



Sistema para la detección de efectos adversos a medicamentos en textos biomédicos en español

Proyecto de Fin de Grado en ETS de Ingeniería Informática de modalidad específica

Realizado por: Roi Arias Rico

Dirigido por: Raquel Martínez Unanue

Codirigido por: María del Soto Montalvo Herranz

Curso académico: 2021/2022 Convocatoria de defensa:



Sistema para la detección de efectos adversos a medicamentos en textos biomédicos en español

Proyecto de Fin de Grado en ETS de Ingeniería Informática de modalidad específica

Realizado por: Roi Arias Rico

Dirigido por: Raquel Martínez Unanue

Codirigido por: María del Soto Montalvo Herranz

Fecha de lectura y defensa del proyecto:.....

Resumen del proyecto

Los efectos adversos de los medicamentos constituyen un problema de salud muy importante tanto a nivel económico como sanitario. A pesar de que existen instituciones dedicadas a recoger las posibles sospechas de efectos adversos a los medicamentos, los casos detectados suelen ser inferiores a los reales. Por ello, una de las posibilidades de detectar efectos adversos está relacionada con las redes sociales debido a su amplio uso y disponibilidad de datos. En este proyecto se trata de recopilar los mensajes de los usuarios en una red social y extraer los efectos adversos de los medicamentos mencionados. Debido a los avances en aprendizaje automático, hoy en día se pueden emplear diferentes algoritmos que permitan automatizar y clasificar la existencia de este tipo de relaciones. A pesar de los avances, no es un proceso fácil ya que se trata de documentos desestructurados. Las dificultades de este tipo de procesos se basan en que los textos no tienen una gramática clara, con abundantes errores de ortografía o presencia de símbolos como *emojis* que dificultan la tarea de identificación. Por eso, en este tipo de documentos se inicia con un proceso de limpieza de textos para evitar los errores de identificación. A menudo, es acompañado de etapas de preprocesamiento como tokenización, lematización, etiquetado PoS o *stemming*. Una vez que disponemos de un texto sin tantos errores, se procede al reconocimiento de entidades, ya sea mediante algoritmos de aprendizaje automático o mediante la creación de reglas o patrones de búsqueda. Este trabajo se apoya en el empleo de diccionario elaborados *ad hoc* para este trabajo y del diccionario UMLS para términos médicos. Una vez identificados los términos, se inicia la clasificación que permite detectar la presencia o ausencia de efectos adversos.

Lista de palabras clave

Efectos adversos de medicamentos; procesamiento de lenguaje natural; minería de datos; aprendizaje automático.

Abstract

System for the detection of adverse drug effects in biomedical texts in Spanish

The adverse effects of medications are a very important health problem both economically and health-wise. Despite the fact that there are institutions dedicated to collecting possible suspicions of adverse drug effects, the cases detected are usually lower than the real ones. Therefore, one of the possibilities of detecting adverse effects is related to social networks due to their extensive use and availability of data. This project tries to collect the messages of the users in a social network and extract the adverse effects of the mentioned medicines. Due to advances in machine learning, today different algorithms can be used to automate and classify the existence of this type of relationship. Despite the advances, it is not an easy process since it deals with unstructured documents. The difficulties of this type of process are based on the fact that the texts do not have a clear grammar, with abundant spelling errors or the presence of symbols such as emojis that make the identification task difficult. For this reason, this type of document begins with a text cleaning process to avoid identification errors. It is often accompanied by pre-processing steps such as tokenization, lemmatization, PoS tagging, or stemming. Once we have a text without so many errors, we proceed to the named-entity recognition, either through machine learning algorithms or through the creation of rules or search patterns. This work is supported by the use of dictionaries prepared ad hoc for this work and the UMLS dictionary for medical terms. Once the terms have been identified, the classification begins, which allows detecting the presence or absence of adverse effects.

Keywords

Adverse effects; natural language processing; data mining; machine learning.

Índice

1. Introducción	1
1.1 Descripción del problema	1
1.2 Motivación	3
1.3 Objetivos	4
1.4 Estructura del trabajo.....	5
2. Antecedentes	7
2.1 Extracción de la información aplicada al dominio biomédico	7
2.2 Revisión de las fuentes de información biomédicas	8
2.3 Enfoques empleados en NER	9
2.3.1 Enfoque basado en conocimiento.....	9
2.3.2 Enfoque basado en aprendizaje automático	10
2.4 Enfoques empleados para la extracción de características	12
2.4.1 BoW	13
2.4.2 Modelos distribucionales	13
2.4.3 Modelos de lenguaje	15
2.5 Enfoques empleados para la clasificación.....	16
2.6 Elección de los enfoques empleados en este trabajo y justificación.....	16
3. Planteamiento del problema	19
3.1 Construcción de diccionario de medicamentos	19
3.1.1 CIMA.....	19
3.1.2 Vademecum	21
3.2 Construcción de diccionario de términos médicos	23
3.2.1 MedDRA- (español)	23
3.2.2. Diccionario de términos médicos de la Real Academia Nacional de Medicina	25
3.3 Minado de mensajes en Twitter	26
3.4 Extracción de conceptos de documentos no estructurados.....	31
3.5 Extracción de características (<i>feature extraction</i>)	32
3.6 Clasificación y evaluación.....	35
4. Descripción del sistema desarrollado.....	39
4.1 Esquema de la solución.....	39
4.2 Limpieza de texto	39
4.3 Preprocesamiento	43

4.3.1 Tokenización.....	43
4.3.2 Lematización/Stemming.....	44
4.3.3 Stopwords	46
4.4 Reconocimiento de entidades (NER).....	46
4.5 Extracción de características (<i>feature extraction</i>) en el sistema.....	49
4.6 Clasificación de efectos adversos.....	50
4.6.1 Algoritmos considerados para la clasificación	51
4.7. Desarrollo de una aplicación.....	51
5. Metodología de desarrollo y diseño	53
5.1 Metodología	53
5.2 Herramientas utilizadas	53
5.3 Planificación	54
5.4 Requisitos del sistema.....	56
5.4.1 Requisitos funcionales.....	56
5.4.2 Requisitos no funcionales	57
5.5 Casos de uso.....	57
5.6 Presupuesto y cálculo de costes.....	59
5.6.1 Descripción del proyecto.....	59
5.6.2 Cálculo de costes.....	60
5.6.3 Presupuesto.....	62
6. Evaluación del sistema	63
6.1 Métricas para la extracción de información.....	63
6.2 Resultados	65
6.2.1 <i>Corpus</i> de prueba	65
6.2.2 Limpieza.....	65
6.2.2 Reconocimiento de entidades.....	66
6.2.3 Clasificación de textos	67
7. Conclusiones.....	75
7.1 Conclusiones.....	75
7.2 Posibles mejoras y líneas futuras de desarrollo	76
Bibliografía	79
Listado de siglas, abreviaturas y acrónimos.....	81
Anexo	83
Anexo A Guía de instalación.....	83

Anexo B Manual de usuario	85
---------------------------------	----

Indice de figuras

Figura 1. Representación gráfica de los modelos CBOW y Skip-gram	14
Figura 2. Ejemplo de entrada del archivo DICCIONARIO_ATC.xml	20
Figura 3. Ejemplo de entrada del archivo DICCIONARIO_PRINCIPIOS_ACTIVOS.xml	20
Figura 4. Ejemplo de estructura en CIMA de nombres comerciales de medicamentos	21
Figura 5. Distribución de menciones de marcas comerciales agrupadas por ATC.....	28
Figura 6. Distribución de menciones de principios activos agrupadas por ATC	30
Figura 7. Ejemplos de mensajes extraídos de la red social Twitter	31
Figura 8. Pipeline de detección de efectos adversos	38
Figura 9. Pipeline del sistema para la detección de efectos adversos	39
Figura 10. Ejemplos de preprocesamiento de tweets con eliminación del usuario	41
Figura 11. Ejemplo de tokenización	43
Figura 12. Componentes de la librería SpaCy (Fuente: SpaCy.io).....	44
Figura 13. Ejemplo de reconocimiento de entidades empleando stemming	46
Figura 14. Definición de la entidad "dosis"	47
Figura 15. Pantalla con los resultados finales del sistema	52
Figura 16. Diagrama de Gantt del proyecto	55
Figura 17. Matriz de confusión.....	63
Figura 18. Resultados de la limpieza del corpus	65
Figura 19. Extracto de un texto tras NER	66
Figura 20. Resultados del reconocimienro de entidades.....	66
Figura 21. Número de efectos adversos detectados en los tweets.	67
Figura 22. Matriz de confusión obtenida en la regresión logística	71
Figura 23. Curva ROC resultado de la Regresión Logística	72

Indice de tablas

Tabla 1. Ventajas e inconvenientes de enfoque basados en conocimiento y en aprendizaje automático	12
Tabla 2. Distribución de tipos semántica de MedDRA.....	25
Tabla 3. Clasificación de marcas comerciales según apariciones en la red social	27
Tabla 4. Clasificación de principios activos según apariciones en la red social	29
Tabla 5. Ejemplo de BoW	33
Tabla 6. Algoritmo estadístico de aprendizaje	35
Tabla 7. Ejemplos de preprocesamiento de emojis	42
Tabla 8. Distribución del esfuerzo.....	55
Tabla 9. Caso de uso.....	57
Tabla 10. Flujo básico del caso de uso	58
Tabla 11. Flujo alternativo 1.....	59
Tabla 12. Flujo alternativo 2.....	59
Tabla 13. Costes de personal	60
Tabla 14. Costes de equipamiento.....	60
Tabla 15. Costes de software	61
Tabla 16. Costes de material fungible y otros gastos.....	61
Tabla 17. Presupuesto del proyecto.....	62
Tabla 18. Resultados de la precisión de los algoritmos	68
Tabla 19. Resultados de los algoritmos.....	69
Tabla 20. Resultados de la validación cruzada.....	70
Tabla 21. Resultados de precisión, recall y f1 del modelo final	71
Tabla 22. Matriz de confusión con regresión logística (modelo final)	71
Tabla 23. Curvas ROC de algoritmos estudiados.....	72
Tabla 24. Comparativa de resultados TF-IDF vs combinada con modelos distribucionales	73

1. Introducción

1.1 Descripción del problema

De acuerdo con la Organización Mundial de la Salud (OMS), una reacción adversa al medicamento (RAM) se puede definir como una respuesta a un fármaco que es nociva y no intencionada y que tiene lugar cuando este se administra en dosis utilizadas normalmente en seres humanos para la profilaxis, diagnóstico o tratamiento de una enfermedad, o para la modificación de una función fisiológica. Como consecuencia de ello, se puede definir la Farmacovigilancia como la actividad de salud pública que tiene por objetivo la identificación, cuantificación, evaluación y prevención de los riesgos del uso de los medicamentos una vez comercializados [1].

Esta actividad recibe un elevado grado de atención tanto por las autoridades competentes en el ámbito de la autorización de comercialización de fármacos como de la propia industria farmacéutica [2].

En una situación ideal, todos los efectos adversos asociados a un fármaco son detectados antes de la comercialización del propio fármaco. Sin embargo, esto no siempre es posible porque el número de individuos en un ensayo clínico es pequeño comparado con toda la población lo que limita la capacidad de detectar efectos adversos, sobre todo si además son muy poco frecuentes. Otro motivo es la duración de los ensayos clínicos que evita la detección de reacciones adversas a largo plazo. Por último, se debe destacar que los ensayos clínicos muestran unas condiciones ideales de utilización de fármacos, mientras que las condiciones reales sólo se manifiestan tras la autorización de comercialización del medicamento, eso significa que las posibles interacciones con otros medicamentos no se pueden predecir en cuanto a la magnitud e implicaciones para la salud de los pacientes.

Las autoridades competentes en cada país o en cada región como *Federal Drugs Administration* (FDA) en Estados Unidos o la Agencia Española de Medicamentos y Productos Sanitarios (AEMPS) en España obligan a los profesionales sanitarios a

informar de las reacciones adversas siempre que atribuyan un problema de salud a la utilización de algún fármaco. Cada organismo nacional ha diseñado sistemas de notificación como MedWatch por la FDA o la tarjeta amarilla en el caso español con el objeto de facilitar la comunicación de estos efectos adversos. A pesar de ello, un gran inconveniente de estos sistemas de notificación es que se ha llegado a comprobar que informan menos RAM de las que realmente existen [3], incluso se ha llegado a estimar que las notificadas representan 2-10% del total [4-6].

El aumento en la disponibilidad de historiales clínicos electrónicos y la capacidad de procesar grandes volúmenes automáticamente gracias al Procesamiento de Lenguaje Natural (PLN) y algoritmos de aprendizaje automático han abierto nuevas oportunidades a la Farmacovigilancia [7]. Dentro de la Red, el gran desarrollo de las redes sociales como Twitter, Facebook, Instagram y Pinterest las han catapultado como importantes fuentes de datos en diferentes ámbitos. Sin embargo, las redes sociales no están exentas de inconvenientes para estas tareas por lo que los datos obtenidos de las redes sociales deben poder evaluarse según su credibilidad, frecuencia, novedad, unicidad o importancia. Estas cinco propiedades son fundamentales para seleccionar el tipo de red social. Por ejemplo, una red social como Twitter, al tener un límite bajo de caracteres en sus mensajes, no podría tener una alta relevancia (*salience data*). Además, algunos autores exponen que existe una pequeña proporción de mensajes que hagan referencia a los efectos adversos de los medicamentos [8], por lo que son necesarios grandes volúmenes de datos para la inclusión de suficientes mensajes que mencionen efectos adversos.

Algunos estudios han propugnado por la combinación tanto de los sistemas de información espontáneos como de los historiales clínicos digitales ya que aumenta la exactitud en la detección de efectos adversos [9]. En países de nuestro entorno se han establecido guías por parte de la *Association of the British Pharmaceutical Industry* (ABPI), estableciendo los datos mínimos de información necesarios para informar del efecto adverso, como que se pueda identificar al paciente, medicamento sospechoso, efectos adversos e informador identificable (correo electrónico, nombre de usuario).

1.2 Motivación

Estudios en otros países han demostrado que las RAM son responsables de ingresos hospitalarios, aumentos en la estancia hospitalaria y de la mortalidad [10]. Como complemento a los sistemas de información tradicionales, la explotación de datos en redes sociales podría contribuir a su detección temprana.

A pesar de esto, la detección de efectos adversos todavía se encuentra inmadura ya que se deben crear *lexicones* propios para crear un sistema de reconocimiento de entidades y suelen estar limitados a un grupo de fármacos objetos del estudio [11, 12]. Con el objetivo de extraer información y de detectar potenciales RAM se han empleado técnicas de PLN, no sólo en el minado de textos en redes sociales sino también en hojas de información de fármacos [13], informes médicos [14] e historiales clínicos [15].

El PLN emplea funciones estadísticas y algoritmos computacionales para analizar texto desestructurado y extraer información cuantitativa de él. Debido al pequeño tamaño de los textos de algunas redes sociales, los métodos tradicionales de PLN no son adecuados [16] por lo que se han desarrollado herramientas específicas para datos de las redes sociales [17].

Con el objetivo de auxiliar en la detección de los términos biomédicos presentes en un documento, se han empleado bases de datos que contienen miles de terminologías referidas a un campo específico o al campo global de la salud. Destacan entre las más utilizadas la *International Classification of Diseases* (ICD), *Systematized Nomenclature of Medicine – Clinical Terms* (SNOMED CT) y *Unified Medical Language System* (UMLS). De todas estas bases de datos, UMLS posee como gran ventaja que proporciona códigos estándar para miles de conceptos biomédicos e incluye los vocabularios de ICD y SNOMED.

Tal y como se ha comentado anteriormente, el PLN se ha empleado para extraer información de prospectos de medicamentos o de historiales clínicos pero cuando se ha utilizado, con ese mismo objetivo, en redes sociales ha necesitado de etapas adicionales de procesamiento. Esto se debe a las connotaciones particulares de los mensajes en redes sociales que se caracterizan por incluir un gran número de *hashtags*, abreviaturas

no convencionales, jergas, *emojis*, ausencia de signos de puntuación, errores ortográficos o gramaticales [18]. Además de estos problemas habituales de procesamiento de textos, existen otros relacionados con el ruido y desequilibrio ya que es habitual, en estos *corpus*, que la mención a posibles efectos adversos esté presente en menos del 10% de los tweets [19].

Por último, no es menos importante que el hecho de extraer información de estos medios podría representar un desafío a nivel técnico, político o de la protección de datos. A pesar de ello, la hipótesis de este trabajo es que las redes sociales pueden suponer un importante apoyo en las tareas de Farmacovigilancia para detectar efectos adversos desconocidos y, por lo tanto, incrementar la seguridad de los fármacos comercializados. Aunque existe una gran cantidad de estudios empleando como idioma de trabajo el inglés, no existe una gran cantidad de estudios que extraigan información del español.

De esta forma, este trabajo podría contribuir a aumentar el número de estudios que utilizan el idioma español para establecer el estado del arte en este campo y poder comparar los resultados obtenidos con otras técnicas incluidas en la bibliografía.

1.3 Objetivos

Este trabajo tiene como objetivo primordial el diseño e implementación de un sistema para la detección de efectos adversos a medicamentos en textos biomédicos. En este estudio y, debido al auge de las redes sociales, se ha seleccionado como textos biomédicos los mensajes disponibles en una red social como Twitter.

Los objetivos de este trabajo son:

- Creación de un *corpus* y de diccionarios formados por los principios activos/medicamentos autorizados en España y terminología médica en el idioma español.
- Recolección de datos de la red social Twitter.

- Implementación de un sistema para la detección de fármacos y efectos adversos a través de un enfoque basado en diccionario.
- Evaluación del sistema con el *corpus* previamente creado y comparación con otros sistemas según la bibliografía.

1.4 Estructura del trabajo

El trabajo se encuentra estructurado como sigue:

- Capítulo 1 “Introducción”: breve introducción al proyecto, motivaciones para su realización, continuando con los objetivos que se pretenden cubrir con el proyecto, así como la estructura del trabajo aquí expuesto.

- Capítulo 2 “Antecedentes”: descripción de trabajos realizados en el ámbito del análisis de las redes sociales para detectar reacciones adversas a medicamentos. Además, se presenta el análisis de las fuentes de términos en español tanto para fármacos como de efectos adversos, junto con los métodos empleados en la extracción de entidades.

- Capítulo 3 “Planteamiento del problema”: descripción de la base teórica sobre la que se construye la propuesta de solución de este trabajo, junto con las herramientas utilizadas para su implementación.

- Capítulo 4 “Descripción del sistema desarrollado”: propuesta de la solución al problema, descripción de recursos utilizados, alternativas valoradas en cada paso y las etapas o tareas realizadas.

- Capítulo 5 “Metodología de desarrollo y diseño”: se describen los métodos empleados a lo largo de este trabajo, así como su diseño.

- Capítulo 6 “Resultados”: exposición de los resultados y comparación con los resultados obtenidos de otros estudios incluidos en la bibliografía.

· Capítulo 7 “Conclusiones”: discusión de los resultados y resumen de las conclusiones, así como potenciales líneas de desarrollo.

2. Antecedentes

2.1 Extracción de la información aplicada al dominio biomédico

El ámbito principal de este proyecto es el PLN. Mediante el PLN se investiga la manera de comunicar las máquinas con las personas mediante el uso de lenguas naturales, como el español, y cuyo principal objetivo es automatizar este proceso de traducción. Como parte del PLN, este proyecto se enfoca en la extracción de información que consiste en extraer automáticamente información estructurada a partir de documentos no estructurados, en este caso mensajes de redes sociales. Los sistemas de extracción de entidades y ,más concretamente, el Reconocimiento de Entidades Nombradas (Named Entity Recognition, por sus siglas en inglés NER) realiza la tarea de buscar información muy concreta en colecciones de documentos, detectar información relevante, extraerla y etiquetarla en un formato adecuado para su procesamiento automático.

Una Entidad Nombrada es una palabra (o conjunto de palabras) que contiene un nombre, como puede ser en este trabajo la marca comercial de un medicamento o un nombre de una persona.

Las entidades de este proyecto pertenecen al dominio biomédico, aunque existen ciertas especificidades dentro de este dominio que añaden complejidad a la detección de efectos adversos a medicamentos:

- Los nombres comerciales y los nombres de los principios activos suelen diferir.
- Los nombres comerciales no son siempre los mismos a lo largo del tiempo, sino que pueden variar por razones de *marketing*, retirada comercial o novedades en el mercado.
- El empleo extenso de abreviaturas, junto con modificadores léxicos como el plural o la incorporación de palabras compuestas para reflejar el sistema de liberación o un incremento de dosis, por ejemplo: “retard”, “forte” o “flas”.

Este proyecto consistirá en aplicar técnicas de NER para reconocer fármacos y términos biomédicos en *tweets* de usuarios. Este reconocimiento permitirá, posteriormente, su clasificación como efecto adverso asociado a un fármaco o no.

