. Corresponde a un booleano que activa/desactiva la normalización en las frecuencias del gráfico.

## **6. HISTORIA 3: EXTRAER MÉTRICAS SOBRE LAS FUENTES DE INFORMACIÓN Y PERIODISTAS.**

El científico requiere extraer datos relacionados y diferentes tipos de información relacionado a los personajes que aparecen mencionados en las noticias almacenadas en las noticias de Sun.

## **6.1 Requerimientos del problema**

- El usuario debe poder obtener los nombres de todos los personajes que aparecen en las noticias para un determinado país dentro de un periodo de tiempo definido entre fechas.
- El usuario debe poder obtener la popularidad de un personaje dentro de un periodo de tiempo entre dos fechas.
- El usuario debe poder obtener todas las noticias que involucran a un personaje determinado por él.
- El usuario debe poder obtener un listado de los personajes más nombrados en un país.
- El usuario debe poder conocer estadísticas relacionadas a la cantidad de personas por sexo de los personajes que aparecen en las noticias de un país en un determinado periodo de tiempo.

## **6.2 Diseño de la solución**

El enfoque elegido para dar cumplimiento a los requerimientos solicitados considera obtener toda la información mediante consultas de SQL realizadas a través del cliente mariadb para Python. Se ha optado por definir funciones dentro del submódulo DB de la biblioteca con el objetivo de mantener la consistencia, es decir, todas las funciones que acceden a alguna de las bases de datos se encuentran en este módulo.

Se definió una función para cada uno de estos requerimientos en una consulta independiente, la cual se encarga de realizar la consulta, obtener la respuesta, y posteriormente formatear estos datos en un DataFrame con las columnas según corresponda a la pregunta.

## **6.3 Implementación**

A continuación, se describen cada uno de los métodos creados para estas funcionalidades

### **6.3.1 Métodos**

- **list\_sources():** este método está disponible en la api de la biblioteca y permite obtener un listado de todos los personajes que aparecen nombrados en las noticias de un determinado país para un determinado periodo.
  - Parámetros:
    - **country:** parámetro de carácter obligatorio. Corresponde al país para el cual se requiere obtener el listado de los personajes.
    - **\_from:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
    - **\_to:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
  
- **popularity():** este método está disponible en la api de la biblioteca y permite obtener la popularidad mensual para una lista de personajes. El puntaje de la popularidad se encuentra registrado en la base de datos de sophia2 y es calculado mediante una metodología elaborada por Eleazar Vásquez, estudiante perteneciente al grupo de investigación sophia2.
  - Parámetros:
    - **sources:** parámetro de carácter obligatorio. Corresponde a un arreglo de *strings* en el cual cada elemento es el nombre de un personaje.
    - **\_from:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
    - **\_to:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
  
- **mentions():** Este método está disponible en la api de la biblioteca y permite obtener todas las noticias en las cuales han sido mencionados una lista de personajes determinada por el usuario.
  - Parámetros:

- **sources:** parámetro de carácter obligatorio. Corresponde a un arreglo de strings en el cual cada elemento es el nombre de un personaje.
  - **\_from:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
  - **\_to:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
- **gender():** este método está disponible en la api de la biblioteca y permite obtener la cantidad de veces que son mencionado en el contenido de las noticias personajes del sexo masculino vs personajes del sexo femenino. Los sexos se encuentran asignados en la base de datos para cada uno de los personajes.
  - Parámetros:
    - **country:** parámetro de carácter obligatorio. Corresponde al país para el cual se requiere obtener el listado de los personajes.
    - **\_from:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
    - **\_to:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
- **top\_mentions():** este método está disponible en la página de la biblioteca y permite obtener un ranking de los personajes con más menciones en las noticias de un país para un determinado periodo.
  - Parámetros:
    - **country:** parámetro de carácter obligatorio. Corresponde al país para el cual se calculará el ranking de los personajes.

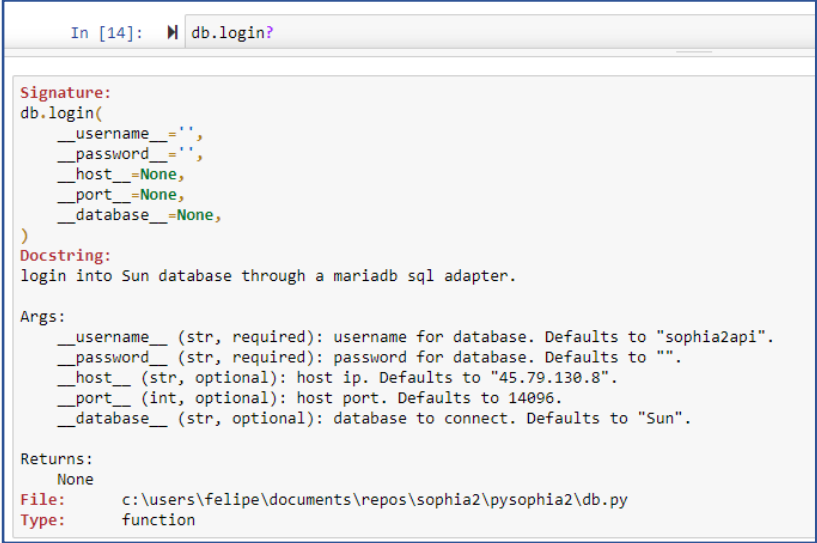
- **top\_n:** parámetro de carácter obligatorio. Corresponde al largo del ranking que se quiere obtener.
- **\_from:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.
- **\_to:** es un parámetro de carácter obligatorio y corresponde a la fecha de inicio del periodo en formato YYYY/MM/DD.

## 7. RESULTADOS

Como resultado de este proyecto se obtuvo una biblioteca completamente funcional alojada en el repositorio oficial de paquetes de Python (Sophia2, 2021). La librería es fácilmente instalable como cualquier otro paquete de Python alojado en PyPI, su código fuente es de libre acceso y puede ser consultado por todo quien lo desee (Sophia2, 2021).