2.2 Revisión de las fuentes de información biomédicas

Todas las fuentes de información empleadas para la creación de este proyecto se encuentran en la web de forma libre y de fácil acceso a todos los usuarios. Los términos médicos pertenecen a UMLS que es un conjunto de archivos y software que agrupa los vocabularios y estándares biomédicos y del campo de la salud para permitir la interoperabilidad entre sistemas informáticos. Este sistema ha sido desarrollado por el *National Institute of Health* (NIH) perteneciente al gobierno de Estados Unidos. A pesar de que existen otros vocabularios y bases de datos como *Medical Subject Headings* (MeSH) o *Logical Observation Identifier Names and Codes* (LOINC), UMLS es ya considerado como un estándar, que además es una herramienta con licencia gratuita.

El UMLS emplea los términos médicos relacionados con los síntomas siguiendo los conceptos incluidos en el vocabulario *Medical Dictionary for Regulatory Activities* (MedDRA). Este diccionario de términos médicos fue desarrollado por la *International Conference of Harmonisation* (ICH) y es propiedad de *International Federation of Pharmaceutical Manufacturers and Associations* (IFPMA). MedDRA hace especial hincapié en la entrada, recuperación, análisis y visualización de datos. Se aplica en todas las fases del desarrollo de fármacos excepto en la experimentación animal, además de los efectos sobre la salud y funcionamiento defectuoso de dispositivos médicos. Además, esta terminología tiene como ventaja que está traducida a múltiples idiomas, incluido el español (conocida como MDRSPA) . La última edición fue en Marzo 2022, si bien para este proyecto se ha empleado una versión anterior incluida en el metatesauro de UMLS (Noviembre 2021).

En cuanto a las fuentes de medicamentos suelen estar disponibles a través de las agencias sanitarias nacionales competentes. En el caso de España, se denomina Centro de Información de Medicamentos de la AEMPS (CIMA) aunque en ocasiones otros

autores han optado por la disponibilidad de información en otras webs no institucionales dedicadas a la información de medicamentos. Es habitual para la extracción de información y creación de un diccionario de términos utilizar la información disponible en la red mediante técnicas de *webscraping* [20]. El empleo de *webscraping* implica un ahorro de tiempo y costes al no depender de sistemas de pago y poder aprovecharse de recursos fácilmente accesibles como pueden ser los elementos contenidos en listas de una página HTML.

Un importante obstáculo es el idioma empleado, aunque ha habido multitud de estudios y trabajos destinados a explotar estas redes sociales, como foros especializados o Twitter hay una gran mayoría en inglés y en mucha menor medida en español [21, 22]. La primera autora que realizó un *corpus* de efectos adversos y de fármacos en español fue Segura-Bedmar et al [20], a través de la extracción de datos de un foro clínico en español. Mediante posteriores trabajos de esta misma autora, se construyó un diccionario de fármacos y sus efectos adversos utilizando la base de datos MedDRA, llegando hasta tener registrados 74865 efectos y 7593 fármacos [21].

2.3 Enfoques empleados en NER

Para la detección de efectos adversos a medicamentos en redes sociales existen dos enfoques fundamentales. Estos dos enfoques son los basados en reglas (donde incluimos diccionarios, patrones y reglas sintácticas u ortográficas) y aquellos basados en técnicas de aprendizaje automático. A continuación, se detallan en qué consisten cada uno de los enfoques.

2.3.1 Enfoque basado en conocimiento

En el ámbito de este trabajo, los lexicones son recursos que contienen listas de entidades de efectos adversos recogidos de diferentes fuentes como pueden ser prospectos de medicamentos, ensayos clínicos o mensajes de redes sociales. En los sistemas de detección de efectos adversos han sido empleados como enfoques únicos o como parte complementaria de un enfoque basado en técnicas de aprendizaje

automático. Con este enfoque se pretende incorporar conocimiento específico del dominio mediante coincidencias léxicas en el proceso de clasificación. Los textos así analizados podrán clasificarse de forma cualitativa entre la ausencia o presencia de efectos adversos o de forma cuantitativa por la frecuencia de menciones de efectos adversos presentes en un texto. Los lexicones continúan siendo un recurso ampliamente utilizado en la farmacovigilancia que emplea las redes sociales como fuentes de datos. Además de efectos adversos, también pueden existir lexicones de medicamentos o de términos biomédicos, un ejemplo de lexicon en español que relaciona fármacos y efectos es el SpanishDrugEffectDB [23].

Este enfoque presenta como ventaja que es muy sencillo de realizar pero presenta algunas limitaciones como que el resultado final de NER puede estar directamente relacionado con el tamaño del lexicon o que su elaboración puede requerir una gran cantidad de tiempo. Otra desventaja relacionada con los textos de redes sociales es que sus usuarios raramente emplean términos técnicos por lo que no podrían ser reconocidos por los lexicones creados.

A estos lexicones se pueden incorporar reglas y patrones de entidades que pueden aparecer en un texto. Con ellas, se pueden incorporar conocimiento del dominio al estudio, por ejemplo, si consideramos una entidad como “dosis”, el patrón podría ser cualquier dígito que venga acompañado de una abreviatura como “mg”. Las ventajas que pueden aportar estas reglas y patrones consisten en que no necesitan un conjunto de entrenamiento y presentan una alta precisión. Aunque no están exentas de inconvenientes ya que aumentan la dificultad y el tiempo dedicado a crear listas de patrones o un menor *recall* comparado con otros enfoques.

2.3.2 Enfoque basado en aprendizaje automático

Este enfoque reconoce las entidades sin necesidad de que las palabras estén incluidas en diccionarios o cumplan unos patrones como el enfoque anterior. Algunos modelos empleados en este enfoque son el modelo oculto de Markov, el modelo de entropía

máxima, máquinas de soporte vectorial (SVM, abreviatura en inglés) o un campo aleatorio condicional (CRF, abreviatura en inglés). A su vez, el aprendizaje automático utilizado para NER se ha dividido en dos tipos fundamentales: el aprendizaje supervisado y el no supervisado.

A grandes rasgos, en el aprendizaje supervisado se trabaja con datos anotados, intentando encontrar una función que, dadas las variables de entrada, les asigne una etiqueta de salida adecuada. El algoritmo se entrena con un conjunto de datos y así aprende a asignar la etiqueta de salida adecuada a un nuevo valor, prediciendo el valor de salida. Sin embargo, en el caso del aprendizaje no supervisado no se disponen de datos anotados para el entrenamiento. Sólo se conocen los datos de entrada, pero no existen datos de salida que correspondan a una determinada entrada.

Las ventajas del aprendizaje automático se pueden analizar comparándolas con los sistemas basados en conocimiento. Los algoritmos de aprendizaje automático no necesitan el trabajo de personas para desarrollar los recursos requeridos como sucede en los basados en lexicones (diccionarios, reglas) por lo que, en este aspecto, existe una reducción de tiempo. Por otro lado, este enfoque permite su extensión a otros campos ya que, si se trata de algoritmos supervisados, se podrían añadir más datos en el conjunto de entrenamiento mientras que en el enfoque basado en conocimiento se deberían modificar las reglas o patrones o incluir más diccionarios que añadirían una mayor complejidad.

De igual forma, se podrían analizar las desventajas de este enfoque frente al basado en el conocimiento. Al contrario que los sistemas basados en diccionarios o reglas, estos sistemas se caracterizan por su gran coste computacional, con grandes necesidades de memoria o tiempos de proceso elevados. Otro de los inconvenientes es que, una vez construido un modelo, no es sencillo modificarlo en el caso que los resultados esperados no sean los resultados reales. En el caso del enfoque basado en conocimiento, se pueden modificar reglas de forma directa o incluir elementos en un diccionario fácilmente.

A continuación, se resumen las ventajas y desventajas de los dos enfoques en la **tabla 1**:

Enfoque	Ventajas	Inconvenientes	Ejemplos de herramientas
Basado en conocimiento	<ul style="list-style-type: none"> · No requieren entrenamiento. · Alta precisión. 	<ul style="list-style-type: none"> · Coste elaboración diccionarios y reglas. · Falta de extensión a campos diferentes. 	<ul style="list-style-type: none"> · Tipo general: lexicones. · Dominio biomédico: GATE, UMLS.
Basado en aprendizaje automático	<ul style="list-style-type: none"> · No se dedica tiempo a mantener diccionarios y reglas. · Posibilidad de ampliación a otros campos. 	<ul style="list-style-type: none"> · Recursos computacionales. · Disponibilidad de datos anotados en el conjunto de entrenamiento (si algoritmo supervisado). · Dificultad en modificar modelo elegido. 	Stanford NER, NLTK

Tabla 1. Ventajas e inconvenientes de enfoque basados en conocimiento y en aprendizaje automático

2.4 Enfoques empleados para la extracción de características

Una vez detectadas las entidades, se hace necesaria la clasificación para comprobar la presencia o ausencia de un efecto adverso. Con el objetivo de iniciar la clasificación, los textos deben convertirse a un espacio estructurado de características con el objetivo de emplear clasificadores matemáticos. Existen tres técnicas de extracción de características: *bag-of-words* (BoW), modelos distribucionales (los más usados son *Word2Vec* y *FastText*) y modelos de lenguaje (por ejemplo, BERT). A continuación, se detallan las características, ventajas e inconvenientes de estas tres técnicas principales.

2.4.1 BoW

Ha sido ampliamente utilizada en este ámbito. Considera al texto como una colección de palabras, ignorando información semántica. Una bolsa de palabras (bag-of-words, en inglés) es una representación del texto que describe la ocurrencia de las palabras dentro de un documento. Esta bolsa implica dos cosas: un vocabulario de palabras conocidas y una medida de la presencia de palabras conocidas. El modelo sólo se preocupa si existen palabras conocidas en el documento, no dónde se encuentran. Este modelo codifica cada palabra de un vocabulario como un vector, por ejemplo, para un vocabulario de tamaño $|S|$, cada palabra se representa por un vector de dimensión $|S|$ con un 1 en el índice correspondiente de la palabra y un 0 en cualquier otro índice.

Las ventajas de este modelo se basan en su simplicidad, tanto en su comprensión como en su implementación. La principal desventaja de este modelo es la escalabilidad, ya que si tenemos un vocabulario de millones de palabras no es factible emplear este método. Otra desventaja es que se pierde la información semántica ya que la representación es discreta por lo que no puede capturar la relación semántica entre las palabras, además no tiene en cuenta el orden de las palabras en el texto.

2.4.2 Modelos distribucionales

La hipótesis distribucional considera que las palabras que se presentan en los mismos contextos tienden a tener significados similares. De esta forma, se puede aprender el significado de las palabras teniendo en cuenta el contexto en el que las palabras aparecen. A partir de este modelo se han desarrollado diferentes métodos que se proceden a resumir:

- *Word2Vec*

Emplea redes neuronales superficiales (*shallow model*) con dos capas ocultas, una *BoW* continua (*CBOW*) y el modelo *Skip-gram* con el objetivo de crear un vector para cada palabra. En la arquitectura *Skip-gram*, el modelo predice las palabras contexto

a partir de la palabra central. En el caso de la arquitectura *CBOW*, el modelo predice la palabra central a partir de una ventana de palabras contexto (la predicción es independiente del orden de las palabras contexto).

En la **figura 1** se muestra una representación gráfica de ambas arquitecturas:

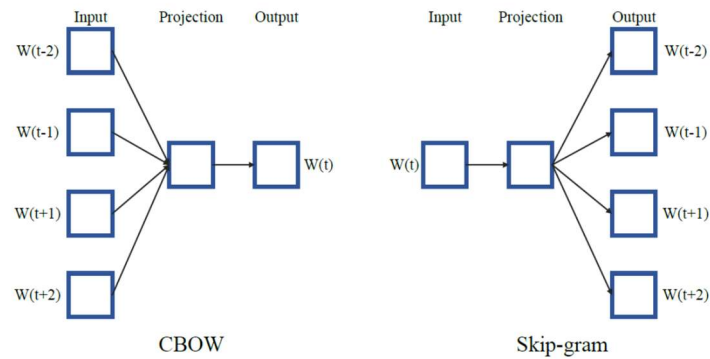


Figura 1. Representación gráfica de los modelos *CBOW* y *Skip-gram*

En el caso del modelo *CBOW*, la función objetivo J sería:

$$J = -\log \hat{P}(w_t | w_{t-n+1} \dots w_{t-1})$$

En el modelo *Skip-gram*, la función objetivo J sería:

$$J = \frac{1}{T} \sum_{t=1}^T \sum_{-n \leq j \leq n} \log p(w_{t+j} | w_t)$$

•*FastText*:

Similar a *Word2Vec*, pero en su caso cada palabra es representada como una bolsa de caracteres *n-gram*, utilizando por defecto el modelo *Skip-gram*. Por ejemplo, si se considera la palabra “introduce” y $n = 3$, *FastText* producirá la siguiente representación basada en 3-grams:

< in, int, ntr, tro, rod, odu, duc, uce, ce >

Si tenemos un diccionario de *n-grams* de tamaño G , siendo $G_w \subset 1, \dots, G$ el conjunto de *n-grams* que aparecen en la palabra w . El parámetro z_g es la representación vectorial de cada *n-gram* g . La función objetivo sería:

$$s(w, c) = \sum_{g \in G_w} z_g^T v_c$$

·*Glove*:

Muy similar a *Word2Vec*, emplea un algoritmo no supervisado para obtener representaciones vectoriales de las palabras. El entrenamiento es realizado sobre la base de estadísticas de co-ocurrencias de un enorme *corpus*. La función objetivo se define como:

$$f(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

w_i : vector de la palabra i

P_{ik} : probabilidad de que la palabra k ocurra en el contexto de la palabra i

Por último, es necesario comentar las ventajas e inconvenientes de este tipo de modelos. La ventaja de este modelo es que puede capturar la posición de las palabras en el texto, a diferencia del modelo *BoW*. Como inconvenientes más destacados se encuentran el alto consumo de recursos hardware (memoria, tiempo de computación) que requieren y, además, no pueden capturar la polisemia de las palabras.

2.4.3 Modelos de lenguaje

Los modelos de lenguaje han ganado mucho protagonismo en los últimos años, la clave en su método es que emplea el conocimiento de grandes *corpora* empleando redes neuronales y luego se transfiere el conocimiento a las tareas encomendadas. Estos

modelos de lenguaje (también llamadas incrustaciones de palabras contextualizadas) superaron la mayor limitación del enfoque clásico de los modelos distributivos: la polisemia y homonimia. Uno de los modelos llamado BERT lo consigue mediante una arquitectura que recuerda a una pila de codificadores de *transformers* entrenados. El vector que BERT asigna a una palabra es una función de la sentencia por lo que una palabra puede tener diferentes vectores basado en los contextos.

Las ventajas de este modelo son los inconvenientes que tenían los modelos distributivos que no podían diferenciar los diferentes significados de una palabra, pero eso viene acompañado de su gran desventaja que es la enorme cantidad de recursos que se requieren para que funcione correctamente.

2.5 Enfoques empleados para la clasificación

Con la elección de un buen clasificador se puede determinar qué modelo es el más eficiente para la tarea encomendada. Tradicionalmente, se han empleado algoritmos sencillos como la regresión logística o el clasificador de Naïve Bayes, que no requieren una gran cantidad de recursos. Otras técnicas empleadas han sido el algoritmo de k vecinos más cercanos (k-NN, abreviatura en inglés), máquinas de soporte vectorial (SVM) o el campo aleatorio condicional (CRF). En los últimos tiempos se han empleado modelos de redes neuronales cuyo éxito reside en su capacidad para modelar relaciones complejas y no lineales en los datos. A pesar de sus obvias ventajas, los modelos neuronales no están exentos de inconvenientes como establecer los pesos de las conexiones neuronales, dificultad en construir el modelo o el alto esfuerzo computacional que necesitan comparado con las técnicas más tradicionales.

2.6 Elección de los enfoques empleados en este trabajo y justificación

El objetivo de este trabajo es diseñar un método que identifique las RAM en mensajes de redes sociales como Twitter. La identificación de estas RAM se consigue mediante el

empleo de técnicas de NER basadas en el conocimiento, pero también en técnicas de aprendizaje automático. Mediante la filtración de palabras contenidas en un diccionario, se consigue un menor esfuerzo computacional y, a la vez, se consigue un mejor rendimiento en cuanto a precisión. No obstante, cuando la aplicación de diccionarios sea, por rendimiento, imposible se utilizarán técnicas de aprendizaje automático. Por último, la extracción de características permitirá una posterior clasificación y evaluación de nuestro sistema. A continuación, se detallan los enfoques elegidos en cada paso.

Con el objetivo de reconocer entidades se ha empleado un sistema híbrido basado en el conocimiento y en el aprendizaje automático. Por un lado, se puede construir un *lexicon* de términos referidos a medicamentos, con un número finito de entidades a reconocer, relativamente fácil de construir y que además se puede emplear en diferentes etapas, ya que en la búsqueda de *tweets* se ha empleado un filtro que contiene alguno de los términos incluidos en ese diccionario. Esto hace que la principal limitación de los *lexicones* como es el tiempo dedicado a construirlo se reduce al ser necesario en dos etapas diferentes del sistema. Por otro lado, emplear sólo algoritmos de aprendizaje automático para NER implicaría un coste computacional elevado comparado con nuestro diccionario. Sin embargo, en cuanto a términos biomédicos, se da como única opción válida evitar construir un *lexicon* con todos los términos. Aunque en posteriores etapas de esta memoria se desarrolla, para la detección de términos biomédicos se emplea una herramienta basada en un método no supervisado (*QuickUMLS*). La ventaja fundamental frente al método supervisado es que se evita la necesidad de un *corpus* anotado que requiere dedicarle una gran cantidad de tiempo en su elaboración, además del tiempo de entrenamiento y test del propio algoritmo.

Con respecto a la extracción de características, los puntos fuertes de los modelos de lenguaje son obvios, aunque también sus puntos débiles. Por otro lado, no siempre es tan evidente que sistemas más complejos que emplean BERT consiguen unos resultados mucho mejores en tareas de clasificación, especialmente cuando los datos propios difieren de los datos empleados para el entrenamiento en BERT u otros modelos preentrenados. Además, nuestra tarea de clasificación es binaria que consiste en detectar si existe o no existe un efecto adverso por lo que se podrían emplear sistemas más simples. Ante la duda entre modelos *BoW* o modelos distribucionales como

Word2Vec, ambos pueden ser útiles y sencillos de implementar por lo que se optará por una combinación de ambos modelos. Mediante la extracción de características por *BoW* se pueden calcular las ocurrencias de un término en el texto considerado, mientras que un modelo como *Word2Vec* podrían servir como refuerzo de la clasificación. Las ventajas de esta combinación serán la sencillez en la implementación, una primera etapa de extracción con poco esfuerzo computacional y la capacidad de incluir la posición de las palabras en el texto. Los modelos distribucionales considerados han sido *Word2Vec* y *FastText*, ya que *Glove* es muy similar a *Word2Vec* y se considera que dan resultados similares para muchas tareas. Se considera que los datasets con los que son entrenados o la longitud de los vectores tienen un mayor impacto que los propios modelos.

Finalmente, en cuanto a los algoritmos empleado en la clasificación, se ha optado por algoritmos sencillos, ampliamente utilizados, evitando las redes neuronales que podrían incrementar el esfuerzo computacional con un empeoramiento del rendimiento del sistema y con poco beneficio. Teniendo en cuenta que se trata de una clasificación binaria y que los algoritmos tradicionales presentan un buen rendimiento, se opta por la elección de algoritmos como regresión logística, Naïve Bayes, SVM, CRF y k-NN. Los resultados de estos algoritmos se registran y comparan para seleccionar el que obtenga una mayor precisión y mejor rendimiento.

3. Planteamiento del problema

3.1 Construcción de diccionario de medicamentos

En la construcción del diccionario de medicamentos se emplearon fuentes que fuesen completas, de fácil acceso y que incluyesen los medicamentos autorizados en España. Aunque existen varias de ellas que cumplen los dos primeros requisitos, en ocasiones no contemplan los medicamentos autorizados sólo en España y añaden los autorizados en otros países de habla hispana o en Estados Unidos.

Después de este filtrado, se consideraron dos fuentes principales de información:

- CIMA [24]
- Vademecum [25]

3.1.1 CIMA

CIMA es una fuente de información que contiene todos los fármacos y medicamentos autorizados y comercializados en España. Esta base de datos es creada, mantenida y actualizada por la AEMPS, que es un organismo autónomo que entre otras tareas se encarga de la autorización, mantenimiento o revocación de la comercialización de todos los medicamentos existentes en España. Está disponible para los profesionales sanitarios y el público general a través de su página web, e incluye entre su información: nombre del medicamento, principios activos, excipientes, código nacional, número de registro de la autorización, fecha de autorización de la comercialización, laboratorio responsable, forma farmacéutica, vía de administración, forma de la presentación, estado actualizado de la comercialización (autorizado/temporalmente suspendido/revocado), códigos *Anatomical Therapeutic Chemical* (ATC) , condiciones de prescripción y dispensación, ficha técnica y prospecto del paciente.

Además de toda esta información, se encuentra disponible un recurso llamado “Base de datos completa con el Nomenclator de prescripción” en el que se encuentra toda la información disponible en forma de archivos xml, son 15 archivos en los que cada archivo contiene información de medicamentos clasificada por ATC, denominación comercial, forma farmacéutica, laboratorios, principios activos, situación del registro, etcétera.

```
<atc>
  <nroatc>3</nroatc>
  <codigoatc>A01A</codigoatc>
  <descatc>A01A - PREPARADOS ESTOMATOLÓGICOS</descatc>
</atc>
<atc>
  <nroatc>4</nroatc>
  <codigoatc>A01AA</codigoatc>
  <descatc>A01AA - Agentes para la profilaxis de las caries</descatc>
</atc>
<atc>
  <nroatc>5</nroatc>
  <codigoatc>A01AA01</codigoatc>
  <descatc>A01AA01 - Fluoruro de sodio</descatc>
</atc>
<atc>
  <nroatc>6</nroatc>
  <codigoatc>A01AA02</codigoatc>
  <descatc>A01AA02 - Monofluorofosfato de sodio</descatc>
```

Figura 2. Ejemplo de entrada del archivo DICCIONARIO_ATC.xml

```
<principiosactivos>
  <nroprincipioactivo>3219</nroprincipioactivo>
  <codigoprincipioactivo>2A</codigoprincipioactivo>
  <principioactivo>RANITIDINA</principioactivo>
</principiosactivos>
<principiosactivos>
  <nroprincipioactivo>3220</nroprincipioactivo>
  <codigoprincipioactivo>2CH</codigoprincipioactivo>
  <principioactivo>RANITIDINA HIDROCLORURO</principioactivo>
</principiosactivos>
<principiosactivos>
  <nroprincipioactivo>5600</nroprincipioactivo>
  <codigoprincipioactivo>5A</codigoprincipioactivo>
  <principioactivo>RIBES NIGRUM</principioactivo>
</principiosactivos>
<principiosactivos>
  <nroprincipioactivo>6380</nroprincipioactivo>
  <codigoprincipioactivo>9A</codigoprincipioactivo>
  <principioactivo>RETINOL</principioactivo>
</principiosactivos>
<principiosactivos>
  <nroprincipioactivo>6381</nroprincipioactivo>
  <codigoprincipioactivo>9AC</codigoprincipioactivo>
  <principioactivo>RETINOL ACETATO</principioactivo>
</principiosactivos>
<principiosactivos>
  <nroprincipioactivo>8343</nroprincipioactivo>
  <codigoprincipioactivo>9B</codigoprincipioactivo>
  <principioactivo>VITAMINA A</principioactivo>
</principiosactivos>
```

Figura 3. Ejemplo de entrada del archivo DICCIONARIO_PRINCIPIOS_ACTIVOS.xml

A fecha de marzo del 2022, la base de datos contiene 41172 presentaciones comerciales con 24048 medicamentos únicos. Debido a esta enorme y exhaustiva información, este recurso ha sido el preferido para obtener el diccionario de medicamentos.

Con el objetivo de eliminar información no importante en este proyecto, se hizo un filtrado para eliminar cualquier información superflua que no contenga ni principios activos ni nombres comerciales del medicamento. Esta información correspondía, como se puede observar en la **figura 4**, a que el nombre comercial contiene información tal como la dosis, forma farmacéutica, método de administración o presentación comercial. Así quedaron nombres de presentaciones comerciales únicas al igual que los principios activos sin tener en cuenta otras salvedades.

ALLYNAT 240 mg COMPRIMIDOS RECUBIERTOS, 100 comprimidos
ALLYNAT 240 mg COMPRIMIDOS RECUBIERTOS, 20 comprimidos
ALLYNAT 240 mg COMPRIMIDOS RECUBIERTOS, 40 comprimidos
ALMAX 1g/7,5 ml SUSPENSION ORAL , 1 frasco de 225 ml
ALMAX 500 mg COMPRIMIDOS MASTICABLES , 30 comprimidos
ALMAX FORTE 1,5 g SUSPENSION ORAL , 16 sobres
ALMOGRAN 12,5 mg COMPRIMIDOS RECUBIERTOS CON PELICULA , 3 comprimidos

Figura 4. Ejemplo de estructura en CIMA de nombres comerciales de medicamentos

3.1.2 Vademecum

El sitio web que corresponde a www.vademecum.es es una guía de productos farmacéuticos que se publica y actualiza de forma periódica, y en la cual aparece recopilada la información otorgada por las compañías farmacéuticas. El Vademécum es un instrumento dirigido exclusivamente a los profesionales médicos y que es distribuido por la compañía editora únicamente a dichos profesionales. Presenta un buscador propio con diferentes índices ya sea por indicaciones, principio activos, marca comercial, laboratorio o por clasificación ATC. Si bien presenta un catálogo muy extenso y muy completo, en nuestro caso sólo sirvió para obtener una información complementaria que nos daba CIMA.

La información disponible en este sitio web aunque gratuita requirió el empleo de técnicas de *webscraping*. El webscraping es una técnica que se utiliza para obtener de forma automática el contenido que hay en páginas web a través de su código HTML. El uso de estas técnicas tienen como finalidad recopilar grandes cantidades de datos de diferentes páginas web cuyo uso posterior puede ser muy variado, aunque en el caso de este proyecto fue construir un diccionario de medicamentos y principios activos.

Para ello se emplearon las siguientes herramientas:

- Selenium WebDriver: Esta herramienta permite a los desarrolladores probar y registrar las interacciones con una aplicación web y luego repetirlas las veces que se desee, de forma completamente automática. Selenium engloba un número amplio de proyectos de software libre empleados para automatización del desarrollo web, que soporta diferentes lenguajes, entre ellos el lenguaje de programación de este proyecto (Python). Mediante una API como WebDriver permite al desarrollador simular las interacciones con cualquier navegador ya sea Firefox, Chrome, Edge, Safari o Internet Explorer. Desde 2018, la API es un estándar W3C oficial.

- BeautifulSoup (bs4): Se trata de una librería que facilita extraer información de las páginas web. Extrae información de archivos HTML, XML y otros lenguajes de marcado. Esta librería ayuda a descargar el contenido que interesa de una página web y lo almacena sin necesidad de guardar también todo el contenido de HTML.

- urllib2: Se trata de un módulo Python para acceder y utilizar recursos de internet identificados por *Uniform Resource Locators* (URLs). Ofrece una interfaz muy simple, a través de la función `urlopen`. Esta función es capaz de acceder a URLs usando una variedad de protocolos diferentes.

Posteriormente, se eliminaron duplicidades de principios activos, así como aquellas presentaciones comerciales que tuviesen diferentes sufijos que corresponden a formas de liberación como *flas*, *retard* o *prolong* para evitar la reiteración de marcas comerciales o falsos positivos en la posterior minería de datos de la red social.

Una vez comparada la información tanto de CIMA como la obtenida de Vademecum se obtuvieron 5636 medicamentos y principios activos que constituyeron el diccionario de fármacos de este proyecto.

3.2 Construcción de diccionario de términos médicos

Al igual que en la construcción del diccionario de medicamentos, se emplearon fuentes que fuesen completas, de fácil acceso y que presentasen los términos en español. Uno de los grandes inconvenientes en la elaboración de estos diccionarios de términos médicos es que, si se comparan con medicamentos, no existe una base de datos oficial y completa que incluya cualquier tipo de terminología biomédica que pudiera aparecer en un texto biomédico. Mientras en los medicamentos se tienen bases de datos elaboradas por administraciones públicas y otras instituciones con todos los medicamentos, principios activos y sustancias químicas comercializadas en España, no existen las mismas que engloben a términos biomédicos.

Aunque existen diferentes plataformas, se optó por dos fuentes de términos médicos:

- MedDRA (español)-MDRSPA [26]
- Diccionario de términos médicos de la Real Academia Nacional de Medicina [27]

3.2.1 MedDRA- (español)

Como se ha comentado anteriormente, MedDRA tiene la ventaja que es multilingüe de forma que cada país puede trabajar en su idioma oficial. Cada término tiene asociado un código que es el mismo con independencia del idioma.

Es un vocabulario con una estructura jerarquizada, compuesto por cinco niveles, desde un nivel más general hasta otro más específico, llamado LLT (*Lowest Level Terms*). Estos términos se utilizan para definir hallazgos en la práctica clínica. En el nivel siguiente llamado PT (*Preferred Terms*), donde cada término es un concepto médico individual

que puede ser un síntoma, signo, diagnóstico de enfermedad, indicación terapéutica, investigación, procedimiento quirúrgico o médico y, por último, historia familiar o sociomédica. La relación de PT con LLT es que cada LLT está relacionado con un PT pero cada PT está relacionado con al menos un LLT.

Subiendo en la jerarquía, se encuentran los HLT (*High Level Terms*) que están relacionados con los PT ya sea por anatomía, patología, fisiología, etiología o función. A su vez, los HLT se agrupan en HLGT (*High Level Group Terms*). Por último, estos HGLT se agrupan en SOC (*System Organ Classes*), basados en etiología, sitio de manifestación o propósito.

Como ejemplo de la exhaustividad de este vocabulario se puede ver el número total de términos en la **tabla 2**. Es, por ello, que al ser utilizado como diccionario de clasificación de efectos adversos por el *International Conference of Harmonisation* (ICH) ha sido elegido como principal diccionario de términos médicos.

Como principal inconveniente es que al emplear redes sociales que utilizan un lenguaje informal y no dado a términos científicos, se podrían infraestimar muchos efectos adversos indicados por los usuarios en un lenguaje coloquial.

ID Tipo Semántico	Nombre del Tipo Semántico	Términos
T047	Enfermedad o Síndrome	18471
T033	Hallazgos	10314
T191	Proceso neoplásico	5771
T037	Lesión o intoxicación	4576
T061	Procedimiento terapéutico o preventivo	4277
T046	Función patológica	3535
T184	Signo o síntoma	2314
T059	Procedimiento de laboratorio	1800
T048	Disfunción mental o de comportamiento	1542
T034	Resultado de laboratorio o de pruebas	1418
T019	Anormalidad congénita	1210
T060	Procedimiento diagnóstico	1141

Tabla 2. Distribución de tipos semántica de MedDRA

3.2.2. Diccionario de términos médicos de la Real Academia Nacional de Medicina

La Real Academia Nacional de Medicina ha elaborado un diccionario de términos médicos. Mediante técnicas de *webscraping* de la misma forma que se realizó para la creación del diccionario de medicamentos, se obtuvo un conjunto de términos. Se han

empleado las mismas herramientas y librerías utilizadas anteriormente en el diccionario de medicamentos.

Este diccionario de términos contiene 109706 términos, aunque no presenta una estructura jerarquizada ni tan exhaustiva como la que presenta MedDRA. Se pensó en ser utilizado como fuente complementaria a MedDRA para mejorar la detección de efectos adversos. Tras su uso y debido a la alta generación de falsos positivos, se descartó como fuente de diccionario de términos médicos.

3.3 Minado de mensajes en Twitter

Tal y como se ha comentado en la Introducción, Twitter ha sido la red social seleccionada para la recolección de los mensajes de usuarios. Durante dos meses se procedió a hacer una búsqueda de los mensajes de los usuarios. Se emplearon dos librerías fundamentalmente:

- Tweepy: Es una de las librerías más populares para acceder a Twitter. Se accede, previa creación de cuenta de desarrollador de Twitter y obtención de credenciales. Tiene como principal desventaja un límite de recolección de 3200 tweets y sólo permite hacer recolección de tweets que tengan menos de una semana.

- Snsrape: Se trata de una herramienta para *scraping* especializada en redes sociales. Con esta herramienta es posible capturar elementos como los perfiles del usuario, hashtags, búsquedas o mensajes relevantes. Se puede emplear en muchas redes sociales como Facebook, Instagram, Telegram, Twitter,...Entre los atributos que puede rescatar de un tweet los más destacables para este proyecto fueron la localización geográfica, fecha, texto del mensaje, número de identificación del tweet, usuario. Además con respecto a Tweepy, presenta como ventaja fundamental el poder realizar búsquedas sin importar la fecha ni el límite de una semana, además había un número ilimitado de tweets que se podían analizar. Otra ventaja sustancial es que se podía seleccionar la eliminación de retweets ya que podían perturbar los hallazgos en las fases posteriores del proyecto. Aunque en un principio se optó por Tweepy, al final se obtuvo una mayor cantidad y calidad de tweets mediante Snsrape, por lo que todos los mensajes obtenidos fueron mediante esta herramienta. La búsqueda se realizó

utilizando el diccionario de medicamentos y fármacos y empleando como intervalo de tiempo 2 meses. Finalmente se obtuvieron 1029001 tweets (previos al preprocesamiento) que contenían en algún espacio del tweet una referencia a alguno de los medicamentos o principios activos incluidos en nuestro diccionario. En la **tabla 3** se indican las menciones de las primeros 30 marcas comerciales detectadas.

MARCA COMERCIAL	APARICIONES
IBUPROFENO	34055
RIVOTRIL	29427
ASPIRINA	24122
CLONAZEPAM	20170
OMEPRAZOL	15535
HIDROXICLOROQUINA	8536
AZITROMICINA	8369
DEXAMETASONA	6799
TRAMADOL	6596
DIAZEPAM	6538
LORATADINA	6289
FOLICO	6204
MORFINA	6019
ENEAS	5758
NAPROXENO	5668
VALIUM	5159
BUSCAPINA	4902
EVISTA	4445
AILYN	4354
DALSY	4310
BCG	4043
PROZAC	4022
ALPRAZOLAM	3926
DICLOFENACO	3919
SERTRALINA	3907
AAS	3878
MISOPROSTOL	3739
MIDAZOLAM	3695
PARACETAMOL	3649
NOLOTIL	3565

Tabla 3. Clasificación de marcas comerciales según apariciones en la red social

De acuerdo a las menciones según la marca comercial, hacemos una distribución según ATC como refleja la **figura 5**:

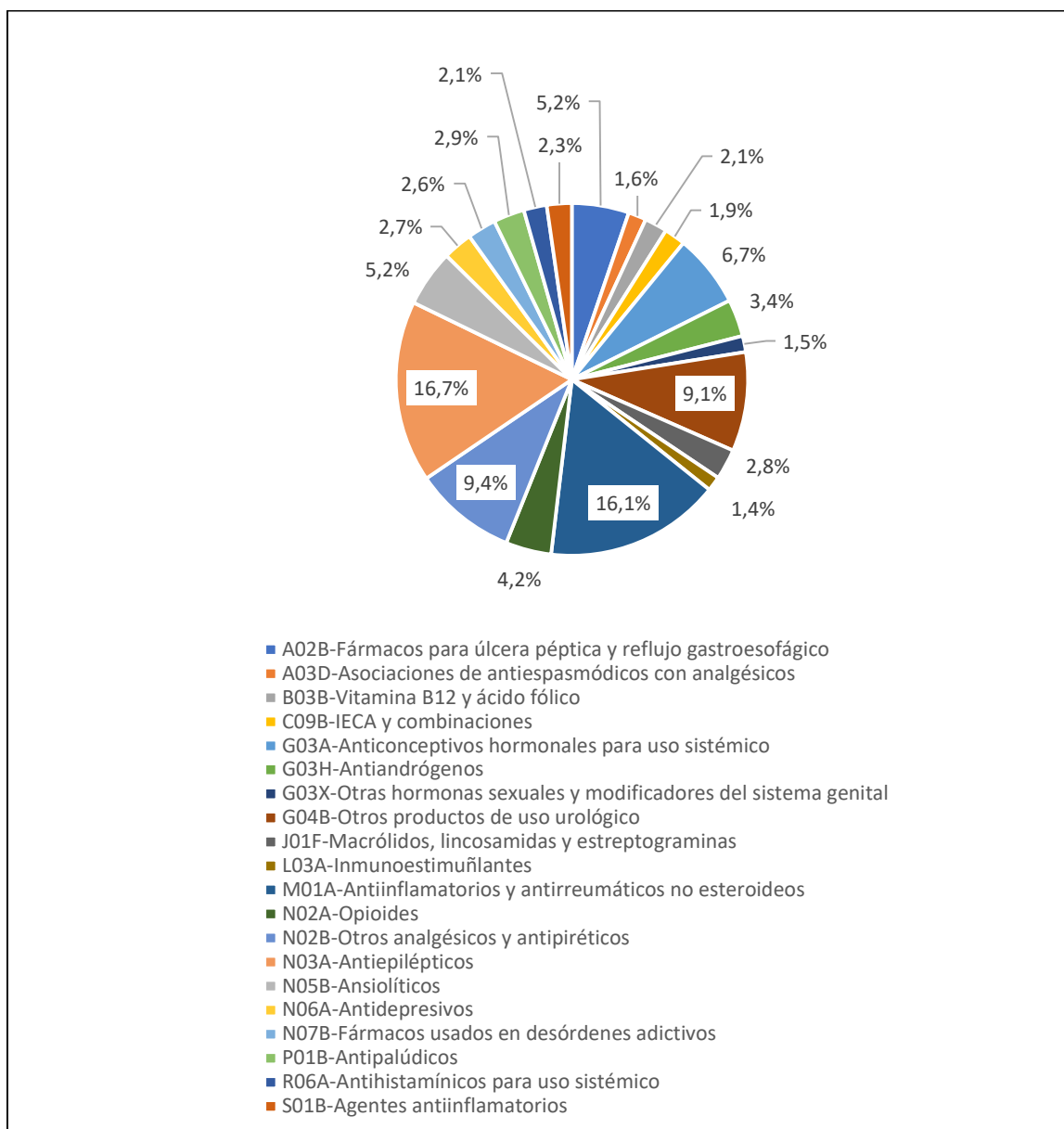


Figura 5. Distribución de menciones de marcas comerciales agrupadas por ATC

En la **tabla 4** se indican las menciones según el principio activo, así como en la **figura 6** se encuentra la distribución de menciones agrupadas según la clasificación ATC, donde prácticamente son similares a las aparecidas por marca comercial.

PRINCIPIO ACTIVO	APARICIONES
IVERMECTINA	38668
IBUPROFENO	34055
CLONAZEPAM	20170
OMEPRAZOL	15535
HIDROXICLOROQUINA	8536
AZITROMICINA	8369
REMEDESIVIR	7873
DEXAMETASONA	6799
TRAMADOL	6596
DIAZEPAM	6538
LORATADINA	6289
FOLICO	6204
MORFINA	6019
NAPROXENO	5668
BCG	4043
ALPRAZOLAM	3926
DICLOFENACO	3919
SERTRALINA	3907
MISOPROSTOL	3739
MIDAZOLAM	3695
PARACETAMOL	3649
PREDNISONA	3418
METFORMINA	3242
FENTANILO	3135
SALBUTAMOL	3058
FLUOXETINA	3029
QUETIAPINA	2992
KETOROLACO	2915
ATRACURIO	2788
MINOXIDIL	2705

Tabla 4. Clasificación de principios activos según apariciones en la red social

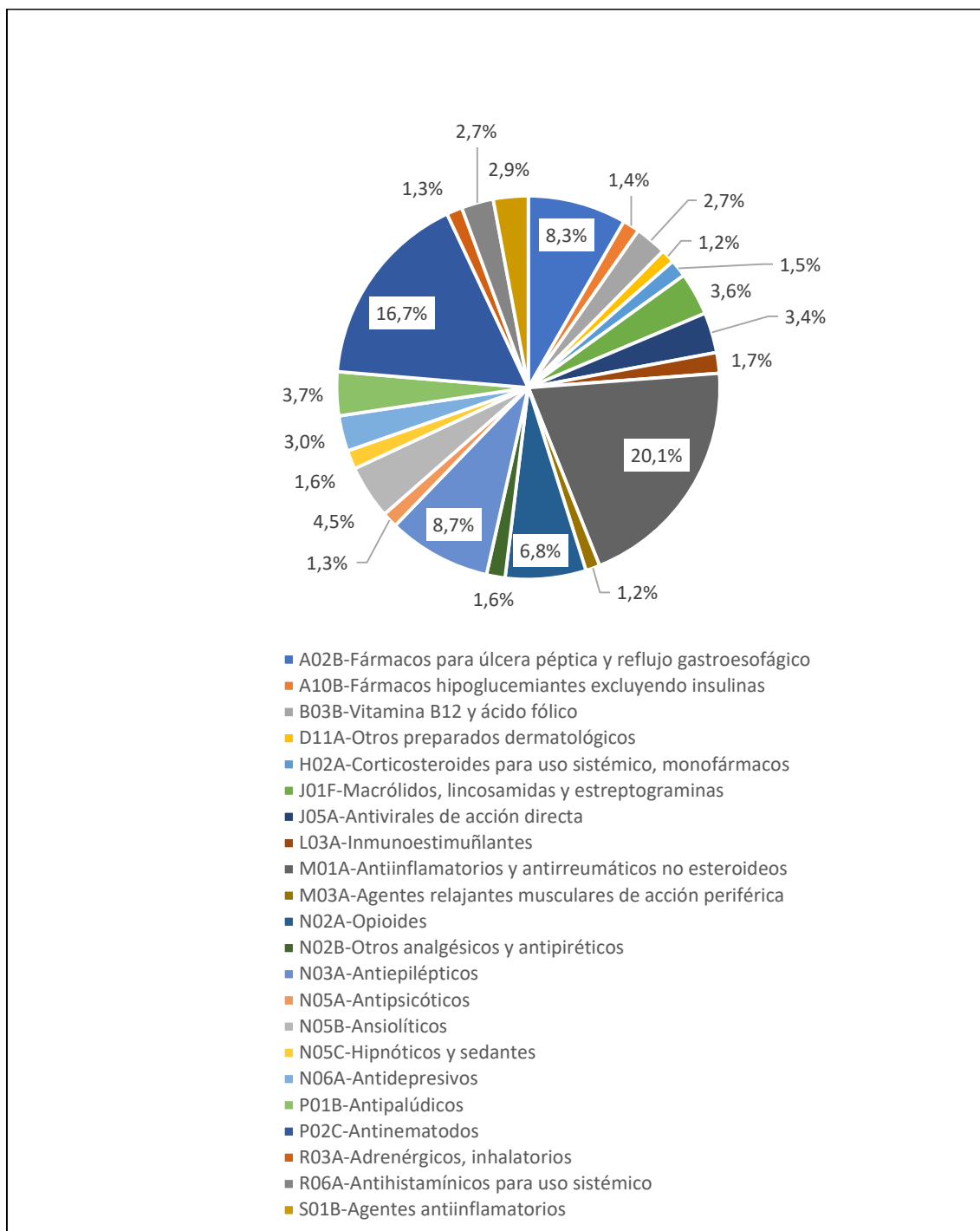


Figura 6. Distribución de menciones de principios activos agrupadas por ATC

A continuación, en la **figura 7** se exponen mensajes extraídos de la red social (se han borrado los nombres de usuarios):

1372451868870332420," Trastornos de pánico, con problemas de adaptabilidad al cambio, imagínate el año que llevo. Además la sanidad y la sociedad, con 5 años me mandaron tranxilium pediátrico para poder dormir los días que tenía un examen. A los 5 años."