## 7.1 Documentación

Todas las funciones presentes, tanto las de interfaz de usuario como las de uso interno se encuentran debidamente documentadas según el estándar definido en la documentación oficial de Python (Python, 2021) y pueden ser consultadas mediante el signo de interrogación (ver Figura 8).



```
In [14]: db.login?

Signature:
db.login(
    __username__='',
    __password__='',
    __host__=None,
    __port__=None,
    __database__=None,
)
Docstring:
login into Sun database through a mariadb sql adapter.

Args:
__username__ (str, required): username for database. Defaults to "sophia2api".
__password__ (str, required): password for database. Defaults to "".
__host__ (str, optional): host ip. Defaults to "45.79.130.8".
__port__ (int, optional): host port. Defaults to 14096.
__database__ (str, optional): database to connect. Defaults to "Sun".

Returns:
None
File: c:\users\felipe\documents\repos\sophia2\pysophia2\db.py
Type: function
```

Figura 8: Documentación para la función login().

## 7.2 Notebook de ejemplo

De cara al usuario, la manera recomendada de utilizar pysophia2 es mediante un entorno de ejecución de Jupyter Notebook (Jupyter Notebook, 2021). Como parte de este trabajo de tesis se ha elaborado un código de ejemplo (Sophia2, 2021) con todas las funciones disponibles para el usuario. Este notebook cuenta con un índice interactivo que permite desplazarse dentro del código a cualquier función simplemente presionando sobre ella (Figura 9).

## Date

date format YYYY/MM/DD

example: 2020-03-29

## INDEX

### 1. DATA

#### A. [INFO](#)

- [login\(\\_\\_password\\_\\_, \\_\\_username\\_\\_\)](#)
- [list\\_countries\(\)](#)
- [list\\_media\\_outlets\(\)](#)
- [stats\\_countries\(\)](#)
- [stats\\_countries\\_by\\_date\(from, to\)](#)
- [stats\\_media\\_outlet\(\)](#)
- [stats\\_media\\_outlet\\_by\\_country\(country\)](#)

#### B. [DATASETS](#)

- [get\\_dataset\(\)](#)

### 1. NGRAM

- [ngram.calculateDist\(dataset, group\\_by\)](#)
- [ngram.view\(word\\_distribution, keywords, annotate\)](#)
- [ngram.freq\(word\\_distribution, top\\_n\)](#)
- [ngram.freq\(word\\_distribution, words\)](#)

### 1. SOURCES

- [popularity\(sources, from, to\)](#)
- [list\\_sources\(country, from, to\)](#)
- [mentions\(sources, from, to\)](#)
- [gender\(country, from, to\)](#)
- [top\\_mentions\(country, top\\_n, from, to\)](#)

Figura 9: Índice del Jupyter Notebook de ejemplo.

A continuación, se adjuntan imágenes del notebook de ejemplo que permiten ver el funcionamiento real de la librería para cada una de las funciones, sus respectivos parámetros y retornos.

### 7.2.1 Construcción de un dataset de noticias a partir de determinados parámetros

Desde la Figura 10 a la Figura 19 se puede visualizar un ejemplo de aplicación para cada uno de los diferentes métodos que dan solución a la historia de usuario 1 con sus respectivas salidas.

```
login(__password__,__username__)  
login into database  
back to index  
  
In [5]: from pysophia2 import db  
  
In [6]: %%time  
        db.login(__password__="fsreaduser",__username__="sophia2api")  
        Wall time: 617 ms  
  
Out[6]: 'successful connection'
```

Figura 10: Ejemplo de uso de la función login.

```
list_countries()  
back to index  
  
In [12]: %%time  
         db.list_countries()  
         Wall time: 297 ms  
  
Out[12]: ['argentina',  
          'australia',  
          'brasil',  
          'canada',  
          'chile',  
          'china',  
          'colombia',  
          'costarica',  
          'cuba',  
          'ecuador',  
          'elsalvador',  
          'espana',  
          'estadosunidos',  
          'francia',  
          'guatemala',  
          'honduras',  
          'irlanda',  
          'islascock'
```

Figura 11: Ejemplo de uso de la función list\_countries.



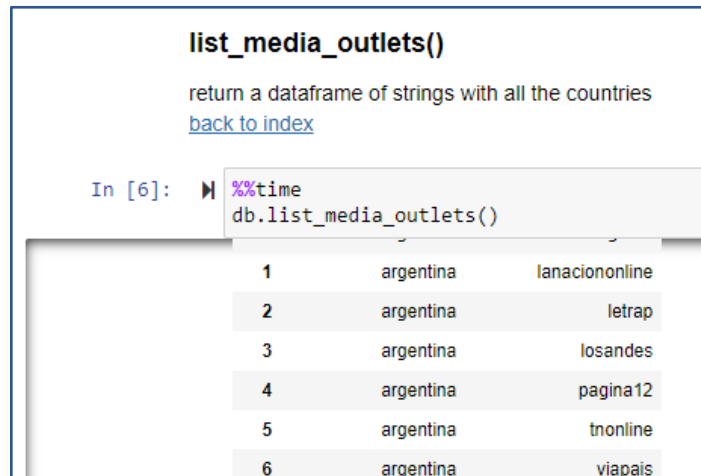


Figura 12: Ejemplo de uso de la función `list_media_outlets`.



Figura 13: Ejemplo de uso de la función `list_media_outlets`.

## stats\_countries()

List how many news are in caleuche countries

return a list of strings with all the countries

[back to index](#)

```
In [9]: %%time  
db.stats_countries()
```

Wall time: 314 ms

Out[9]:

|    | country     | quantity |
|----|-------------|----------|
| 0  | argentina   | 795753   |
| 1  | australia   | 541364   |
| 2  | brasil      | 1095610  |
| 3  | canada      | 1415989  |
| 4  | chile       | 1098397  |
| 5  | china       | 659437   |
| 6  | colombia    | 190707   |
| 7  | costarica   | 419698   |
| 8  | cuba        | 173541   |
| 9  | ecuador     | 8051     |
| 10 | el salvador | 126144   |

Figura 14: Ejemplo de uso de la función stats\_countries.

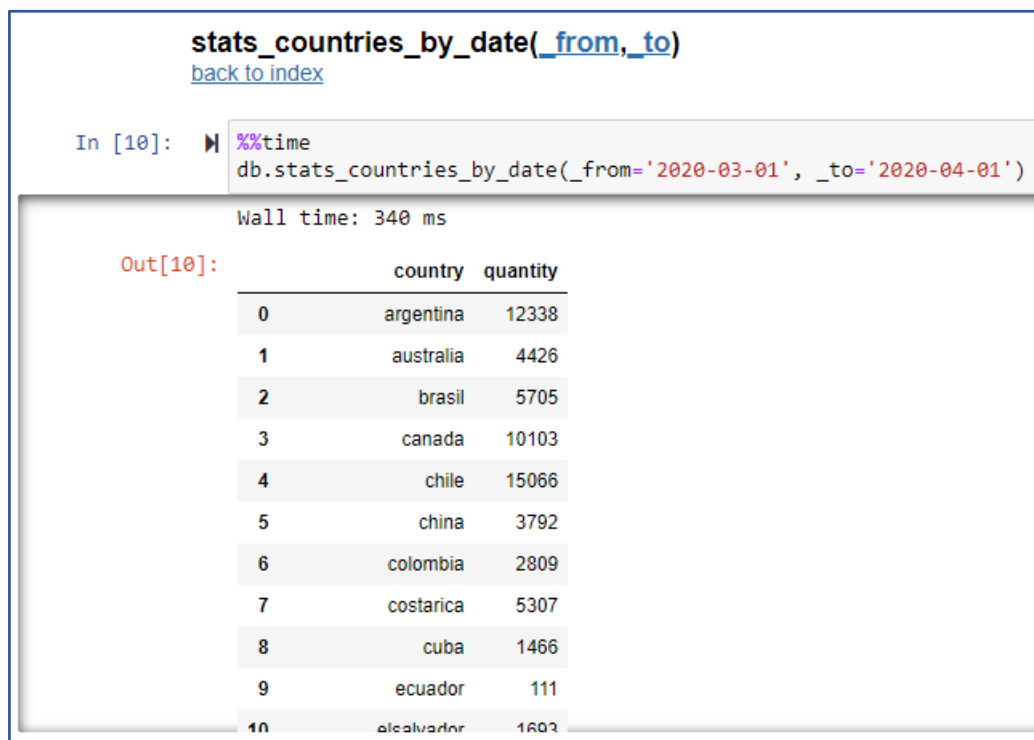


Figura 15: Ejemplo de uso de la función stats\_countries\_by\_date.

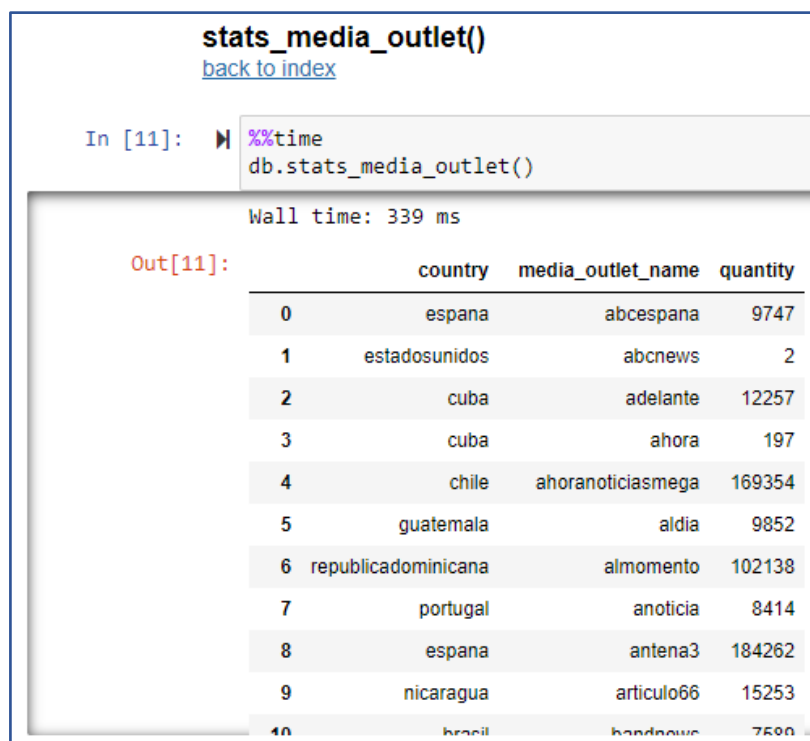


Figura 16: Ejemplo de uso de la función stats\_media\_outlets.

```
In [12]: db.stats_media_outlet("chile")
```

Out[12]:

|   | country | media_outlet_name | quantity |
|---|---------|-------------------|----------|
| 0 | chile   | ahoranoticiasmega | 169354   |
| 1 | chile   | biobiochile       | 122126   |
| 2 | chile   | elciudadano       | 81609    |
| 3 | chile   | elmostrador       | 18901    |
| 4 | chile   | emol              | 334527   |
| 5 | chile   | horas24           | 264725   |
| 6 | chile   | latercera         | 107155   |

Figura 17: Ejemplo de uso de la función stats\_media\_outlet.

**get\_dataset(keywords, country, \_from, \_to)**  
[back to index](#)

Para obtener las noticias para un determinado país en un periodo de tiempo es necesario no pasar el parametro keywords o utilizar un string vacío `keywords=""`

```
In [3]: %%time
df=db.get_dataset(country="chile", _from="2018-01-01",
                 _to="2019-01-01")
```

Se encontraron mas de 10000 noticias, por favor acotar la fecha de busqueda  
Wall time: 1min 11s

Para obtener todas las noticias que contengan las palabras biodiversidad y plantas en la misma noticia utilizar el parametro `keywords_operator="and"`

```
In [4]: %%time
df=db.get_dataset(keywords="biodiversidad plantas",
                 country="chile", _from="2018-01-01",
                 _to="2019-01-01", keywords_operator="and",
                 phrase=False)
```

Son 58 noticias encontradas...  
Wall time: 938 ms

Figura 18: Ejemplo de uso de la función get\_dataset sin keywords y con listado de keywords y operador and.