1358706841866424322,"Por el Tranxilium y la Quetiapina me ha salido un sarpullido en la espalda y lo estoy odiando muchísimo. Que asco de efectos secundarios, pero claro, como para no tenerlos si tomo casi la cantidad máxima diaria recomendada"

1375016053575716867," Y la doxilamina, tan deseada y necesitada en este último año, un antihistamínico con indicación para el insomnio debido a su efecto secundario "

1372100799900303361,"Me habían recetado antes Clonazepam, pero tampoco funciona. Ya en crisis de ansiedad la opción era Diazepam vía intravenosa, una dosis alta que me mandaba a dormir."

Figura 7. Ejemplos de mensajes extraídos de la red social Twitter

3.4 Extracción de conceptos de documentos no estructurados

El problema relacionado con la extracción de conceptos de documentos no estructurados ha sido formalizado de la siguiente manera [28] :

Sea un diccionario S un conjunto de cadenas, C una colección de conceptos, de forma que un mapa asocia un concepto de la colección a uno o más cadenas en el diccionario.

$$C: C \rightarrow S$$

Dado un documento d que podemos representar como un orden secuencial de *tokens* $\{d_1, \dots, d_n\}$, una función de similitud $strsim$ y un umbral de similitud $\alpha \in [0,1]$, el algoritmo de extracción de conceptos debería ser:

$$\{(d_{ij}, c_k) \mid \exists s_h \in C(c_k) \text{ s.t. } strsim(d_{ij}, s_h) \geq \alpha\}$$

donde d_{ij} representa una secuencia de *tokens* en d y $s_h \in S$ es una cadena representando un concepto $c_k \in C$.

El método de coincidencia aproximada empleando un diccionario se puede formular como: dada una cadena objetivo x , un umbral de similitud α , un diccionario S y una función de similitud $strsim$, se desea encontrar el subconjunto $\mathcal{Y}_{x,\alpha} \subseteq S$, tal que:

$$\mathcal{Y}_{x,\alpha} = \{y \in S \mid strsim(x, y) \geq \alpha\}$$

Una solución de este problema podría ser computar la similitud de cada cadena en el diccionario S hasta conseguir la cadena objetivo x . Sin embargo, el coste computacional sería muy alto ya que tendríamos una complejidad dependiente del tamaño del diccionario S . Esto es especialmente importante cuando se pretende emplear un diccionario como UMLS que puede contener más de 6 millones de cadenas según el idioma elegido.

Esto se ha podido solventar mediante el empleo de una herramienta como QuickUMLS cuyo mecanismo se detalla a continuación:

Dado un documento d de una longitud n , un umbral de similitud α y un tamaño de ventana w , QuickUMLS genera para cada token $d_1 \in d$, todas las posibles secuencias de token $d_{ij} = \{d_i, \dots, d_j\}$, $j \in \{i, \dots, i+w-1\}$. Luego, mediante heurística, determina si la secuencia d_{ij} es una secuencia válida de tokens. Si así lo es, se procede al siguiente paso de identificar cadenas en S que sean similares a d_{ij} . Una vez que el subconjunto de todas las posibles cadenas coincidentes se determine, QuickUMLS selecciona el subconjunto más apropiado $\mathcal{Z}_{d,\alpha}$, donde α representa el umbral seleccionado y $\mathcal{Y}_{d_{ij},\alpha}$ representa el subconjunto de las cadenas dentro del diccionario S similares a d_{ij} dentro del umbral de similitud α seleccionado.

$$\mathcal{Z}_{d,\alpha} = \bigcup_{d_{ij}} (\mathcal{Y}_{d_{ij},\alpha})$$

3.5 Extracción de características (*feature extraction*)

Como se ha comentado en Antecedentes, para la extracción de características existen diferentes modelos con sus ventajas e inconvenientes. El objetivo de esta técnica en

este trabajo es convertir a segmentos de texto desestructurado en un espacio de características estructurado que, posteriormente, podrá ser empleado por un clasificador.

El modelo *BoW* es una representación simplificada y reducida de un texto basada en criterios específicos como puede ser la frecuencia de palabras en un texto. De esta forma, un texto puede representarse como un vector de bolsa de palabras. Si consideramos tres textos como:

Texto 1 : “el alprazolam me hace daño”

Texto 2: “el alprazolam no hace nada”

Texto 3: “ el alprazolam es efectivo”

El vocabulario consta de 9 palabras: “el”, “alprazolam”, “me”, “hace”, “daño”, “no”, “nada”, “es”, “efectivo”.

Si se toma cada una de las palabras y se toma cada aparición o ausencia como 1 o 0, se tiene:

	el	alprazolam	me	hace	daño	no	nada	es	efectivo	longitud texto
Texto 1	1	1	1	1	1	0	0	0	0	5
Texto 2	1	1	0	1	0	1	1	0	0	5
Texto 3	1	1	0	0	0	0	0	1	1	4

Tabla 5. Ejemplo de *BoW*

El vector del Texto 1 será : [1,1,1,1,1,0,0,0,0]

El vector del Texto 2 será : [1,1,0,1,0,1,1,0,0]

El vector del Texto 3 será : [1,1,0,0,0,0,0,1,1]

Esto nos lleva a un gran inconveniente, ya que el vector es de tamaño 9 pero en el caso de que aumente el vocabulario, también su dimensión aumentaría. Además, con *BoW* no se tiene información sobre el orden de las palabras en el texto o que sea una palabra poco común.

Teniendo en cuenta los defectos del modelo *BoW*, se puede emplear una técnica estadística que tenga en cuenta la importancia de una palabra en un *corpus* llamada TF-IDF (abreviatura del concepto inglés *Term Frequency-Inverse Document Frequency*). El valor de TF-IDF tiene en cuenta dos conceptos:

$TF(t) = (N.º \text{ de veces que el término } t \text{ en un documento}) / (N.º \text{ total de términos en el documento})$

$IDF(t) = \log(N.º \text{ total de documentos} / N.º \text{ de documentos con el término } t \text{ en ellos}).$

Finalmente, TF-IDF es calculado mediante la siguiente fórmula:

$$tfidf(t, d, D) = tf(t, d) * idf(d, D)$$

Una puntuación alta de TF-IDF se obtiene mediante un término que tiene una frecuencia alta en un documento y una frecuencia baja en el corpus. Para una palabra que aparece en casi todos los documentos, el valor de IDF se acerca a 0, lo que hace que TF-IDF también se acerque a 0. El valor de TF-IDF es alto cuando los valores de IDF y TF son altos, es decir, la palabra es rara en todo el documento, pero frecuente en un documento. Mediante esta técnica es posible reducir la dimensionalidad de los vectores del modelo *BoW* clásico, aunque sigue sin capturar la relación semántica entre palabras.

Debido a las limitaciones mencionadas de los modelos *BoW*, se han empleado otros modelos, entre los que se incluyen *Word2Vec* y *FastText*. Mediante estos modelos se consigue que los vectores obtenidos sean más densos, se tenga en cuenta el orden de aparición de las palabras y puede capturar el significado de las palabras. Además, se ha optado por una combinación de TF-IDF con estos modelos porque cuando se ha empleado la combinación de *Word2Vec* y TF-IDF, se han obtenido mejores resultados que cualquiera de los dos modelos por separado [29]. Este método permite incorporar

los pesos obtenidos en TF-IDF basados en la frecuencia de cada palabra a los vectores obtenidos con *Word2Vec*. Por otra parte, añadir *Word2Vec* a TF-IDF permite incorporar características semánticas a la extracción.

3.6 Clasificación y evaluación.

Tras haber realizado la extracción de características, se pasa a la fase de clasificación. Tal y como se comentó en el capítulo de Antecedentes, se ha optado por algoritmos tradicionales al tratarse de una clasificación binaria donde estos pueden conseguir resultados similares a las redes neuronales con una menor utilización de recursos. En la **tabla 6** se indica el algoritmo empleado en esta etapa de clasificación:

Algoritmo estadístico de aprendizaje
Entrada: todas las instancias con al menos un par de términos biomédicos y fármacos/medicamento $R(\text{fármaco}, \text{término})$
Salida: donde las instancias tienen una relación efecto adverso y fármaco/medicamento
Procedimiento: <ol style="list-style-type: none"> 1. Para cada instancia $R(\text{fármaco}, \text{término})$: Características: Extracción mediante TF-IDF solo o más <i>Word2Vec</i>. 2. Separar las instancias de la relación entre los conjuntos de entrenamiento y test. 3. Entrenar un clasificador basado en el conjunto de entrenamiento 4. Utilizar el clasificador para clasificar instancias en el set de prueba en dos clases $R(\text{fármaco}, \text{término}) = \text{positivo}$ y $R(\text{fármaco}, \text{término}) = \text{negativo}$

Tabla 6. Algoritmo estadístico de aprendizaje

Los clasificadores considerados han sido:

- Regresión logística: Técnica de clasificación que emplea una función logística para modelar la variable dependiente. La variable dependiente es binaria, por ejemplo, es o no es un efecto adverso. Matemáticamente, el planteamiento del problema se puede formular así, donde x representan los atributos del problema (en este caso, una cadena de texto), de forma que sus valores podrían corresponder a cuantas veces aparece una palabra en un texto mientras que la predicción y representa la probabilidad de que aparezca un efecto adverso como se observa en la siguiente fórmula:

$$y = \sigma(z) = \sigma(WX) = \sigma\left(\sum (w_i x_i)\right) = \sigma\left(\sum (w_0 x_0 + w_1 x_1 + \dots + w_n x_n)\right)$$

La función empleada llamada función logística o sigmoidea, se obtiene mediante la fórmula:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Las principales ventajas de la regresión logística se basan en que es fácil de implementar, interpretar y fácil de entrenar, así como la rapidez en la clasificación de registros desconocidos mientras que los inconvenientes se basan en que, si el número de observaciones es menor que el número de características podría conducir al *overfitting*, así como que asume la linealidad entre la variable dependiente y la variable independiente.

· Support Vector Machine (SVM): Es un modelo lineal para problemas de clasificación y de regresión. Puede resolver problemas lineales y no lineales, ya que el algoritmo que usa se basa en la creación de una línea o un hiperplano que separa los datos en clases. Cuando los datos se pueden separar mediante una línea recta se habla de SVM lineal mientras que cuando no se puede realizar linealmente se recurren a las funciones kernel que transforman espacios no lineales en lineales. Las ventajas que presenta SVM en tareas de clasificación son la efectividad en espacios altamente dimensionales, incluso donde el número de dimensiones es mayor que el número de muestras y se pueden emplear diferentes funciones *kernel* que pueden ser diseñadas específicamente para el problema a tratar. Por otra parte, el rendimiento puede empeorar si no se elige un valor adecuado de validación cruzada. Otro inconveniente es la posibilidad de *overfitting* en caso de que el número de características sea superior al número de muestras. Por último, si se emplean grandes *datasets*, el tiempo de entrenamiento puede ser muy elevado, e incluso elegir la función kernel correcta podría ser computacionalmente intensa.

· Naïve Bayes Multinomial: es un clasificador relativamente sencillo de implementar. Uno de los aspectos desfavorables de estos clasificadores es que asume que los términos que aparecen en un documento son todos independientes entre sí lo que puede no ser totalmente cierto debido a la estructura del lenguaje. La forma multinomial considera el número de apariciones del término para evaluar la contribución de la probabilidad condicional dada la clase con lo que el modelado de cada documento se ajusta mejor a la clase a la que pertenece. Las ventajas que presenta este método es que se pueden integrar múltiples variables en los cálculos para clasificar datos y es fácil de integrar con características conocidas en un conjunto de datos. Por otra parte, presenta como desventajas que los predictores se consideran independientes entre sí, que puede no ser cierto y generar un modelo que no se ajuste correctamente. El esfuerzo computacional aumenta a medida que aumentan las características de la muestra.

· k-NN: el algoritmo clasifica cada dato nuevo en el grupo que corresponda, según tenga k vecinos más cerca de un grupo o de otro. De esta forma, calcula la distancia del elemento nuevo a cada uno de los existentes, y ordena dichas distancias de menor a mayor para ir seleccionando el grupo al que pertenecer. Este grupo será, por tanto, el de mayor frecuencia con menores distancias. Se denomina también un algoritmo de aprendizaje vago, ya que en la fase de entrenamiento sólo almacena el *dataset* y cuando llegan nuevos datos, clasifica los datos en la categoría que es más similar a los nuevos datos. Este algoritmo presenta alta precisión, pero es computacionalmente costoso ya que almacena todos los datos y utiliza una gran cantidad de memoria.

· Bosque aleatorio (CRF): conjunto de árboles de decisión combinados en el que ninguno de ellos ve todos los datos de entrenamiento. Esto hace que cada árbol se entrene con distintas muestras de datos para un mismo problema. De esta forma, al combinar sus resultados, unos errores se compensan con otros y tenemos una predicción que generaliza mejor. La principal ventaja de este algoritmo es que es muy fácil de usar con buenos resultados de predicción ya que es fácil de entrenar, sin embargo, existe una alta tendencia al *overfitting* y puede ser ineficiente cuando se utilizan un número alto de árboles.

En cuanto a la evaluación de los resultados obtenidos mediante técnicas de detección/extracción de efectos adversos, existen dos tipos de evaluaciones:

- Cualitativa: Comparando tipos de medicamentos, según la gravedad de sus efectos adversos [6].

- Cuantitativa: Se han empleado métricas como precisión, exactitud, exhaustividad o *f-score*. Estas métricas se emplearon en los primeros trabajos que empleaban aprendizaje supervisado y, posteriormente, con la inclusión del aprendizaje no supervisado se han empleado para comparar los resultados de los dos enfoques [28].

Como resumen de todas las etapas que se van a desarrollar en este proyecto a grandes rasgos, se puede considerar que la detección de efectos adversos a través de las redes sociales sigue las etapas descritas en la **figura 8**.

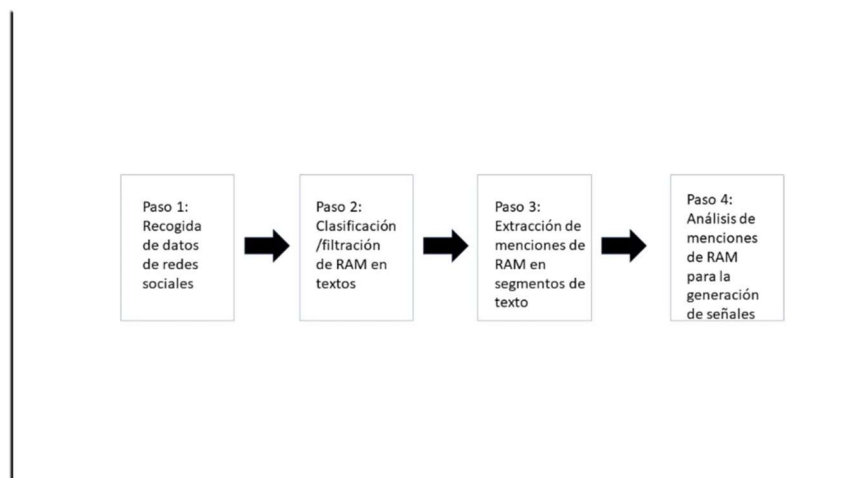


Figura 8. Pipeline de detección de efectos adversos

4. Descripción del sistema desarrollado

4.1 Esquema de la solución

El sistema se inicia con los textos de los tweets que sufren una etapa de limpieza para intentar eliminar el ruido que afecta a las fases posteriores perjudicando la etapa final de extracción de entidades. Tras finalizar cada una de las fases, el resultado final será la extracción de entidades de forma que sea posible detectar la relación entre efectos adversos y medicamentos. En la **figura 9** se muestra el pipeline correspondiente al proyecto desarrollado:

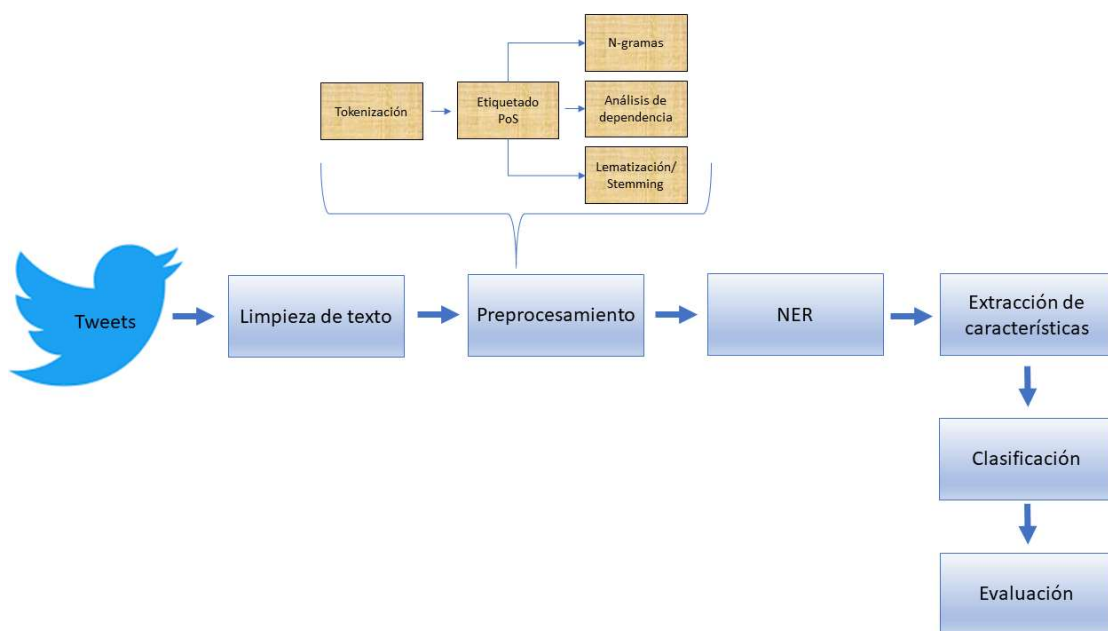


Figura 9. Pipeline del sistema para la detección de efectos adversos

4.2 Limpieza de texto

Una vez obtenidos los mensajes, es necesaria la limpieza de los mensajes mediante una etapa de preprocesamiento de los textos obtenidos, para evitar los posibles errores de codificación.

Esta primera etapa se hizo con métodos generales (algunos ejemplos se pueden encontrar en la **figura 10**):

- Pasar texto a minúsculas (función minusculas)
- Eliminación de datos de usuario o de respuestas a un usuario en concreto (función eliminaArroba).
- Eliminación de comillas (función eliminaComillas)
- Eliminación de direcciones url (función eliminaWeb).
- Eliminación de hashtags (función eliminaHashtag).
- Sustitución de exclamaciones y de interrogaciones con más de un carácter (función sustituyeMultiExclamaciones y sustituyeMultiInterrogacion)
- Eliminación de asteriscos (función eliminarAstericos)
- Sustitución de *emojis* (función sustituyeEmoji).
- Eliminación de espacios en blanco (función eliminarWhiteSpace)
- Eliminación de guiones (función eliminarGuiones)
- Eliminación de repetición de 3 o más caracteres dentro de una cadena.
(función eliminarRepeticiones)
- Eliminación de otros signos de puntuación, como – o [] (función encontrarCorchetes, eliminarCorchetes, eliminarParentesis)

Mediante estos métodos por ejemplo se consiguió que los mensajes que mencionaban *rivotril* pasaran de 297332 *tokens* a 267161 *tokens*.