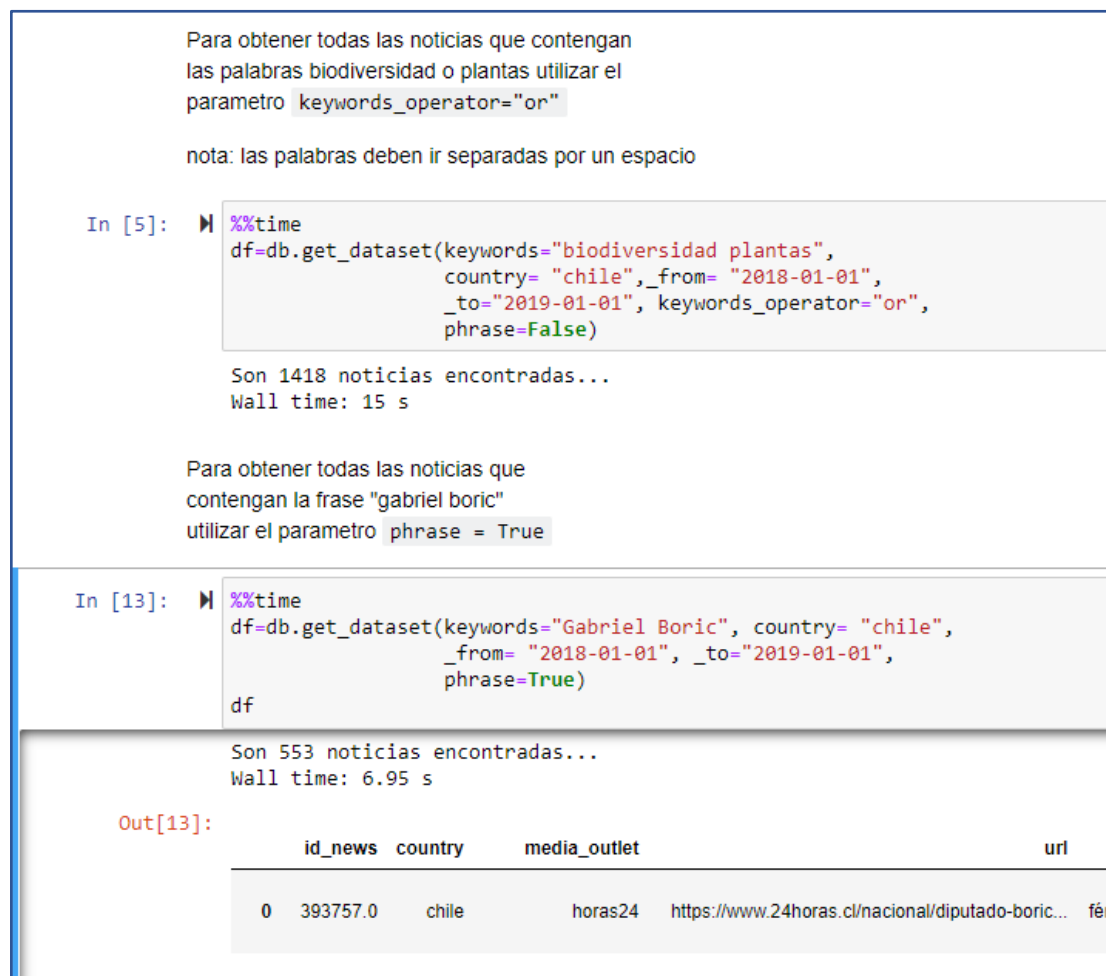
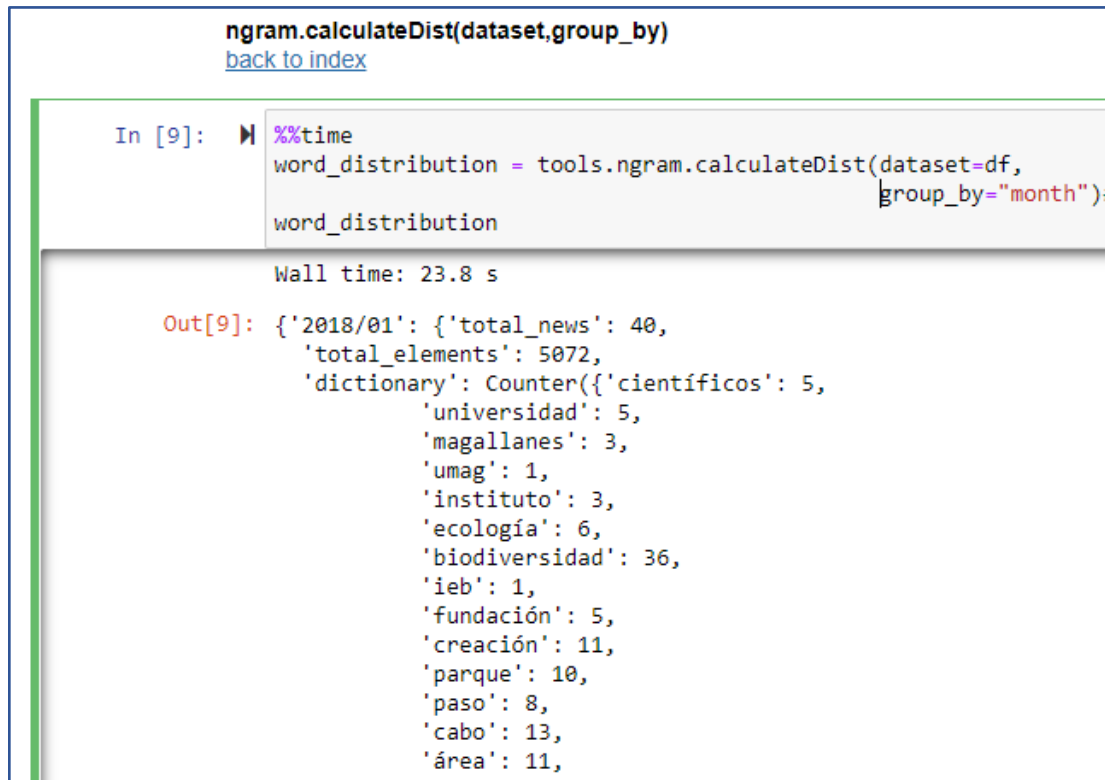


Figura 19: Ejemplo de uso de la función `get_dataset` para listado de keywords con operador `or` y por frase completa

### 7.2.2 Visualizar la evolución de la frecuencia de uso de palabras

Desde la Figura 20 a la Figura 23 se puede visualizar un ejemplo de aplicación para cada uno de los diferentes métodos que dan solución a la historia de usuario 2 con sus respectivas salidas.



The screenshot shows a Jupyter Notebook interface. At the top, there is a header with the text `ngram.calculateDist(dataset,group_by)` and a link [back to index](#). Below this, the input cell (In [9]) contains the following code:

```
In [9]: %%time
word_distribution = tools.ngram.calculateDist(dataset=df,
                                             group_by="month")
word_distribution
```

The output of the cell (Out[9]) shows the wall time and the resulting dictionary:

```
Wall time: 23.8 s
Out[9]: {'2018/01': {'total_news': 40,
                    'total_elements': 5072,
                    'dictionary': Counter({'científicos': 5,
                                           'universidad': 5,
                                           'magallanes': 3,
                                           'umag': 1,
                                           'instituto': 3,
                                           'ecología': 6,
                                           'biodiversidad': 36,
                                           'ieb': 1,
                                           'fundación': 5,
                                           'creación': 11,
                                           'parque': 10,
                                           'paso': 8,
                                           'cabo': 13,
                                           'área': 11,
```

Figura 20: Ejemplo de uso de la función `calculateDist`.

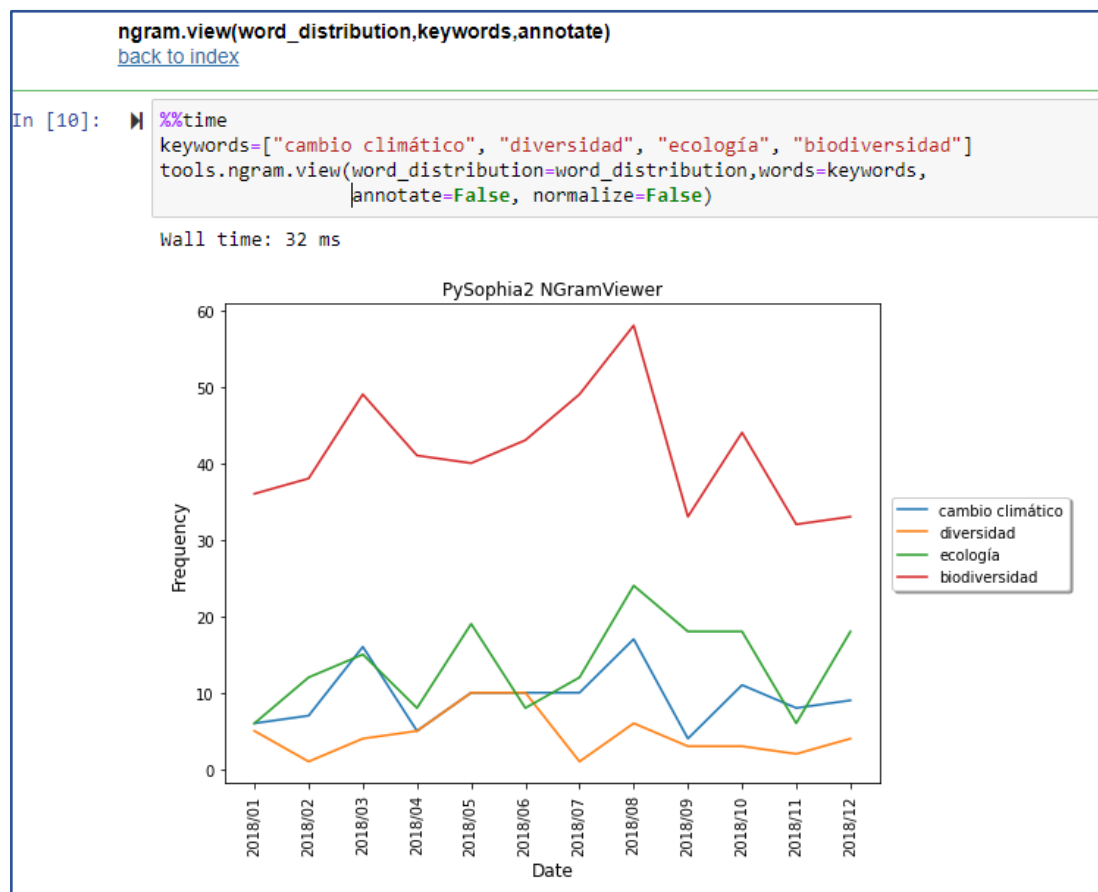


Figura 21: Ejemplo de uso de la función view generando un PySophia2NgramViewer.