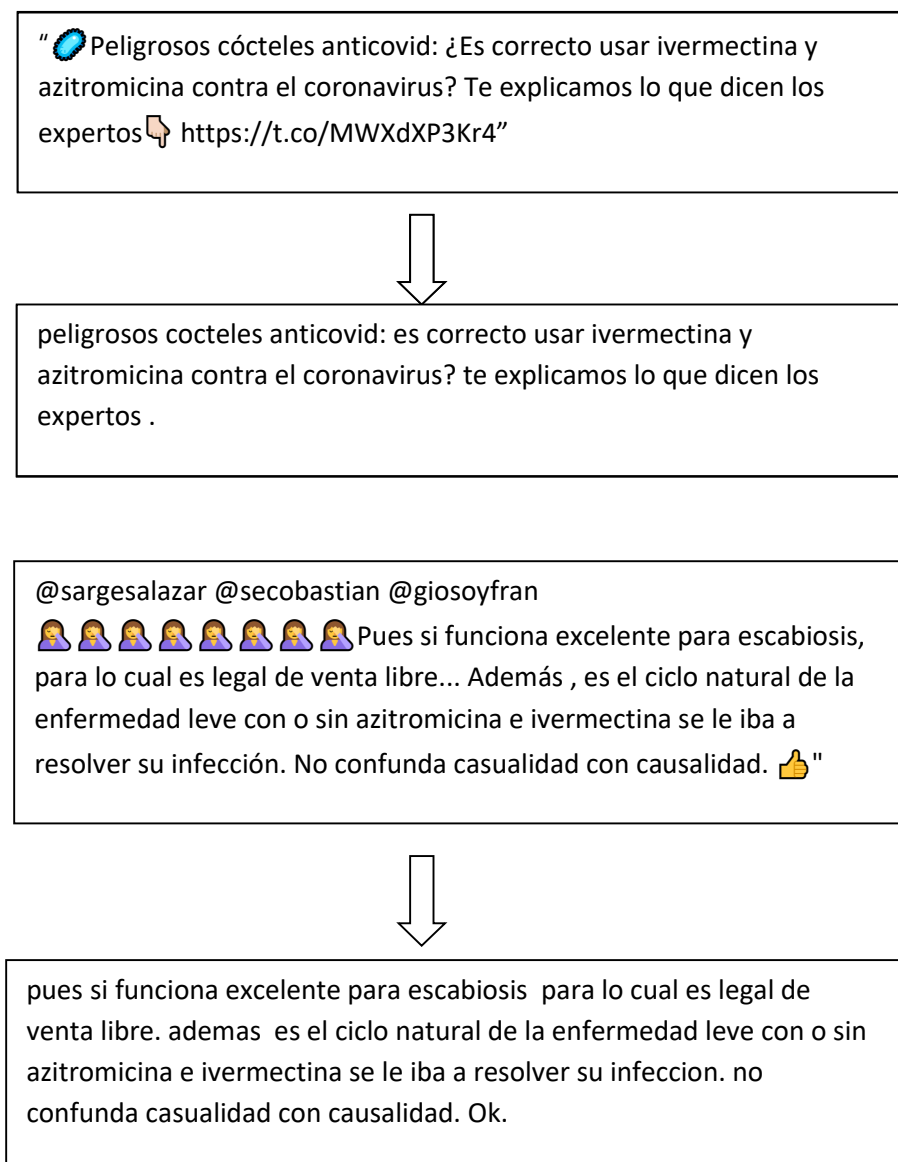


Figura 10. Ejemplos de preprocesamiento de tweets con eliminación del usuario

Posteriormente, se valora si los tweets siguen conteniendo “impurezas” en esta etapa que pueden consistir en espacios en blanco duplicados, signos de puntuación inadecuados, direcciones web o *hashtags*... Para averiguar el nivel de “impureza”, se elaboró un método propio que consistió en localizar aquellos caracteres objeto de la limpieza, como pueden ser @, &, #, guiones o cualquier otro carácter de los limpiados. El nivel de “impurezas” sería el cociente entre el número total de caracteres que deberían ser eliminados pero no lo fueron y el número total de caracteres presentes en

el texto. Por ejemplo, en el análisis de la limpieza de los textos que contienen “rivotril” se obtuvo una limpieza final de aproximadamente de 0.0260%.

Uno de los grandes problemas en esta etapa son los *emojis*, no existe una óptima solución al problema que representa, ya que si se eliminan se puede perder información y, por lo tanto, exactitud en el proceso final y en la extracción de información. Por otro lado, la posibilidad de transformar esos símbolos en unicode es posible y de ahí extraer su significado de acuerdo a las tablas normalizadas[30].

Considerando las limitaciones de analizar todos los *emojis* de forma exhaustiva, se consideró que aquellos *emojis* que afectasen al “sentimiento” de un mensaje, deberían ser traducidos a su correspondiente código y su significado de acuerdo a las tablas normalizadas. Aquellos *emojis* que sólo reflejan lugares, género o figuras geométricas fueran eliminados al no aportar ningún tipo de información emocional o de síntomas y signos clínicos.

En la **tabla 7** se observan ejemplos de procesamiento de *emojis*:



Emoji	Unicode	Significado	Resultado
	2196 FE0F	“up-left arrow”	(eliminado)
	1F620..1F625	“angry face”	enfadado

Tabla 7. Ejemplos de preprocesamiento de *emojis*

4.3 Preprocesamiento

4.3.1 Tokenización

La tokenización consiste en segmentar el texto en tokens, en este caso, palabras y signos de puntuación. Un ejemplo de tokenización se puede ver en la **figura 11**:

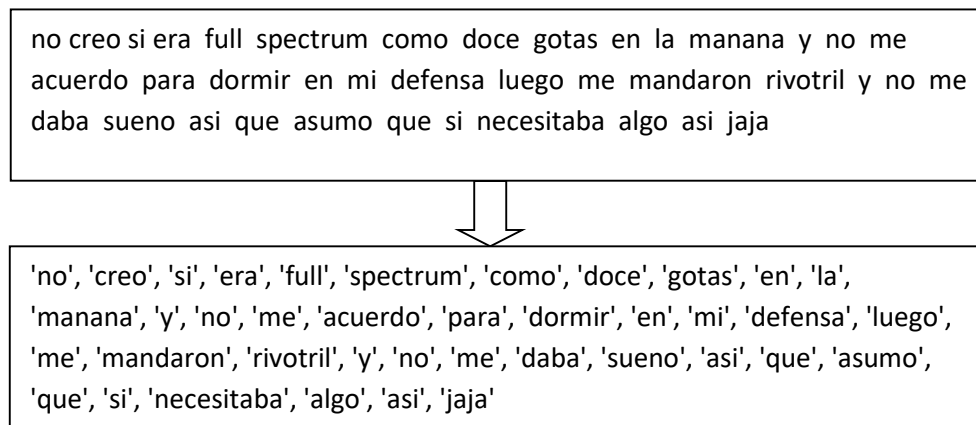


Figura 11. Ejemplo de tokenización

Tras el preprocesamiento o limpieza de los datos, la tokenización corresponde a la primera fase para la detección y es fundamental para posteriores fases como el etiquetado Part-of-Speech (*PoS tagging*). En esta fase y en próximas fases del procesamiento se ha empleado la librería SpaCy. Esta librería soporta la tokenización y el entrenamiento para más de 60 lenguajes, incluyendo el español. Trabaja con modelos neuronales para el etiquetado, *parsing*, reconocimiento de entidades y clasificación de textos, además de otros sistemas de entrenamiento y gestión del flujo de trabajo. La versión empleada en este proyecto es 3.2.4.

SpaCy como librería presenta una *pipeline* como se indica en la **figura 12**, de forma resumida se pueden considerar:

1. *Tokenizer*(tokenizador): divide el texto entrante en tokens, estos tokens pueden ser palabras, signos de puntuación o cualquier espacio en blanco.

2. *Tagger*(etiquetado gramatical): aplica un modelo estadístico al texto de entrada. Analiza la función morfológica de cada token que compone el texto original.
3. *Dependency parsing*(análisis sintáctico): Descubre las dependencias sintácticas de la frase.

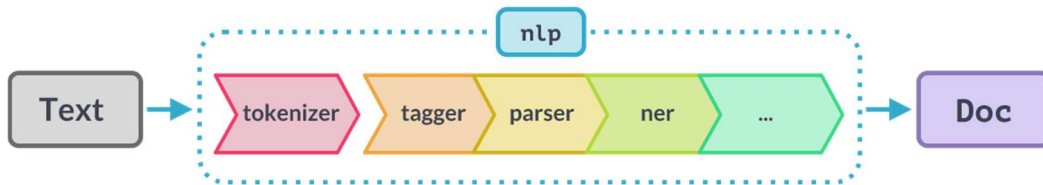


Figura 12. Componentes de la librería SpaCy (Fuente: SpaCy.io)

En la librería Spacy existen dos modelos en español ampliamente utilizados como son “es_core_news_sm” y “es_core_news_md”, la diferencia entre ambos radica en que “sm” se refiere a modelos reducidos, más rapidos pero menos precisos. La referencia “md” son modelos medios entrenados con mayor datos y mayor precision. Por todo ello, se decide emplear como modelo “es_core_news_md” version 3.2.0 [31].

4.3.2 Lematizacion/Stemming

Los procesos de lematizacion y su alternativa llamada *stemming* son otra etapa dentro del preprocesamiento. Con el objetivo de entender estos dos conceptos se podría explicar con que los textos gramaticalmente utilizan diferentes formas de una palabra, así por ejemplo, tenemos palabras como sanar, sana y sanando. Además, existen familias de palabras relacionadas con significados similares, siguiendo con el ejemplo anterior, podríamos considerar sanidad o sanitario. Por ello, ambos procesos como la lematización o el *stemming* tienen como objetivo reducir una palabra a una forma básica común que la relaciona con otras palabras, es decir:

sanar, sana, sanando \Rightarrow san

En un texto tras aplicar la búsqueda de esa raíz podría ser de la forma:

La herida esta en proceso de sanar \Rightarrow La herida esta en proceso de san

Sin embargo, estas dos técnicas se diferencian en su mecanismo. *Stemming* se refiere a un proceso heurístico que habitualmente recorta los finales de las palabras para lograr su objetivo, que resulta en la eliminación de los afijos en la mayor parte de los casos. La lematización, en cambio, utiliza habitualmente un vocabulario y un análisis morfológico de las palabras, que tiene como resultado eliminar los finales de palabra para obtener una base llamada *lema* (en inglés, *lemma*). Habitualmente, estos procesos se realizan mediante extensiones o componentes de un *pipeline*, ya sean comerciales o de código abierto.

Ambos procesos tienen efectos beneficiosos como la reducción del número de palabras que el sistema requiere procesar, reduciendo la dimensión y, por tanto, la complejidad del texto a procesar. Sin embargo, no dejan de tener inconvenientes ya que, en el caso del *stemming*, se podría reducir la precisión (aumentando los falsos positivos), y podría retornar datos relevantes del texto junto con irrelevantes que no tengan nada que ver con los objetivos de la búsqueda. En el caso de la lematización, suele ser un proceso más preciso que el *stemming* ya que tiene en cuenta el significado de la palabra, empleando otros procesos como etiquetado PoS para mejorar la exactitud.

A pesar de ello, el problema de la lematización es que las bibliotecas más empleadas no disponen de módulos para otros idiomas diferentes al inglés, así el paquete NLTK, ampliamente utilizado en PLN, emplea *WorldNet Lemmatizer* que sólo está disponible en inglés. En español es habitual emplear el *stemming* como método alternativo, utilizando una herramienta como *Snowball* que dispone de módulos en español y en otros idiomas. Comparando ambas técnicas, se puede decir que el proceso de *stemming* es más sencillo que la lematización ya que este último requiere de un conocimiento de la lingüística para crear diccionarios que, posteriormente, permitirán al algoritmo buscar la forma apropiada de la palabra.

En nuestro proyecto se ha empleado *Snowball*, aunque se ha implementado como una opción voluntaria en el reconocimiento de entidades. Esto se debe a que su selección ha implicado una pérdida de reconocimiento de entidades, por lo que en el sistema desarrollado no se recomienda. En la **figura 13** se encuentra un ejemplo del reconocimiento de entidades, empleando previamente *stemming*. Para entender el

resultado mostrado, hay que señalar los pasos de *Snowball*. El primer paso de esta técnica es señalar el idioma empleado (en este caso, español). A continuación, se tokeniza el texto y el algoritmo busca la raíz de cada uno de los tokens, por ejemplo, en la **figura 13** “tratamient” de la palabra original “tratamiento” o “fiebr” de la palabra original “fiebre”. Por último, una vez que finaliza este paso, se procede al NER. Como se puede observar en la **figura 13**, los resultados muestran que existen ciertas entidades que no fueron reconocidas y que deberían haberlo sido como “ivermectin” o “aspirin” por lo que refleja un inconveniente conocido del *stemming* donde una derivación excesiva provoca que haya entidades que no se reconozcan.



país tratamient prevent ivermectin ibuprofen aspirin inutil paracetamol medicamento variant control tratamient ivermectin
 ibuprofen aspirin darl inutil paracetamol medicamento antiinflamatori cag trazabil inutil serviri seri tratamient ivermectin
 ibuprofen humild aspirin sirv inutil paracetamol medicamento desobedec dios todopodor histori juzg veng diarre consult
 trai fiebr ceftriaxon dex eritromicin naproxen paracetamol medicamento ibuprofen ivermectin cov seman ayno ibuprofen
 hras aspirin diari vitamin azitromicin ivermectin depend pes dosis tratamient envi mam bestial locatel merced gent loc busc

Figura 13. Ejemplo de reconocimiento de entidades empleando stemming

4.3.3 Stopwords

En cada idioma existen palabras que contienen un menor significado que podrían eliminarse. Estas palabras se conocen como *stopwords* que engloban a los determinantes, verbos auxiliares y pronombres. Su eliminación implica un menor consumo de memoria y cálculos más rápidos. Sin embargo, existen autores que consideran que estas palabras contienen un significado del dominio y que la reducción de dimensionalidad del texto (y complejidad) no compensa la pérdida de significado.

En este proyecto se ha empleado el módulo disponible en español del paquete NLTK, aunque no ha tenido un efecto apreciable en los resultados finales de este sistema.

4.4 Reconocimiento de entidades (NER)

Tras la etapa de limpieza y una vez realizada la tokenización, se procede al NER. Con el objetivo de aumentar la homogeneidad de los tweets y evitar una pérdida de

rendimiento con una lista amplia de medicamentos, se seleccionaron sólo los tweets que involucraban a los 10 medicamentos/principios activos más frecuentes. Los medicamentos/principios activos seleccionados fueron: Ibuprofeno, Rivotril, Aspirina, Clonazepam, Omeprazol, Hidroxicloroquina, Azitromicina, Ivermectina, Remdesivir y Dexametasona.

En el reconocimiento de entidades se emplearon las propiedades de la librería *Spacy* que proporciona un etiquetado de dependencia sintáctica y etiquetado gramatical, añadiéndole elementos propios a su *pipeline*. Debido a la disponibilidad de un diccionario conteniendo los principios activos y medicamentos disponibles en España (obtenidos de fases anteriores), se procedió a la creación de una regla de entidades (*entity ruler*, en inglés) que englobasen estos términos. Mediante esta técnica incluida en la librería *Spacy* el usuario establece una serie de instrucciones para encontrar y etiquetar entidades. Tras realizar esta actividad, se pueden añadir al flujo de trabajo como un nuevo componente. No solo realiza el etiquetado, sino que el sistema desarrollado guarda las entidades en un diccionario para ser utilizado en otras fases, pudiendo elegir la etapa en la que se desea incorporar, aunque por defecto es la última etapa y así es en este proyecto.

Además de los medicamentos/principios activos también se añadió una nueva entidad basada en patrones como son la dosis. Esta entidad se programó mediante un patrón como indica la **figura 14**:

```
patterns_dosis = [{"SHAPE": "dd"}, {"ORTH": "mg"}], [{"SHAPE": "dd"}, {"IS_SPACE": True}, {"ORTH": "mg"}], [{"SHAPE": "ddd"}, {"ORTH": "mg"}], [{"SHAPE": "ddd"}, {"IS_SPACE": True}, {"ORTH": "mg"}], [{"SHAPE": "d"}, {"ORTH": "mg"}], [{"SHAPE": "d"}, {"IS_SPACE": True}, {"ORTH": "mg"}]]
```

Figura 14. Definición de la entidad "dosis"

Con respecto a las entidades relacionadas con síntomas clínicos, se ha empleado la herramienta conocida como QuickUMLS. Esta es una herramienta que emplea un algoritmo no supervisado para extraer entidades biomédicas de textos mediante la comparación de cadenas. Esta herramienta emplea MetamorphoSys que contiene todos

los archivos UMLS. Está disponible en un amplio abanico de idiomas, por lo que ha sido ampliamente utilizado en el ámbito de la detección de entidades biomédicas en textos.

QuickUMLS puede instanciarse como objeto en el que se pueden elegir diferentes configuraciones:

- criterios de *overlapping*: en el caso de que varios conceptos puedan reconocerse como diferentes entidades, se elige la forma de actuación. Se puede seleccionar *score* que elige de entre todas las entidades aquella con mayor cercanía a la palabra (similitud, *similarity* en inglés) o mediante *length* que elige aquella entidad con mayor número de caracteres que coincidan.

- umbral : mínimo valor de similitud (*similarity* en inglés) entre las cadenas. Habitualmente se elige un valor de 0.6-0.7. Un valor más bajo, implicará el incremento de falsos positivos. Un valor más alto, implicará un menor número de entidades detectadas. En nuestro sistema se ha empleado 0.6, ya que se ha comprobado la creación de un gran número de falsos positivos cuando es inferior a éste, así como muchos conceptos eran considerados como múltiples entidades. El hecho de emplear un parámetro de similitud medio-bajo como es 0.6 se debe a la prioridad de detectar cuantos más términos biomédicos mejor, incluso cuando esto suponga la creación de falsos positivos en un número controlable.

- función de similitud: por defecto, se emplea Jaccard como medida de la similitud definida matemáticamente por la siguiente fórmula:

$$J(doc_1, doc_2) = \frac{doc_1 \cap doc_2}{doc_1 \cup doc_2}$$

Una similitud de Jaccard se encuentra en el rango de 0 a 1. Si los dos documentos o palabras tienen un valor de 1, significa que son iguales. En cambio, si es de valor 0, indica que no son nada iguales.

- ventana: número máximo de tokens a considerar para el *matching*, en este proyecto se empleó el número por defecto que fue 5.

Las entidades reconocidas por QuickUMLS fueron etiquetadas como UMLS. Debido a la limpieza previa del texto antes de realizar el NER, algunos de los métodos no se emplearon, como la reducción del texto a minúsculas.

Tras esta etapa, los tweets se extrajeron en base a tres características en el que se indicaban la entidad dosis reconocida (una o varias de ellas), entidad medicamento y entidad UMLS, pudiendo detectarse múltiples ocurrencias de cada una de ellas. Debido al hecho que nuestro objetivo era detectar efectos adversos a medicamentos, se extrajeron todos aquellos mensajes que hubiesen detectado alguna entidad UMLS y una entidad medicamento.

4.5 Extracción de características (*feature extraction*) en el sistema

Como se ha comentado en el planteamiento del problema, se decidió por la aplicación de *BoW* y de modelos distribucionales.

A continuación, se detallan los métodos empleados en *BoW* y los parámetros seleccionados:

- CountVectorizer: método que proporciona una forma fácil y sencilla de *tokenizar* los textos y construir un vocabulario de palabras conocidas, pero también codificar nuevos documentos utilizando el vocabulario creado. Además, tiene la capacidad de ignorar las puntuaciones y convertir las mayúsculas en minúsculas cuando aparezcan en el texto. Los parámetros más importantes incluidos en este método son:

- stop_words: debido a que esta técnica sólo cuenta las ocurrencias de cada palabra, existen palabras muy comunes como 'de' o 'y' que serán características muy importantes del texto pero que aporten poco o nada de significado del texto. El modelo podría mejorarse si no se tienen en cuenta estas palabras.

- ngram: permite mejorar la potencia predictiva en algunos casos ya que clasifica las palabras no como tokens únicos, sino que podrían ser formadas por 2- o 3-

gramas, por ejemplo, en un texto con “tengo infección abdominal”, si seleccionamos 2-gramas, tendría “tengo infección” y “infección abdominal”.

- `min_df`, `max_df`: frecuencias mínimas y máximas de palabras/n-gramas para utilizarse como características.

Con el objetivo de reducir la dimensionalidad de los vectores, se empleó también el método estadístico TF-IDF, que puede utilizarse tanto en BoW como complemento de los modelos distribucionales.

En cuanto a los modelos distribucionales:

- *Word2Vec*: es un modelo que crea vectores de palabras que son representaciones numéricas de características de palabras.

- *FastText*: es un modelo que posiciona las palabras de un documento en un espacio vectorial basado en los n-gramas de las letras que las conforman y en *skip-grams* de palabras para detectar el contexto en que se utilizan.