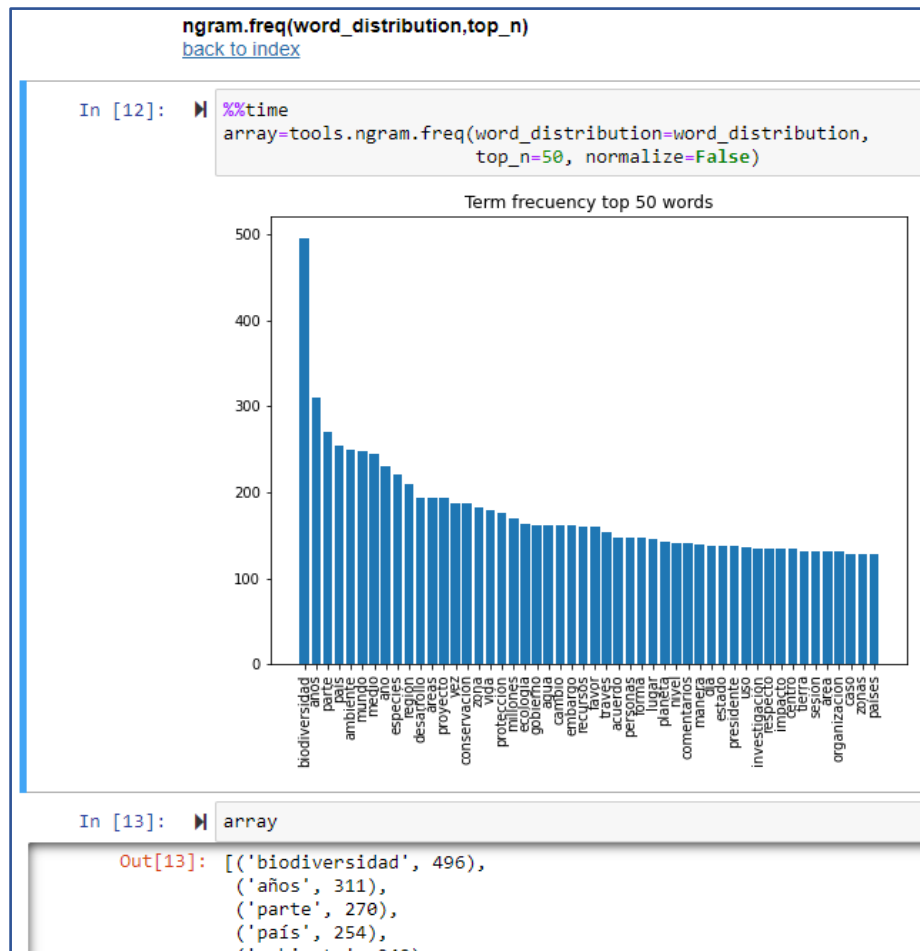


Figura 22: Ejemplo de uso de la función `freq` para generar un gráfico con las primeras 50 palabras con mayor frecuencia.



`ngram.freq(word_distribution, words)`  
[back to index](#)

```
In [24]: %%time  
array=tools.ngram.freq(word_distribution, words=["biodiversidad", "clima"],  
                        normalize=True)
```

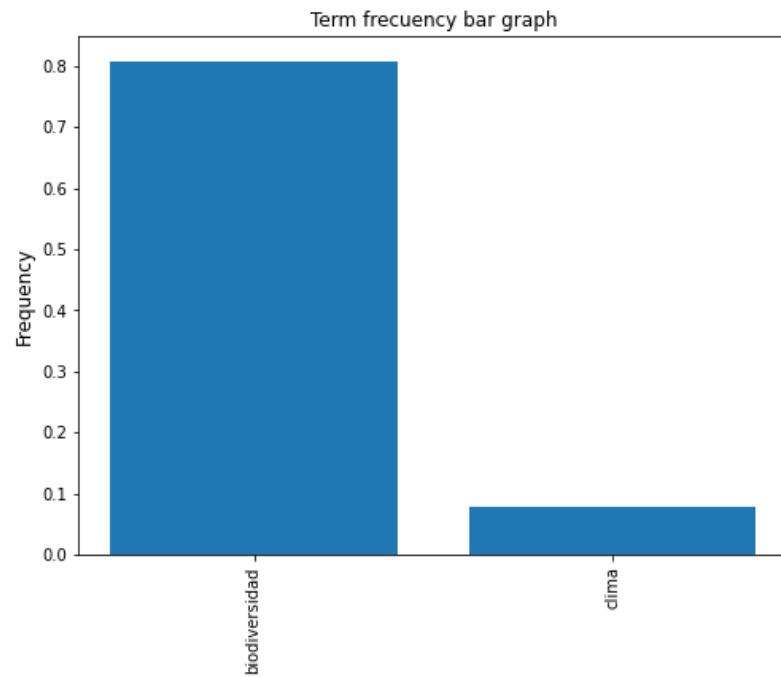


Figura 23: Ejemplo de uso de la función `freq` para graficar la frecuencia de una lista de palabras.

### 7.2.3 Extraer métricas sobre las fuentes de información y periodistas

Desde la Figura 24 a la Figura 28 se puede visualizar un ejemplo de aplicación para cada uno de los diferentes métodos que dan solución a la historia de usuario 3 con sus respectivas salidas.

| popularity(sources,_from,_to)  |               |               |               |               |               |               |       |      |
|--|---------------|---------------|---------------|---------------|---------------|---------------|-------|------|
| <a href="#">back to index</a>  |               |               |               |               |               |               |       |      |
| In [34]: <code>%%time</code><br><code>db.popularity(sources=['Gabriel Boric','José Antonio Kast'],</code><br><code>          [from="2020-01-01",_to="2021-12-31"]) ## por defecto por mes</code> |               |               |               |               |               |               |       |      |
| Out[34]:   |               |               |               |               |               |               |       |      |
|  | source        | popularity_en | popularity_es | popularity_fr | popularity_it | popularity_pt | month | year |
| 0  | Gabriel Boric | 506           | 4269          | 0             | 0             | 0             | 1     | 2020 |
| 1  | Gabriel Boric | 164           | 923           | 0             | 0             | 0             | 2     | 2020 |
| 2  | Gabriel Boric | 203           | 1326          | 0             | 0             | 0             | 3     | 2020 |
| 3  | Gabriel Boric | 132           | 1304          | 0             | 0             | 0             | 4     | 2020 |
| 4  | Gabriel Boric | 333           | 3134          | 0             | 0             | 0             | 5     | 2020 |
| 5  | Gabriel Boric | 225           | 2084          | 0             | 0             | 0             | 6     | 2020 |
| 6  | Gabriel Boric | 379           | 3761          | 0             | 0             | 0             | 7     | 2020 |
| 7  | Gabriel Boric | 236           | 2034          | 0             | 0             | 0             | 8     | 2020 |
| 8  | Gabriel Boric | 241           | 1696          | 0             | 0             | 0             | 9     | 2020 |
| 9  | Gabriel Boric | 412           | 3065          | 0             | 0             | 0             | 10    | 2020 |
| 10   | Gabriel Boric | 299           | 2304          | 0             | 0             | 0             | 11    | 2020 |
| 11   | Gabriel Boric | 232           | 1630          | 0             | 0             | 0             | 12    | 2020 |

Figura 24: Ejemplo de uso de la función *popularity* para una lista de personajes en un periodo.

```

list_sources(country,_from,_to)
back to index

In [35]:  %%time
          db.list_sources(country="chile",
                        from="2020-01-01",
                        [to="2021-12-31"])

Out[35]: [('Gary Hunziker',),
          ('Mollie Maxine Fitzgerald',),
          ('Jimmi Sham',),
          ('Papa Francisco',),
          ('Scott Morrison',),
          ('Shane Fitzsimmons',),
          ('Andrew Crisp',),
          ('Gary Hinton',),
          ('Ellen DeGeneres',),
          ('Tom Hanks',),
          ('Danna Paola',),
          ('Carol Burnett',),
          ('Ludwika Paleta',),
          ('Leonardo Farkas',),
          ('Mark Zuckerberg',)]

```

Figura 25: Ejemplo de uso de la función `list_sources`.

```

mentions(sources,_from,_to)
back to index

In [36]:  %%time
          db.mentions(sources=['Gabriel Boric','José Antonio Kast'],
                    from="2020-01-01",
                    [to="2021-12-31"])

Out[36]:

```

|   | source        | id_news | country | media_outlet |   |
|---|---------------|---------|---------|--------------|---|
| 0 | Gabriel Boric | 23996   | chile   | horas24      | <a href="https://www.horas24.cl">https://www.horas24.cl</a>         |
| 1 | Gabriel Boric | 26176   | chile   | biobiochile  | <a href="https://www.biobiochile.cl">https://www.biobiochile.cl</a> |
| 2 | Gabriel Boric | 31432   | chile   | biobiochile  | <a href="https://www.biobiochile.cl">https://www.biobiochile.cl</a> |
| 3 | Gabriel Boric | 34545   | chile   | biobiochile  | <a href="https://www.biobiochile.cl">https://www.biobiochile.cl</a> |
| 4 | Gabriel Boric | 42223   | chile   | biobiochile  | <a href="https://www.biobiochile.cl">https://www.biobiochile.cl</a> |

Figura 26: Ejemplo de uso de la función `mentions`.

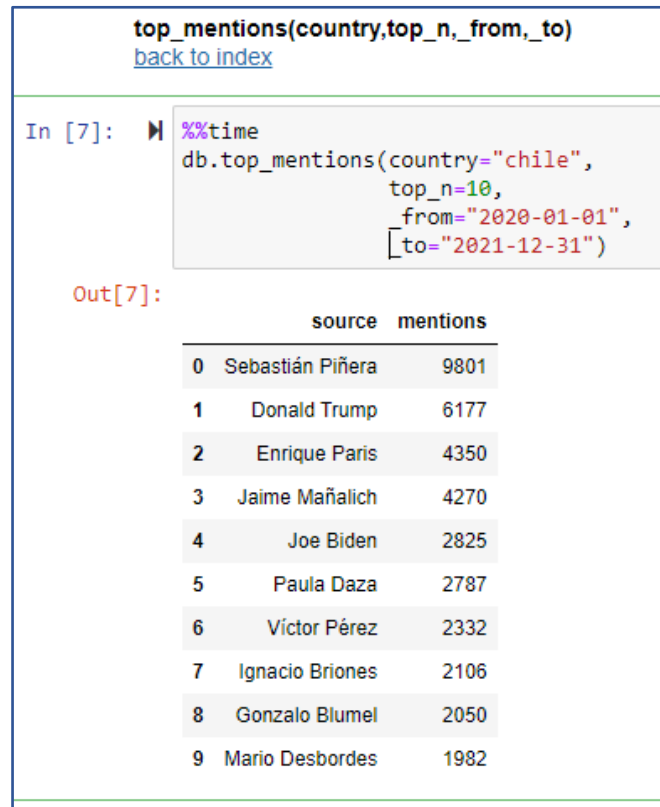


Figura 27: Ejemplo de uso de la función `top_mentions` para obtener el top 10 de personajes más mencionados en noticias entre el año 2020 y 2021.

|  |        |       |       |      |
|--|--------|-------|-------|------|
| <b>gender(country,_from,_to)</b><br><a href="#">back to index</a>  |        |       |       |      |
| In [6]: <code>%%time</code><br><code>db.gender(country='Chile',</code><br><code>          from="2020-01-01",</code><br><code>          to="2021-12-31")</code> |        |       |       |      |
| Out[6]:  |        |       |       |      |
|  | gender | news  | month | year |
| 0  | M      | 34508 | 1     | 2020 |
| 1  | M      | 23697 | 1     | 2021 |
| 2  | F      | 25627 | 2     | 2020 |
| 3  | F      | 3788  | 2     | 2021 |
| 4  | M      | 27222 | 3     | 2020 |
| 5  | M      | 5291  | 3     | 2021 |
| 6  | M      | 26560 | 4     | 2020 |
| 7  | M      | 4091  | 4     | 2021 |
| 8  | M      | 27514 | 5     | 2020 |
| 9  | F      | 30867 | 6     | 2020 |
| 10   | M      | 35967 | 7     | 2020 |

Figura 28: Ejemplo de uso de la función gender para chile en un periodo.

## **8. CONCLUSIONES Y PERSPECTIVAS**

Una vez finalizado este proyecto de título es posible apreciar de mejor manera el valor de abstraer de conocimiento técnico para realizar labores que generalmente requieren un grado de expertiz elevado en informática en conceptos como: conexiones a bases de datos, consultas SQL, manejo de DataFrame, generación de gráficos, librerías avanzadas de Python. Si se oculta de cara al usuario todas las configuraciones y tareas no relevantes para su propósito, es posible generar en una cantidad reducida de líneas código, análisis de gran valor para el estudio de los medios de prensa

### **8.1 La librería pysophia2 cumple con los requisitos funcionales identificados al inicio del proyecto**

Los objetivos, tanto general, como particulares, se cumplieron en su totalidad al igual que las 3 historias de usuario definidas en un comienzo.

El aporte de pysophia2 permite sentar bases sólidas y una estructura para extender las funcionalidades de la librería brindando la posibilidad de incluir futuros trabajos del grupo Sophia2 y acercarlos a la comunidad de investigación de ciencias sociales de una manera simple de utilizar.