Los resultados entre las dos han sido similares, aunque el hecho de que el entrenamiento en *FastText* requiere altos requisitos de memoria cuando se trata de *corpus* de tamaño más grandes, ha hecho que se haya preferido emplear *Word2Vec*.

4.6 Clasificación de efectos adversos

Tras el paso anterior, se etiquetaron los datos comparando el medicamento/principio activo con sus potenciales efectos adversos descritos en la ficha técnica. Este paso de etiquetado puede ser opcional, pudiendo analizar textos ya etiquetados aportados por el usuario o dejar que el sistema los etiquete automáticamente. El etiquetado automático se realizó teniendo en cuenta la selección de los 10 medicamentos/principios activos más frecuentes y los efectos adversos indicados en la ficha técnica de cada uno. Si en un mensaje, existía una entidad UMLS que coincidía con un efecto adverso, era clasificado como “P” (positivo), en caso contrario “N” (negativo).

Con el objetivo de evaluar la capacidad que tiene el sistema de predecir los efectos adversos, se emplearon diferentes algoritmos que se evalúan en el siguiente subapartado, aunque finalmente se decidió por emplear regresión logística por obtener los mejores resultados en cuanto a precisión.

4.6.1 Algoritmos considerados para la clasificación

Los clasificadores elegidos fueron: regresión logística, Naïve Bayes multinomial, k-NN, CRF y SVM. Como ya se ha comentado, al tratarse de una clasificación binaria, estos algoritmos permiten realizar la clasificación sin altos costes computacionales.

Sin embargo, hay que destacar que todos estos algoritmos pueden necesitar herramientas adicionales debido a presentar un dataset con datos muy desequilibrados. Cuando se habla de datos desequilibrados se refiere a un problema de clasificación donde las clases no tienen la misma representación, por ejemplo, en nuestro trabajo se podría tener una clasificación en la que en un 70-80% de mensajes no habría un efecto adverso, mientras que en un 20-30% de mensajes sí que habría. Cuando se emplean estos datos desequilibrados para entrenar el modelo, este estará sesgado hacia la clase mayoritaria. Esto puede generar un problema cuando se está interesado en la predicción de la clase minoritaria, como es en el caso de nuestro trabajo, que interesa detectar los efectos adversos. Los enfoques tradicionales para tratar con datos desequilibrados se han basado en técnicas de remuestreo de los datos o modificación de los algoritmos de clasificación. Una de estas técnicas consiste en optimizar los hiperparámetros de un modelo. Esto consiste en la búsqueda del conjunto de argumentos de configuración de un modelo que den como resultado el mejor rendimiento del modelo para un conjunto de datos específico. Durante este trabajo, se han empleado una herramienta GridSearchCV que ha ayudado a afinar los hiperparámetros del modelo, teniendo en cuenta el peso de cada una de las clases.

4.7. Desarrollo de una aplicación

El sistema se desarrolla para utilizarse como aplicación de escritorio con el objetivo de que el usuario pudiese emplear una interfaz que sea fácil de instalar, utilizar y muy

intuitiva. Para ello, se diseñaron diferentes etapas en las que el usuario cargaba un texto y obtenían unos resultados por pantalla y en forma de archivos. Además, el usuario puede realizar el procesamiento a través de una sola etapa, desde el archivo original hasta la evaluación del sistema. Finalmente, en la última etapa obtiene y puede visualizar el clasificador con el texto del mensaje, la presencia o ausencia de efecto adversos contenida en el texto, la presencia o ausencia de efecto adverso predicha y la probabilidad de la predicción. Esta visualización puede almacenarse como un archivo disponible en formato .csv.

Debido al empleo de una aplicación externa como QuickUMLS, la aplicación requiere de ella para una de las etapas, concretamente el reconocimiento de entidades. El resultado dependerá del empleo de los diccionarios de términos médicos (incluido en QuickUMLS), si bien la utilización de los términos UMLS para la detección de términos biomédicos en textos es estándar.

Un ejemplo del resultado final se observa en la **figura 15**:

tweets	real_RAM	predicha_RAM	
solo porque lo dice? anteriormente tambien decian y se contradecian de otros medicamentos. no sera que al caer al credibilidad en las vacunas los grandes laboratorios le ordenaron dar este mensaje?	P	P	0.912
la ivermectina sufre otro revés. la agencia europea de medicamentos desaconseja usarla contra el coronavirus via 137478221905490490 emta desaconseja el uso de ivermectina porque no sirve ni por	P	P	0.968
la cefepim acaba de autorizar el uso de remdesivir para pacientes con covid19. este antiviral es uno de los únicos fármacos en el mundo que ayudan a recortar el tiempo de enfermedad y evitar compl	N	N	0.849
y sigue la cruzada globoprogre contra la ivermectina por cierto el acetaminofen la vitamina c y la aspirina tampoco sirven? ya es como mucho.	N	N	0.001
y mi mamá dice que no puedo vivir a base de ibuprofeno como que no? me alivia el dolor ya fue.	N	N	0.000
dente una rivotril a sicchito o se va desmayar de la ansiedad.	P	P	0.993
la acidez post alcoholica se levaba mejor con la ranitidina que con el omeprazol. data dura.	N	N	0.000
no existe en el mundo ningún fármaco 100% inocuo le recuerdo la aspirina que incluso puede producir alergias o sangrados y el tan amado paracetamol. el beneficio siempre tiene que ser mayor al ries	N	N	0.000

Figura 15. Pantalla con los resultados finales del sistema

5. Metodología de desarrollo y diseño

5.1 Metodología

El desarrollo del sistema ha sido condicionado por la finalidad investigadora de este proyecto, intentando obtener un ejemplo reflejo de los objetivos propuestos. Por ello, de todas las metodologías de desarrollo de software existentes, la más adaptada a este proyecto es la metodología ágil, permitiendo una alta flexibilidad en el desarrollo del propio software. Por un lado, permite adaptar el software a las necesidades que van surgiendo y ,por otro lado, en cada ciclo de desarrollo se van agregando nuevas funcionalidades a la aplicación final, permitiendo añadir pequeñas funcionalidades en lugar de grandes cambios.

Con vistas a obtener un prototipo rápido se ha preferido desarrollar una aplicación de escritorio, fácilmente instalable y con capacidad para trabajar sobre los recursos del usuario tras su instalación. A su vez, se ha preferido el lenguaje Python sobre otros lenguajes de programación ya que permite la creación de *scripts* fáciles de implementar con la ayuda de las librerías disponibles en el campo del Aprendizaje Automático y PLN. A pesar de ello, se ha intentado mantener una alta cohesión y bajo acoplamiento en el diseño del programa, reduciendo la interacción entre diferentes módulos y permitiendo que cada módulo codificado estuviese compuesto por elementos altamente relacionados.

5.2 Herramientas utilizadas

Python ha sido el lenguaje de programación empleado en este proyecto, en concreto la versión 3.9.7. Considerando el objetivo principal investigador del proyecto y las opciones valoradas en cuanto al lenguaje de programación, se ha optado por Python frente a Java. Aunque la discusión de qué lenguaje es mejor está fuera del ámbito de este proyecto, se ha seleccionado Python debido a su sencillez, la obtención de resultados rápidos con

diferentes estrategias y la amplia disponibilidad de librerías relacionadas con aprendizaje automático y PLN.

Las librerías empleadas en este proyecto han sido:

- NumPy: librería especializada en el cálculo numérico y el análisis de datos, especialmente para un gran volumen de datos.
- Pandas: librería especializada en el manejo y análisis de estructuras de datos.
- SpaCy: librería que permite construir aplicaciones de PLN, proporcionando modelos preentrenados de diferentes idiomas. SpaCy es un software comercial de código abierto bajo licencia de Massachusetts Institute of Technology (MIT).
- Natural Language Tool Kit (NLTK): librería que ofrece una variedad de capacidades de manipulación de cadenas. Contiene un gran repositorio de plantillas con modelos de análisis de texto, gramáticas basadas en características y ricos recursos léxicos para construir un modelo de lenguaje completo.
- Pyinstaller: librería que transforma todos tus archivos Python en un solo paquete. Es ideal para distribuir la aplicación desarrollada ya que el usuario sólo tiene que instalar la aplicación y no requiere la instalación de más módulos.
- InnoSetup: sistema de instalación en un entorno Windows.

A lo largo de las etapas del proyecto cuando se empleen por primera vez cada librería se indicará la versión utilizada.

5.3 Planificación

En la planificación del trabajo se ha tenido en cuenta el tiempo dedicado a la obtención de los tweets en redes sociales y la implementación diccionarios externos o la elaboración de diccionarios internos como parte de la fase de codificación.

La planificación del trabajo se puede observar en la **figura 16** mediante el diagrama de Gantt:

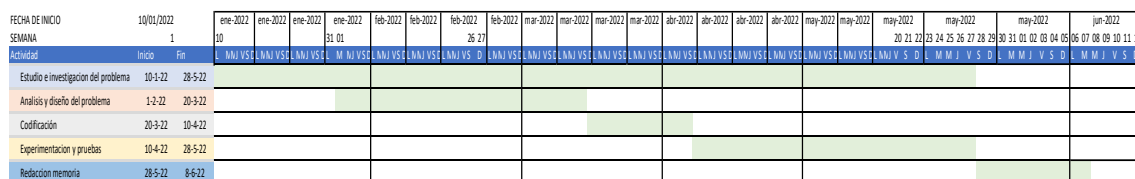


Figura 16. Diagrama de Gantt del proyecto

Como se ha comentado anteriormente, la metodología ágil permite la incorporación de nuevas funcionalidades a medida que se avanza en el desarrollo del proyecto. En este caso, se puede observar en el diagrama de Gantt como el estudio y la investigación del problema ha continuado durante todo el tiempo dedicado, solapándose con otras etapas. El trabajo en total ha llevado sumando todas las etapas 520 horas.

La distribución de los esfuerzos se indica en la **tabla 8**:

Estudio e investigación (10%)	Análisis y diseño (35%)	Codificación (15%)	Experimentación y pruebas (35%)	Redacción de memoria (5%)	Total
1 persona * 6,5 jornadas * 8 horas = 52 horas hombre	1 persona * 22,75 jornadas * 8 horas = 182 horas hombre	1 persona * 9,75 jornadas * 8 horas = 78 horas hombre	1 persona * 22,75 jornadas * 8 horas = 182 horas hombre	1 persona * 3,25 jornadas * 8 horas = 26 horas hombre	520 horas hombre

Tabla 8. Distribución del esfuerzo

5.4 Requisitos del sistema

5.4.1 Requisitos funcionales

Los requisitos funcionales del sistema incluyen:

- El sistema aceptará como datos de entrada archivos en formato .csv obtenidos del minado de la red social Twitter a través de la interfaz gráfica de usuario (GUI) de la aplicación.

- El sistema podrá realizar todo el proceso de forma continua o dividido en etapas. El sistema mostrará opcionalmente el resultado de cada etapa: limpieza, preprocesamiento, NER y clasificación. A su vez, en cada etapa se podrá obtener un archivo independiente para proseguir con el resto de las etapas de forma continua o dilatada en el tiempo. La etapa de reconocimiento de entidades también tendrá disponible el acceso a un archivo HTML que muestra gráficamente todas aquellas entidades reconocidas por el sistema, además de la clase de entidad. El usuario podrá emplear tweets anotados y no anotados, según su preferencia. El resultado final de la última etapa de clasificación serán unos gráficos que representan el número total de RAM presentes en el documento, la matriz de confusión y la curva ROC (Receiver Operating Characteristic o Característica Operativa del Receptor) y un archivo en formato .csv que muestre el resultado de la etapa de clasificación.

- La GUI validará que en cualquier momento se cumplan los requisitos de entrada de archivos, además debido al empleo de un sistema de extracción de entidades como QuickUMLS, el sistema comprobará y exigirá la entrada del directorio donde se encuentra este sistema durante la etapa de NER. En caso de que no se cumplan, enviará un mensaje de advertencia al usuario.

- Debido a las limitaciones en cuanto a rendimiento de la capacidad de reconocer entidades en el caso de que los archivos de texto superen los 900000 caracteres, el sistema propondrá al usuario dividir el texto en estudio en partes iguales. Finalmente, tras realizar el NER, el usuario podrá unir los resultados obtenidos para hacer un archivo

final y someterlo a la etapa de clasificación. Si el usuario ha decidido realizar el procesamiento en un solo paso, el sistema hará todo este paso automáticamente.

- En cuanto a los requisitos de hardware o sistema operativo, la aplicación debe poder utilizarse en un sistema Windows. Es aconsejable disponer de una memoria RAM superior a 8 gb por motivos de rendimiento. Los archivos resultantes de la aplicación no requieren ningún software especial , ya que son textos planos o en formato .csv o gráficos en formato .png o página web en formato HTML, éste último se puede abrir según el navegador por defecto del usuario.

5.4.2 Requisitos no funcionales

- El sistema cuenta con un manual de usuario y un video explicativo.
- La aplicación genera mensajes de error y advertencias para orientar al usuario en el uso y navegación de la aplicación.

5.5 Casos de uso

Los casos de uso incluidos en esta memoria se refieren al empleo de la aplicación por parte del usuario. Se describe el principal caso de uso detallado mediante un flujo básico y los flujos alternativos, de acuerdo con la **tabla 9**:

Caso de uso	Obtener un clasificador de efectos adversos en textos de la red social Twitter.
Actor	usuario
Flujo básico	El usuario posee un texto con mensajes de la red social Twitter y entra en el sistema para detectar RAM
Flujo alternativo 1	El usuario posee un archivo obtenido de etapas previas a la clasificación y desea clasificarlo
Flujo alternativo 2	El usuario presenta un archivo con mayor número de caracteres permitido

Tabla 9. Caso de uso

A continuación, se describe el flujo básico en la **tabla 10**:

Flujo básico	Obtener un clasificador de efectos adversos en textos de la red social Twitter.
Descripción	El usuario dispone de un archivo resultante del minado de tweets. Se trata del principal escenario de éxito
1	El usuario ejecuta la aplicación
2	El usuario selecciona la etapa inicial de Limpieza
3	El usuario selecciona el archivo inicial y procede a la limpieza del texto
4	El sistema muestra el resultado de la limpieza, informa de la creación de un nuevo archivo con los mensajes limpios para su posterior uso en el reconocimiento de entidades.
5	El usuario selecciona reconocimiento de entidades
6	El usuario selecciona el directorio donde se encuentra el extractor de entidades QuickUMLS y el archivo limpio obtenida en la etapa de limpieza
7	El sistema muestra el resultado de las entidades reconocidas. El usuario puede obtener el archivo con las entidades reconocidas y su clase, así como una página HTML para visualizarlo gráficamente.
8	El usuario selecciona la clasificación de los textos
9	El usuario selecciona el texto de la etapa de reconocimiento y el sistema procede a la clasificación de textos. El sistema crea un archivo final con el mensaje de texto y el resultado de la clasificación, así como gráficos resultantes de la clasificación en formato .png.
10	El usuario puede repetir el proceso con más textos volviendo a cada una de las etapas en función de la disponibilidad y formato de sus textos.

Tabla 10. Flujo básico del caso de uso

A continuación, se describe el flujo alternativo 1 en la **tabla 11**:

Flujo alternativo 1	Carga de archivo
2	El usuario dispone de un archivo resultante de una etapa posterior de la ejecución del programa y no necesita limpiarlo.
2A1	El usuario selecciona la etapa de reconocimiento de entidades para procesarlo.
2B1	El usuario selecciona la etapa de clasificación para procesarlo.

Tabla 11. Flujo alternativo 1

A continuación, se describe el flujo alternativo 2 en la **tabla 12**:

Flujo alternativo 2	Carga de archivo
6	El usuario presenta un archivo con mayor número de caracteres permitido por el reconocimiento de entidades. El sistema informa de ello.
6A1	El usuario selecciona un archivo de menor número de caracteres
6B1	El usuario selecciona Herramientas para dividir el texto
6B2	El usuario selecciona el archivo a dividir
6B3	El sistema muestra el resultado de la división e informa de la identificación del texto generado
6B4	El usuario vuelve a la etapa de reconocimiento de entidades

Tabla 12. Flujo alternativo 2

5.6 Presupuesto y cálculo de costes

5.6.1 Descripción del proyecto

Autor: Roi Arias Rico

Departamento: Lenguajes y Sistemas Informáticos

Título: Sistema para la detección de efectos adversos a medicamentos en textos biomédicos en español.

Duración: Inicio en febrero 2021 y fin en junio 2022.

5.6.2 Cálculo de costes

· Costes de personal

El proyecto fue realizado por una única persona. En la **tabla 13** se indican las etapas y el coste de horas:

Etapas	Coste/hora	Horas totales	Coste total
Análisis y diseño	€60	182	€10920
Codificación		78	€4680
Experimentación y pruebas		182	€10920
Documentación		26	€1560
TOTAL		520	€28080

Tabla 13. Costes de personal

· Costes de equipamiento

Para el desarrollo de este proyecto se empleó un único ordenador, tanto para la codificación como para las pruebas pertinentes, de acuerdo a la **tabla 14**:

Concepto	Unidad	Coste
Ordenador portátil	1	€ 1000
Coste total		€ 1000

Tabla 14. Costes de equipamiento

- Costes de software

El coste del sistema operativo se incluye en los costes de hardware. En cuanto al software utilizado propiamente para la programación de este proyecto ha sido mediante el empleo de software libre en la **tabla 15**

Concepto	Unidad	Coste
PyCharm Community version	1	€ 0
Unified Medical Language System (UMLS)	1	€ 0
Anaconda3 2021-Distribution	1	€ 0
Coste total		€ 0

Tabla 15. Costes de software

- Otros costes

El coste de material fungible como papel, tinta de impresora, costes de encuadernación y otras que no se han considerado previamente se indica en la **tabla 16**:

Concepto	Unidad	Coste
Papel	1	€ 5
Tinta de impresora	1	€ 25
Costes de encuadernación	1	€ 3
Otros gastos	1	€ 8
Coste total		€ 41

Tabla 16. Costes de material fungible y otros gastos

5.6.3 Presupuesto

El presupuesto se indica en la **tabla 17**, de acuerdo con todos los costes anteriormente señalados:

Concepto	Coste
Costes de personal	€ 28080
Costes de equipamiento	€ 1000
Costes de software	€ 0
Otros costes	€ 41
Coste total	€ 29121

Tabla 17. Presupuesto del proyecto

No se ha tenido en cuenta ni los costes indirectos ni los impuestos resultantes de la actividad al ser un proyecto de investigación.

6. Evaluación del sistema

6.1 Métricas para la extracción de información

Con el objetivo de evaluar los resultados obtenidos, es muy importante seleccionar métricas de evaluación claras, reproducibles y fácilmente comprensibles. Antes de presentar las métricas seleccionadas en este proyecto, se debe definir una estructura ampliamente utilizada en el ámbito de estudio como es la matriz de confusión (también llamada tabla de contingencia). Esta matriz se encuentra dividida en cuatro categorías:

- Verdaderos Positivos (VP): ejemplos correctamente etiquetados como positivos
- Falsos Positivos (FP): ejemplos negativos incorrectamente etiquetados como positivos.
- Verdaderos Negativos (VN): ejemplos negativos correctamente etiquetados como negativos.
- Falsos Negativos (FN): ejemplos positivos incorrectamente etiquetados como negativos.

A continuación, se presenta la matriz de confusión en la **figura 17**:

		Predicción	
		Positivos	Negativos
Observación	Positivos	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Figura 17. Matriz de confusión

La matriz de confusión presenta dos columnas que indican el número de predicciones de cada clase y dos filas que representan las observaciones de cada clase. De tal forma, que cada elemento vendrá determinado por una columna que se predice y una fila que se observa.

A continuación, se definen las métricas empleando los valores de los elementos de la matriz de confusión:

·Sensibilidad (o exhaustividad/*recall*): Mide la fracción de ejemplos positivos que son correctamente etiquetados.

$$\text{Sensibilidad} = \text{recall} = \frac{VP}{VP+FN}$$

·Precisión: Mide la proporción de ejemplos predichos que son realmente positivos:

$$\text{Precisión} = \frac{VP}{VP+F}$$

·valor F: es un parámetro definido por las dos medidas anteriores, ya que es la media armónica entre los valores de *recall* y de precisión, donde el parámetro β indica el peso relativo de precisión respecto al valor de *recall*. En caso de que $\beta=1$, se está dando la misma ponderación a precisión que a *recall*, obteniendo el valor del parámetro F1. La fórmula, donde P representa a precisión y R a *recall*, es la siguiente:

$$F(\beta) = \frac{(1+\beta^2)PR}{\beta^2P+R}$$

Un sistema ideal de extracción de información es aquél en el que no hubiera ni falsos positivos o falsos negativos. Sin embargo, esto es difícilmente conseguible en sistemas reales por lo que se observa que la precisión y el *recall* son parámetros antagónicos, cuando uno aumenta, el otro se reduce y viceversa.

La selección de estos parámetros se ha basado en la literatura disponible ya que la precisión, el *recall* y f1 son ampliamente utilizados para comparar los sistemas. Adicionalmente, con el objetivo de evaluar diferentes algoritmos planteados para el proyecto, se ha empleado el término exactitud definida como :

$$\text{Exactitud} = (VP + VN) / (VP+FP+VN+FN)$$

6.2 Resultados

6.2.1 Corpus de prueba

El *corpus* de prueba consistió en la extracción de tweets que mencionaban cualquier principio activo o medicamento comercializado en España en el momento del minado. El número total de tweets correspondió a 1016753, que se filtraron para los diez medicamentos/principios activos más frecuentes, quedando finalmente en un *corpus* que contenía 216105 tweets (ejemplo de archivo: “./data/output.csv”)

6.2.2 Limpieza

El texto, conteniendo nuestras referencias a los medicamentos y principios activos, se somete a un proceso de limpieza. De forma que, en el ejemplo de la **figura 18**, el archivo utilizado anteriormente presentaba 5437940 caracteres y acaba con 4957280 caracteres. Finalmente, se realiza un análisis de impurezas por si ha quedado algún carácter que se debería haber eliminado o no se ha reconocido, en este caso, 0.009%. Finalmente, genera dos archivos e indica el directorio donde se encuentran, el archivo de texto se utilizará para las fases posteriores. El resultado final del proceso de limpieza aparece reflejado en la **figura 18**:

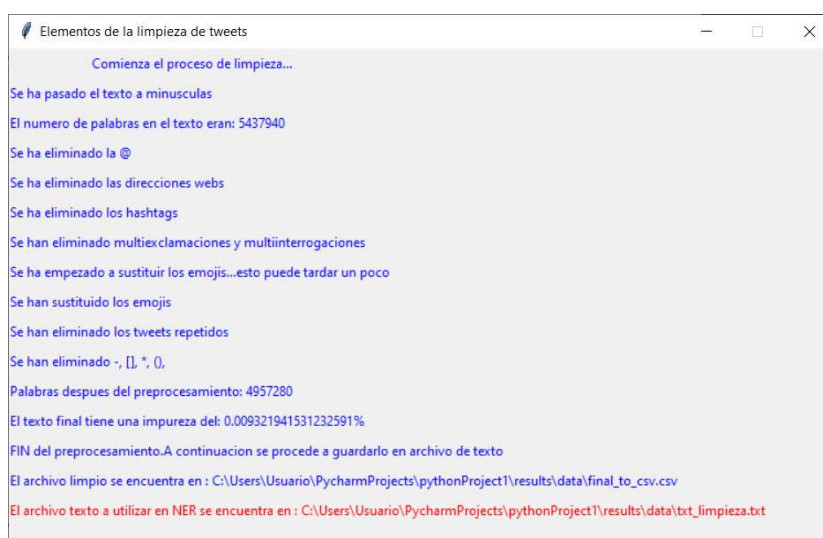


Figura 18. Resultados de la limpieza del corpus

6.2.2 Reconocimiento de entidades

Una vez obtenido el fichero de limpieza de pasos anteriores se procede a la aplicación de los diccionarios y la herramienta QuickUMLS para el reconocimiento de entidades. Los resultados se guardan en un archivo HTML llamado NER_displacy.html en la carpeta data, que se muestra en la **figura 19**:

vengo por que tengo **diarrea UMLS** aja fui a consultar antes traia **fiebre UMLS** de 37o y me dieron 3 dias de **ceftriaxona medicamento** con dexta 3 dias de eritromicina **naproxeno medicamento** con **paracetamol medicamento** **ibuprofeno medicamento** e **ivermectina medicamento** por si era covid por una semana ayno. **ibuprofeno medicamento** de **400mg dosis** cada 8 hrs **aspirina medicamento** **81mg dosis** 1 diaria **vitamina c UMLS** **azitromicina medicamento** **ivermectina medicamento** dependiendo del peso 1 hoy y otra dosis en 48 horas. ese fue el tratamiento q nos enviaron a mi mama y a mi.

Figura 19. Extracto de un texto tras NER

A la vez que se genera un archivo conteniendo cuatro columnas que representan el texto original del tweet (columna texto), la entidad “dosis” (si no hay indicación se indica como no), la entidad “medicamento” y la entidad “UMLS”, como indica la **figura 20**:

texto	dosis	medicamento	UMLS
a un amigo le dieron diagnostico de inicio de infeccion de amigdalas y el tratamiento era ivermectina azitromicina e ibuprofeno. de verdad si no saben no inventen payasadas. ponganse a leer.	no	'azitromicina', 'ibuprofeno', 'ivermectina'	'infeccion'

Figura 20. Resultados del reconocimienro de entidades

Existe una opción de *stemming* en el reconocimiento de entidades, pero en el proyecto no dio buenos resultados porque el número de entidades biomédicas reconocidas se redujo.

6.2.3 Clasificación de textos

Finalmente, con el archivo obtenido en la fase anterior se procede a la clasificación de textos. El número de tweets que han reconocido las entidades biomédicas y medicamentos/principios activos han sido 13098, esto corresponde al 6.06% de los tweets originales filtrados por los 10 medicamentos/principios activos más numerosos. Se realiza una detección automática en función de los fármacos diana y los términos. Cuando se clasifican los efectos adversos se obtiene la siguiente clasificación de los textos en función de si contiene o no efectos adversos, como aparece reflejada en la **figura 21**:

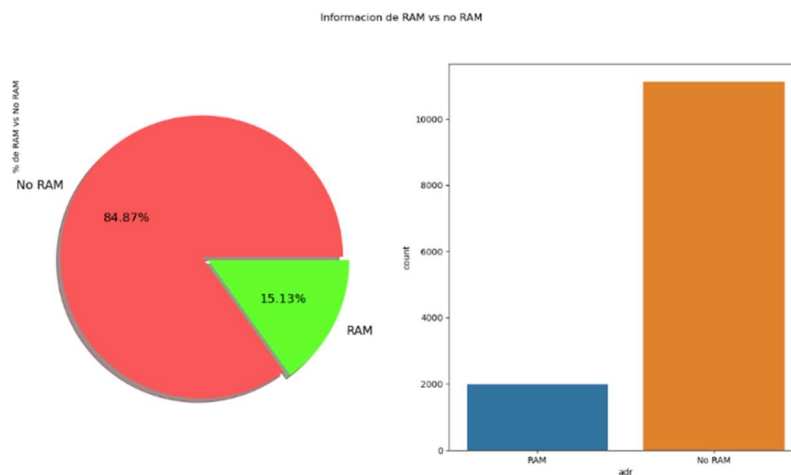


Figura 21. Número de efectos adversos detectados en los tweets.

Posteriormente, se emplean los siguientes algoritmos:

- Naive Bayes Multinomial
- Regresión logística
- k-NN
- Linear SVM
- CRF

Durante toda la etapa de clasificación, el *corpus* se dividió en dos partes: el conjunto de entrenamiento y el de test. La proporción se mantuvo durante todos los experimentos con la proporción 70:30 (entrenamiento : test).

En un primer momento de la experimentación, se optó por experimentar con todos ellos para seleccionar aquel algoritmo con mejores prestaciones para posteriormente elegir uno de ellos. Los valores de exactitud y el tiempo necesario para entrenar a un conjunto están indicados en la **tabla 18**:

Algoritmo	Exactitud sobre el conjunto prueba	Tiempo para entrenamiento
Regresión Logística	0.95598	1.12 s
Linear SVM	0.958015	7.86 s
k-NN	0.907125	0.01 s
Naive Bayes Multinomial	0.922901	0.02 s
CRF	0.940712	1.07 s

Tabla 18. Resultados de la precisión de los algoritmos

Como se puede observar en la **tabla 18**, a pesar de los buenos resultados que se obtienen con SVM linear, tiene un gran problema de rendimiento como se indicó en la presentación de las características de los algoritmos. Naive Bayes y k-NN obtuvieron un gran rendimiento, pero se pierde exactitud.

Midiendo la exactitud en el conjunto de entrenamiento, el tiempo de entrenamiento y la exactitud en el conjunto de test, la precisión, *recall* y F1, se obtienen los resultados de la **tabla 19**:

Algoritmo	Exactitud en el conjunto de entrenamiento	Tiempo de entrenamiento	Exactitud en el conjunto test	Precisión	Recall	F1
Regresión Logística	0.999346	1.185284 s	0.955980	91%	80%	85.21%
Linear SVM	0.989747	7.905696 s	0.958015	93%	80%	85.90%
k-NN	0.925502	0.005614 s	0.907125	95%	44%	60.54%
Naïve Bayes Multinomial	0.996401	0.014771 s	0.877354	96%	25%	39%
CRF	0.997273	1.093027 s	0.934860	96%	59%	72%

Tabla 19. Resultados de los algoritmos

A pesar de sus buenos resultados, el mal rendimiento de SVM la descarta ya que el tiempo empleado es excesivo con respecto al resto de algoritmos con niveles similares de exactitud.

Una vez realizados estos análisis, parece probable emplear Regresión Logística en términos de rendimiento y exactitud, aunque queda establecer la posibilidad de *overfitting* en los algoritmos implicados. El *overfitting* se podría deducir de aquellos casos en los que existe una diferencia ostensible entre la exactitud del conjunto de entrenamiento o del test. Este error consiste en que el algoritmo empleado tiene en cuenta el ruido aleatorio más que el propio patrón. La mejor forma de evitarlo es emplear la validación cruzada, así se dividió el conjunto de datos en tres partes: 2/3 se empleó para el entrenamiento y 1/3 para el conjunto test y se hará el proceso de testing 3 veces.

Los resultados de la validación cruzada de acuerdo con la métrica `cross_val_score` (incluida en el paquete *scikit-learn*) se muestran en la **tabla 20**:

Algoritmo	Cross val score
Regresión Logística	0.953098
CRF	0.921357
k-NN	0.902705
Naive Bayes Multinomial	0.865620

Tabla 20. Resultados de la validación cruzada

Los resultados de esta validación cruzada corroboran el empleo de Regresión Logística como algoritmo elegido para la clasificación de los textos. Una vez seleccionado el algoritmo se ha empleado otro método que permite corregir los errores que se podrían producir por la presencia de un claro desequilibrio entre la presencia de efectos adversos y su ausencia, en nuestro caso, hay un 15.13% de efectos adversos detectados mientras que 84.87% de ausencia de efectos adversos. Con el objetivo de afinar los hiperparámetros del modelo se ha empleado *GridSearchCV*. Mediante esta herramienta se emplean todos los hiperparámetros especificados y evalúa cada una de las combinaciones de los hiperparámetros hasta que selecciona la que obtiene una mejor evaluación. Además, se realiza una validación cruzada.

En el caso que nos ocupa, se seleccionaron como argumentos en el método *GridSearchCV*, el empleo de regresión logística (RL) como estimador y una validación, como en pasos anteriores, igual a 3. A su vez, dentro de los parámetros, se indica el valor desequilibrado de los datos con respecto a la presencia o ausencia de efectos adversos. En este caso, se indicó que la ausencia de efectos adversos existía en el 80% de los mensajes y su presencia en el 20% restante.

Cuando se emplearon los resultados de la clasificación, se muestra que el modelo presenta una precisión del 91%, *recall* del 80% y f1:85.21%, como se indica en la **tabla 21**:

	precisión	recall	f1
Modelo final (RL)	91%	80%	85.21%

Tabla 21. Resultados de precisión, recall y f1 del modelo final

Finalmente, para obtener una visualización de la relación precisión y *recall*, se emplea la curva ROC que permite la representación gráfica de la sensibilidad frente a la especificidad para un sistema clasificador binario según se varía el umbral de discriminación. En el proyecto el valor de ROC fue de 0.9783476623805226.

Empleando regresión logística, la matriz de confusión que se obtuvo fue la indicada en la **figura 22**:



Figura 22. Matriz de confusión obtenida en la regresión logística

Los valores de la matriz de confusión de la **figura 22** se visualizan en la **tabla 22**:

		Predicción	
		RAM	No RAM
Observación	RAM	3247	52
	No RAM	124	507

Tabla 22. Matriz de confusión con regresión logística (modelo final)

A continuación, se presenta la curva ROC en la **figura 23**:

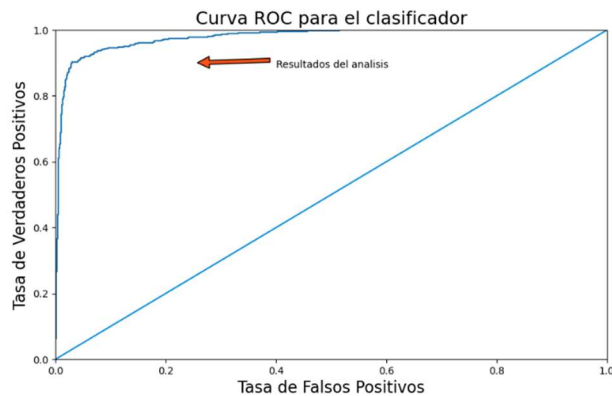


Figura 23. Curva ROC resultado de la Regresión Logística

Los valores de las curvas ROC obtenidas con otros algoritmos se muestran en la **tabla 23**:

Algoritmo	ROC
Regresión Logística	0.978
Linear SVM	0.966
k-NN	0.846
Naive Bayes Multinomial	0.803
CRF	0.977

Tabla 23. Curvas ROC de algoritmos estudiados

La Regresión Logística se aplicó utilizando vectores obtenidos mediante TF-IDF, cuando se planteó el empleo de *Word2Vec* combinado con TF-IDF los resultados obtenidos fueron peores que cuando se empleó TF-IDF. En el caso de *Word2Vec* se empleó un tamaño de vector de 50, con el parámetro de ignorar palabras que tengan una frecuencia menor de 5 y con una distancia máxima entre la actual palabra y la predicha de 5. Una posible explicación es que *Word2Vec* podría haber hecho *overfitting* con respecto a TF-IDF, creando falsos modelos e innecesarios. En el caso de *FastText* los resultados fueron similares como se puede observar en la **tabla 24**. La configuración

seleccionada de *FastText* incluyó un tamaño de vector de 50, con una distancia máxima entre la actual palabra y la predicha de 5, ignorando todas aquellas palabras que tengan una frecuencia menor de 5 y número de iteraciones sobre el *corpus*.

En la **tabla 24** se comparan los datos de empleando TF-IDF o TF-IDF combinada con Word2Vec o FastText:

	precisión	recall	F1
TF-IDF	91%	80%	85%
TF-IDF+Word2Vec	76%	82%	78%
TF-IDF+FastText	82%	84%	78%

Tabla 24. Comparativa de resultados TF-IDF vs combinada con modelos distribucionales

Por último, los resultados se almacenan en un archivo .csv con el texto original, el valor de RAM real, el valor de RAM predicha y la probabilidad de la predicción.

7. Conclusiones

7.1 Conclusiones

En este proyecto se ha pretendido establecer un sistema de detección de efectos adversos de medicamentos en el que a partir de mensajes de usuarios de la red social Twitter pudiera detectarse la relación entre medicamentos y efectos adversos.

Como resultado de este trabajo, el sistema desarrollado ha tenido unos buenos resultados. El valor de F1 fue del 0.85, con una precisión y recall del 0.91 y 0.80, respectivamente. Estos valores se asemejan a otros obtenidos y consultados en la bibliografía. Aunque la detección de términos médicos se ha empleado un algoritmo no supervisado mediante el sistema QuickUMLS, la clasificación se ha realizado mediante un algoritmo supervisado. Aunque se han comparado varios algoritmos, el mejor resultado ha estado presente con Regresión Logística, con resultados óptimos en cuanto a métricas y rendimiento.

Estos buenos resultados se basan en varios aspectos. Por un lado, la limpieza de textos evita los múltiples errores sintácticos que se producen al escribir mensajes en redes sociales, eliminación de tweets duplicados o simplemente publicidad. Por otro lado, la disponibilidad de diccionarios tanto de términos médicos como de medicamentos es fundamental para realizar esta tarea. Se debe hacer hincapié respecto a los diccionarios de términos médicos ya que en nuestro estudio se ha empleado el incluido en el Metathesaurus del NIH por lo que el usuario cuando escribe en una red social podrá tener infinitas entidades para referirse a un determinado efecto adverso que no esté incluida en esa base de datos. Debido a esto tener un diccionario de términos médicos que agrupen todas estas posibilidades lo hacen en cierta forma irreal.

A pesar de los buenos resultados, el sistema puede tener un mejor rendimiento. Gran parte de los tweets obtenidos tuvieron que desecharse ante la imposibilidad de detectar efectos adversos. El resultado final del sistema sólo detectó efectos adversos en el 15%

de los tweets en los que se reconoció una entidad de términos médicos, siendo habitual estos resultados ya que existen autores que obtuvieron que uno de cada 9 tweets [6].

7.2 Posibles mejoras y líneas futuras de desarrollo

El principal objetivo de futuras mejoras debería ser aquellos aspectos en los que peor se desenvuelve el sistema creado. A nivel de arquitectura del sistema, se debería poder contar con un entorno integrado en el que tanto el diccionario de medicamentos como de términos médicos fuesen parte del sistema y no depender de fuentes externas como pueden ser el CIMA en el caso de medicamentos o de MedDRA para los efectos adversos. La idea se basa en que, a mayor nivel de integración, mayor rendimiento se obtenga.

A su vez, el problema en este tipo de textos se basa en el lenguaje coloquial o expresiones que podrían incorporarse mediante la creación de un lexicon que las agrupa para favorecer la obtención de mejores resultados en la detección de efectos adversos.

Otro aspecto podría ser la introducción de mejoras en la detección de las variaciones léxicas. La utilización de librerías destinadas a su utilización en textos de idioma inglés y luego adaptadas al español o directamente no adaptadas, representa un gran obstáculo en la mejora de cualquier sistema. Por ejemplo, la lematización representa un arma poderosa para mejorar la detección de términos, sin embargo, en una gran librería empleada en PLN como NLTK no está presente en español, teniendo que recurrir a otros recursos. Eso implica que, por ejemplo, una palabra como *ansiedad* pudiera reconocer variaciones léxicas relacionadas como *ansioso*, *me da ansias*, *ansiolítico* y , de esta forma, ya no dar falsos negativos en la detección.

Por otro lado, aunque los resultados son óptimos en cuanto a las métricas para el algoritmo empleado, se hace necesario el empleo de otro tipo de algoritmos para mejorar el rendimiento. En textos de tamaño grande mayor de 900000 caracteres, existe una imposibilidad de reconocimiento de entidades para equipos más modestos en recursos. La rapidez en la detección adversos puede llegar a ser un factor limitante cuando se intentan clasificar textos que contienen más de 10 mil tweets.

Otras posibilidades serían la inclusión de más entidades, como pueden ser dosis o frecuencia o incluso duración de tratamiento que podrían complementar al sistema y abordar la posibilidad de vincular todas estas entidades, a pesar de que aumentaría la complejidad y las necesidades computacionales del sistema.

Por último, el sistema se ha implementado como una aplicación de escritorio que puede realizar perfectamente su tarea como sistema, aunque podría ser una limitación en el caso de querer implementarla en un entorno más dinámico o con más recursos humanos. Otro tipo de mejoras incluirían la usabilidad, la accesibilidad o la integración de la extracción de tweets en la propia aplicación. De esta forma, el sistema puede hacer en casi tiempo real, la obtención de textos en la primera etapa y la detección de efectos adversos en la etapa final.