### **8.2 La librería pysophia2 cumple con los requisitos no funcionales identificados al inicio del proyecto**

Los requisitos de calidad para el software definidos en un inicio de cumplieron totalmente. El requerimiento de portabilidad y despleabilidad fueron abordados de manera indirecta al utilizar Python en la construcción de la librería y esta encontrarse alojada en PyPI permitiendo que pueda ser instalada en cualquier sistema operativo que corra Python y cuente con pip instalado en su sistema.

El tema de la usabilidad fue abordado manteniendo los nombres de las funciones simples y sin conceptos abstractos además de hacer disponible el Jupyter Notebook de ejemplo con todas las funciones disponibles para el usuario.

Aparte de los requisitos definidos en un inicio, también fueron abordados otros requisitos no funcionales esenciales como por ejemplo la mantenibilidad del software al mantener la lógica de las funciones simples y que estas agrupen máximo una lógica, además de estar debidamente documentadas en el código, detalles que serán de utilidad para eventuales futuros desarrolladores.

### **8.3 ¡La librería pysophia2 ya se está utilizando!**

Al final del proyecto, son dos grupos de usuarios que ya empezaron a utilizar la librería pysophia2:

- Investigadores en comunicación: proyecto ANID Fake News
- Informáticos trabajando en el equipo Sophia2, proyecto GORE Los Lagos, lanzamiento de la oferta “Informes mensuales MiRegión / MiCiudad”

## **8.4 Recomendaciones y perspectivas sobre la mantención y la evolución de pysophia2**

Por el momento la versión actual de pysophia2 se encuentra en la numero 0.1.3 ya que se han agregado funcionalidades y realizado correcciones de bugs desde su versión inicial 0.0.1 y se espera se sigan agregando mas en un futuro. Para esto se recomienda mantener las directrices establecidas en este trabajo, como por ejemplo aumentar el número de la versión de acuerdo con el versionamiento semántico, mantener la lógica de las funciones simples y enfocadas en una tarea, en vez de crear funciones de mayor tamaño que cueste mas digerir por los nuevos desarrolladores

## 9. REFERENCIAS

- Bienvenu, J. (17 de Dic de 2020). *Deep dive: Create and publish your first Python library*. Recuperado el 25 de 06 de 2021, de <https://towardsdatascience.com/deep-dive-create-and-publish-your-first-python-library-f7f618719e14>
- Blei, & Lafferty. (25 de June de 2006). Dynamic Topic Models. *ACM*. doi:1143844.1143859
- Brownlee, J. (11 de October de 2017). *Machine Learning Mastery: What Are Word Embeddings for Text?* Obtenido de <https://machinelearningmastery.com/what-are-word-embeddings/>
- DataFrame. (2021). *DataFrame-Pandas*. Obtenido de <https://pandas.pydata.org/pandas-docs/stable/reference/frame.html>
- Doc. (2021). *Doc-Spacy*. Obtenido de <https://spacy.io/api/doc>
- ElasticSearch. (2021). *ElasticSearch*. Obtenido de <https://elasticsearch-py.readthedocs.io/en/v7.16.2/>
- es\_core\_news\_md. (2021). *es\_core\_news\_md* Spacy. Obtenido de <https://spacy.io/models/es>
- Google. (2010). *Google Ngram Viewer*. Recuperado el 25 de 07 de 2021, de <https://books.google.com/ngrams>
- groupby. (2021). *groupby Pandas*. Obtenido de <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html>
- Help Net Security. (2021). *HelpNetSecurity: 39% of all internet traffic is from bad bots*. Obtenido de <https://www.helpnetsecurity.com/2021/09/07/bad-bots-internet-traffic/>
- ISO. (1991). *ISO/IEC 9126*. Obtenido de <https://www.iso.org/standard/16722.html>
- ISO. (2011). *ISO/IEC 25010*. Obtenido de <https://www.iso.org/standard/35733.html>
- ISO. (2018). *IEC 9241-11*. Obtenido de <https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en>
- Jupyter Notebook. (2021). *Jupyter Notebook*. Obtenido de <https://jupyter.org/>
- MariaDB. (2021). *MariaDB*. Obtenido de <https://mariadb.com/resources/blog/how-to-connect-python-programs-to-mariadb/>
- Matcher. (2021). *Matcher Spacy*. Obtenido de <https://spacy.io/api/matcher>
- Matplotlib. (2021). *Matplotlib*. Obtenido de <https://matplotlib.org/>
- Pandas. (12 de Dec de 2021). *Pandas*. Obtenido de <https://pandas.pydata.org/>
- PEP427. (20 de Sep de 2012). *PEP427*. Recuperado el 25 de 06 de 2021, de <https://www.python.org/dev/peps/pep-0427/>
- PEP8. (05 de Jul de 2001). *PEP8 Style Guide for Python Code*. Obtenido de <https://www.python.org/dev/peps/pep-0008/>
- PUBLICA, M. D. (03 de NOV de 2017). *LEY DE PROPIEDAD INTELECTUAL N°17336*. Obtenido de <https://www.bcn.cl/leychile/navegar?idLey=17336>
- PyPI. (2021). *Python Package Index*. Obtenido de <https://pypi.org/>
- Python. (2021). *Documentation Strings*.



Sophia2. (2021). *Jupyter Notebook Ejemplo*. Obtenido de [https://github.com/TeamSophia2/pySophia2/blob/main/Example\\_notebook.ipynb](https://github.com/TeamSophia2/pySophia2/blob/main/Example_notebook.ipynb)

Sophia2. (2021). *pysophia2 Pypi*. Obtenido de <https://pypi.org/project/pysophia2/>

Sophia2. (2021). *Sophia2 Github*. Obtenido de <https://github.com/TeamSophia2/pySophia2>

Spacy. (2021). *Spacy*. Obtenido de <https://spacy.io/usage/spacy-101>

test Pypi. (s.f.). *Pypi test respository*. Recuperado el 15 de 05 de 2121, de <https://test.pypi.org/>

Token. (2021). *Token Spacy*. Obtenido de <https://spacy.io/api/token>

Wouter Van, A., Trilling, D., & Arcila, C. (2021). *Computational Analysis of Communication*.