Bibliografía

- [1] AEMPS. "Buenas Prácticas de Farmacovigilancia del Sistema Español de Farmacovigilancia de medicamentos de uso humano."
https://www.aemps.gob.es/vigilancia/medicamentosUsoHumano/docs/BPFV-SEFV_octubre-2008.pdf (accessed 2022/02/12, 2022).
- [2] C. A. Bond and C. L. Raehl, "Adverse drug reactions in United States hospitals," *Pharmacotherapy*, vol. 26, no. 5, pp. 601-8, May 2006, doi: 10.1592/phco.26.5.601.
- [3] M. McClellan, "Drug safety reform at the FDA--pendulum swing or systematic improvement?," *N Engl J Med*, vol. 356, no. 17, pp. 1700-2, Apr 26 2007, doi: 10.1056/NEJMp078057.
- [4] M. D. Rawlins, "Pharmacovigilance: paradise lost, regained or postponed? The William Withering Lecture 1994," *J R Coll Physicians Lond*, vol. 29, no. 1, pp. 41-9, Jan-Feb 1995. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pubmed/7738878>.
- [5] L. Hazell and S. A. Shakir, "Under-reporting of adverse drug reactions : a systematic review," *Drug Saf*, vol. 29, no. 5, pp. 385-96, 2006, doi: 10.2165/00002018-200629050-00003.
- [6] A. Sarker *et al.*, "Utilizing social media data for pharmacovigilance: A review," *J Biomed Inform*, vol. 54, pp. 202-12, Apr 2015, doi: 10.1016/j.jbi.2015.02.004.
- [7] R. Harpaz, W. DuMouchel, N. H. Shah, D. Madigan, P. Ryan, and C. Friedman, "Novel data-mining methodologies for adverse drug event discovery and analysis," *Clin Pharmacol Ther*, vol. 91, no. 6, pp. 1010-21, Jun 2012, doi: 10.1038/clpt.2012.50.
- [8] A. Patki, A. Sarker, P. Pimpalkhute, A. Nikfarjam, and R. Ginn, "Mining adverse drug reaction signals from social media: going beyond extraction," in *Proceedings of BioLinkSig*, 2014.
- [9] S. Yeleswarapu, A. Rao, T. Joseph, V. G. Saipradeep, and R. Srinivasan, "A pipeline to extract drug-adverse event pairs from multiple data sources," *BMC Med Inform Decis Mak*, vol. 14, p. 13, Feb 24 2014, doi: 10.1186/1472-6947-14-13.
- [10] M. Pirmohamed *et al.*, "Adverse drug reactions as cause of admission to hospital: prospective analysis of 18 820 patients," *BMJ*, vol. 329, no. 7456, pp. 15-9, Jul 3 2004, doi: 10.1136/bmj.329.7456.15.
- [11] A. Nikfarjam *et al.*, "Early Detection of Adverse Drug Reactions in Social Health Networks: A Natural Language Processing Pipeline for Signal Detection," *JMIR Public Health Surveill*, vol. 5, no. 2, p. e11264, Jun 3 2019, doi: 10.2196/11264.
- [12] A. Benton *et al.*, "Identifying potential adverse effects using the web: a new approach to medical hypothesis generation," (in eng), *Journal of biomedical informatics*, vol. 44, no. 6, pp. 989-996, 2011, doi: 10.1016/j.jbi.2011.07.005.
- [13] R. Xu and Q. Wang, "Large-scale extraction of accurate drug-disease treatment pairs from biomedical literature for drug repurposing," *BMC Bioinformatics*, vol. 14, p. 181, Jun 6 2013, doi: 10.1186/1471-2105-14-181.
- [14] H. Gurulingappa, A. Mateen-Rajput, and L. Toldo, "Extraction of potential adverse drug events from medical case reports," *J Biomed Semantics*, vol. 3, no. 1, p. 15, Dec 20 2012, doi: 10.1186/2041-1480-3-15.
- [15] S. Sohn, J. P. Kocher, C. G. Chute, and G. K. Savova, "Drug side effect extraction from clinical narratives of psychiatry and psychology patients," *J Am Med Inform Assoc*, vol. 18 Suppl 1, pp. i144-9, Dec 2011, doi: 10.1136/amiajnl-2011-000351.
- [16] S. Tuarob, C. S. Tucker, M. Salathe, and N. Ram, "An ensemble heterogeneous classification methodology for discovering health-related knowledge in social media

- messages," *J Biomed Inform*, vol. 49, pp. 255-68, Jun 2014, doi: 10.1016/j.jbi.2014.03.005.
- [17] R. Ginn, P. Pimpalkhute, A. Nikfarjam, and A. Patki, "Mining Twitter for adverse drug reaction mentions: a corpus and classification benchmark.," in *Proceedings of the fourth workshop on building and evaluating resources for health and biomedical text processing.*, 2014.
 - [18] I. R. Edwards and M. Lindquist, "Social media and networks in pharmacovigilance: boon or bane?," *Drug Saf*, vol. 34, no. 4, pp. 267-71, Apr 1 2011, doi: 10.2165/11590720-000000000-00000.
 - [19] H. Dai and C.-K. Wang, "Classifying Adverse Drug Reactions from Imbalanced Twitter Data," *International Journal of Medical Informatics*, vol. 129, 05/01 2019, doi: 10.1016/j.ijmedinf.2019.05.017.
 - [20] I. Segura-Bedmar, P. Martinez, R. Revert, and J. Moreno-Schneider, "Exploring Spanish health social media for detecting drug effects," *BMC Med Inform Decis Mak*, vol. 15 Suppl 2, p. S6, 2015, doi: 10.1186/1472-6947-15-S2-S6.
 - [21] I. Segura-Bedmar, R. Revert, and P. Martinez, "Detecting drugs and adverse events from spanish social media streams," *Proceedings of the 5th International Louhi Workshop on Health Document Text Mining and Information Analysis*, 2014.
 - [22] A. Yates and N. Goharian, "ADRTTrace: detecting expected and unexpected adverse drug reactions from user reviews on social media sites," *Proceedings of the 35th European conference on advances in information retrieval*, pp. 816-9, 2013.
 - [23] P. Martínez, J. L. Martínez, I. Segura-Bedmar, J. Moreno-Schneider, A. Luna, and R. Revert, "Turning user generated health-related content into actionable knowledge through text analytics services," *Computers in Industry*, vol. 78, pp. 43-56, 2016/05/01/ 2016, doi: <https://doi.org/10.1016/j.compind.2015.10.006>.
 - [24] CIMA. "Centro de Informacion de Medicamentos de la AEMPS." AEMPS. <https://cima.aemps.es/cima/publico/nomenclator.html> (accessed 2022-01-31, 2022).
 - [25] "www.vademecum.es." (accessed 2022-01-31, 2022).
 - [26] NIH. "MedDRA (español)-Unified Medical Language System(UMLS)." NIH. <https://cima.aemps.es/cima/publico/nomenclator.html> (accessed 2022-01-31, 2022).
 - [27] "Diccionario de términos médicos." Real Academia Nacional de Medicina. <https://dtme.ranm.es/busador.aspx> (accessed 2022-01-31, 2022).
 - [28] Soldaini L. and N. Goharian, "A Fast, Unsupervised Approach for Medical Concept Extraction.," presented at the MedIR Workshop SIGIR, 2016.
 - [29] J. Lilleberg, Y. Zhu, and Y. Zhang, *Support vector machines and Word2vec for text classification with semantic features*. 2015, pp. 136-140.
 - [30] "Emoji Sequence Data for UTS #51 Version: 14.0." Unicode. <https://unicode.org/Public/emoji/14.0/emoji-sequences.txt> (accessed 2022-04-25, 2022).
 - [31] "es_core_news_md v.3.2.0." SpaCy. https://github.com/explosion/spacy-models/releases/tag/es_core_news_md-3.2.0 (accessed 2022-04-25, 2022).

Listado de siglas, abreviaturas y acrónimos

ABPI:	Association of the British Pharmaceutical Industry
AEMPS:	Agencia Española del Medicamento y Productos Sanitarios
API:	Application Programming Interface
ATC (Clasificación):	Anatomical Therapeutic Chemical
CIMA:	Centro de Información de Medicamentos de la AEMPS
CRF:	Conditional Random Forest
FDA:	Food and Drug Administration
GUI:	Interfaz Gráfica de Usuario
ICD:	International Classification of Diseases
ICH:	International Conference of Harmonisation
IFPMA:	International Federation of Pharmaceutical Manufacturers and Associations
LOINC:	Logical Observation Identifier Names and Codes
MedDRA:	Medical Dictionary for Regulatory Activities
MeSH:	Medical Subject Headings
MIT :	Massachusetts Institute of Technology
NER:	Named Entity Recognition (Reconocimiento de Entidades Nombradas)
NIH:	National Institute of Health
NLTK:	Natural Language Tool Kit
PLN:	Procesamiento de Lenguaje Natural
PoS:	Part of Speech
RAM:	Reacción adversa al Medicamento
RL:	Regresión Logística
ROC:	Receiver Operating Characteristic (Característica Operativa del Receptor)

SNOMED: Systematized Nomenclature of Medicine

SNOMED CT: Systematized Nomenclature of Medicine – Clinical Terms

SVM: Support Vector Machine

TF-IDF: Term Frequency-Inverse Document Frequency

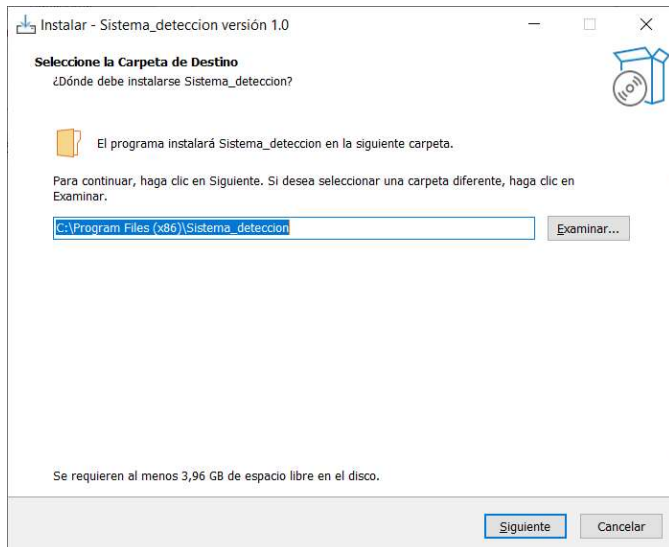
UMLS: Unified Medical Language System

URL: Uniform Resource Locators

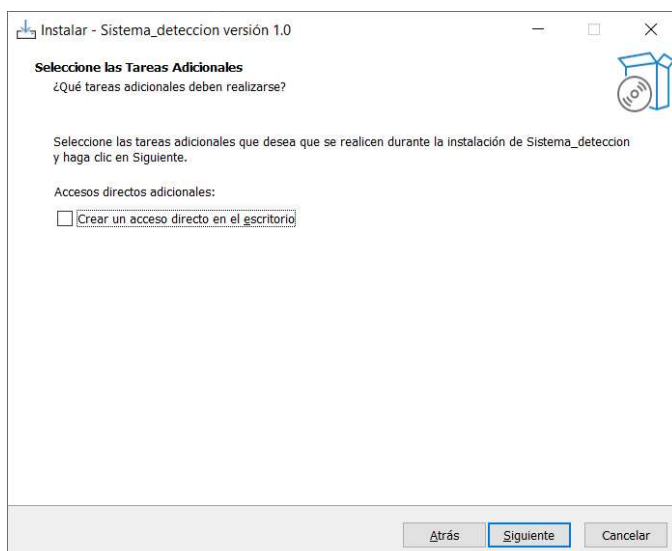
Anexo

Anexo A Guía de instalación

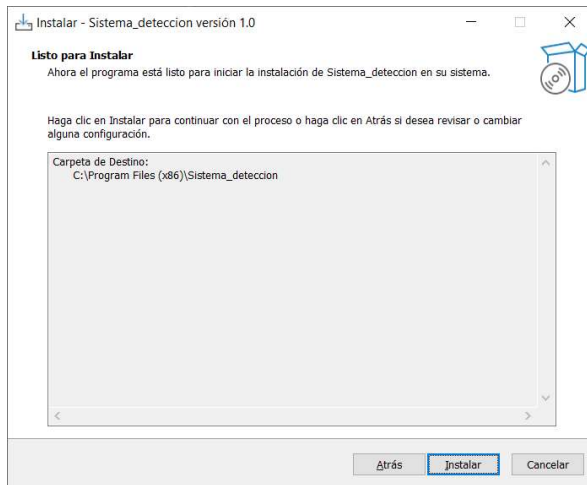
Se ejecuta el instalador llamado “SISTEMA.exe” y aparece la siguiente ventana:



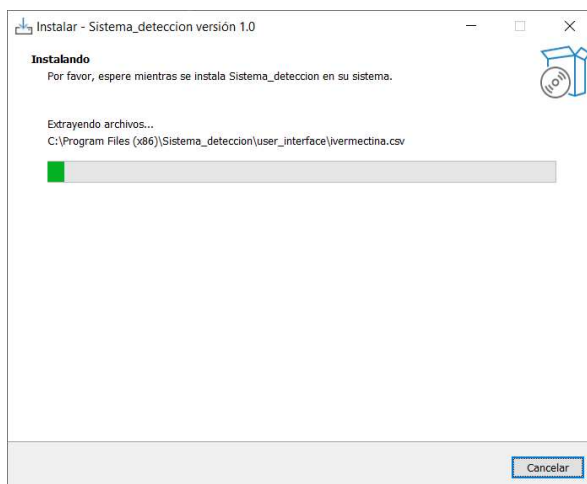
Se presiona “Siguiente” de la ventana de instalación y aparece la siguiente ventana:



Tras pulsar “Siguiente” , aparece la siguiente ventana:



Se presiona “Instalar” y comienza el proceso de instalación:

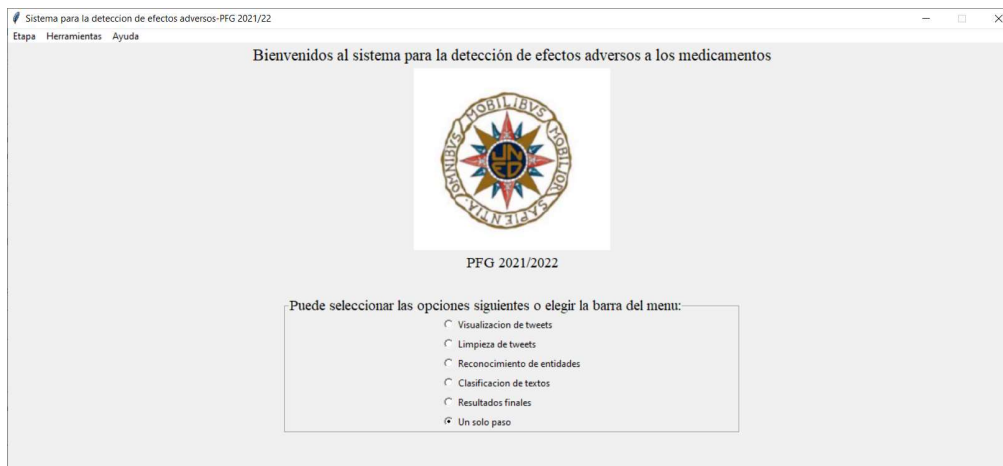


Tras finalizar la instalación, se navega por el directorio de Windows hasta que se llega a su carpeta y se ejecuta el archivo “user_interface.exe”.

La desinstalación se realiza como cualquier programa instalable de Windows, mediante Panel de control y ,posteriormente, seleccionando el programa.

Anexo B Manual de usuario

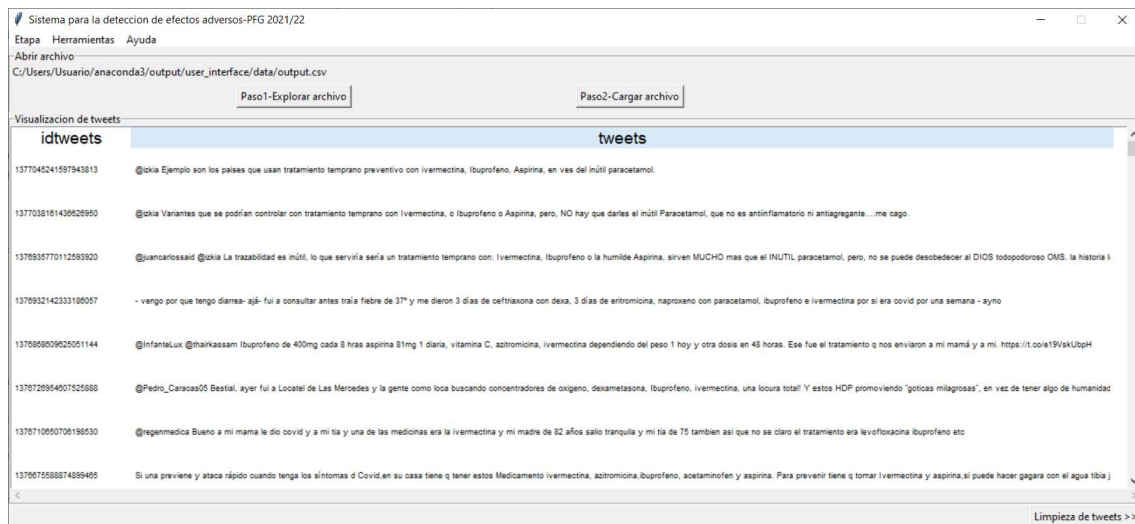
Una vez instalado, el programa se ejecuta con permisos de **Administrador** mediante el archivo “user_interface.exe”. Tras su ejecución aparece la ventana de bienvenida:



Cada una de las ventanas funciona de forma autónoma y, a la vez, conectada de forma secuencial según las necesidades del usuario, es decir, el usuario podría trabajar de forma continua o guardar el archivo resultante de cada etapa y procesarlo cuando lo necesitare. Las fases se pueden acceder pulsando en la ventana de inicio o a través de las barras de herramientas.

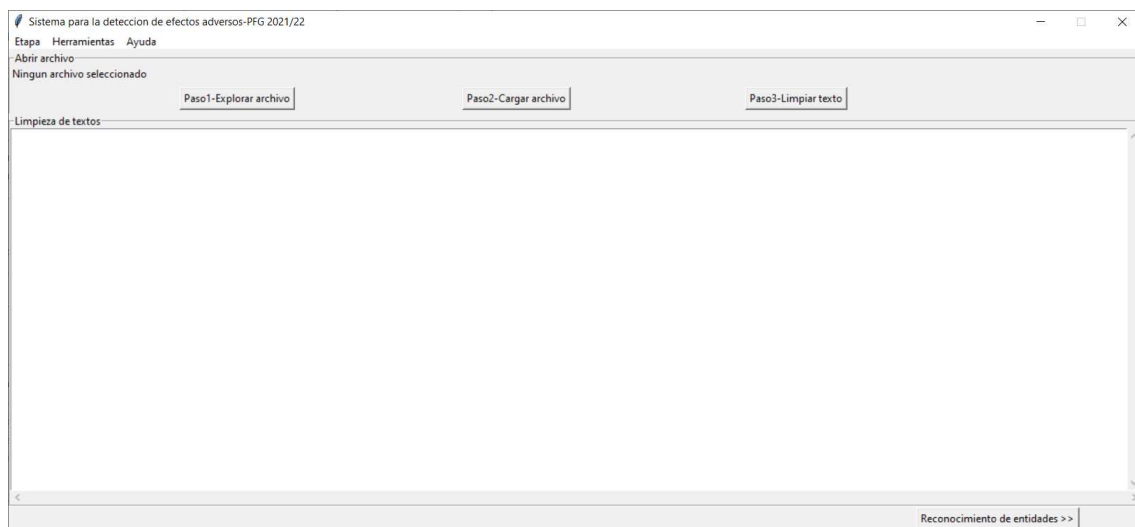
A continuación, se explica con detalle cada una de las ventanas:

1.- Visualización de tweets: únicamente realiza la visualización de los tweets obtenidos del minado. Se selecciona el archivo mediante “Paso1-Explorar archivo” y , posteriormente, se selecciona su visualización mediante “Paso2-Cargar archivo”. En el proyecto si se explora hasta la carpeta “data” se encuentra el archivo disponible del minado empleado en este proyecto llamado “./data/output.csv”. El resultado de visualizar este archivo es:

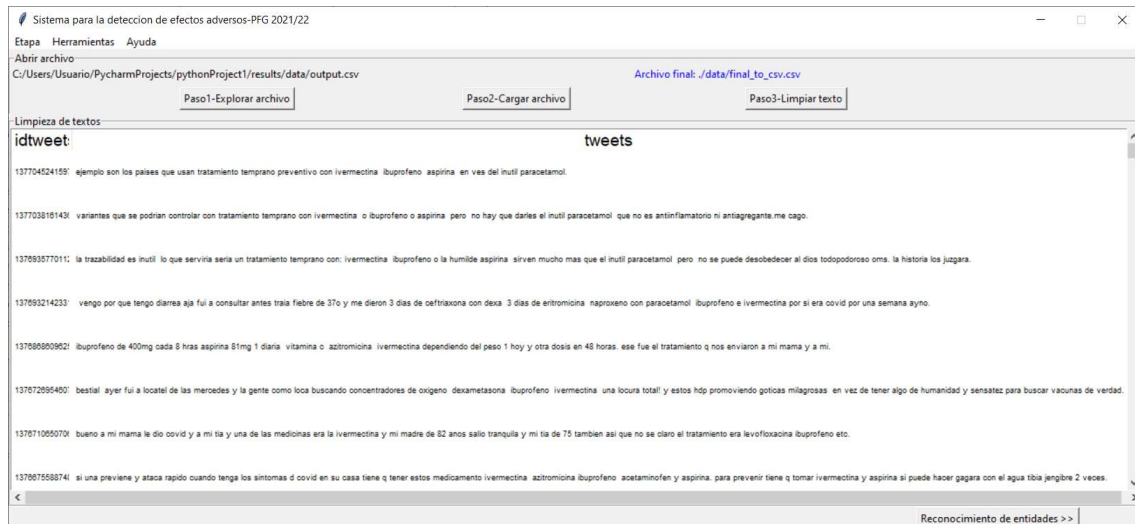


En esta etapa, podemos presionar el botón de “Limpieza de tweets” para ir a la fase siguiente.

2.-Limpieza de textos: En esta etapa se procede de la misma forma que en la etapa anterior, es decir, se pulsa botón que indica “Paso 1-Explorar archivo”, luego, el botón que indica “Paso 2-Cargar archivo” y , por último, se presiona el “Paso 3-Limpiar archivo”.



Los resultados de la siguiente visualización, tras cargar los datos y proceder a la limpieza del archivo utilizado en la etapa anterior y disponible en la carpeta de la instalación (“./data/output.csv”), se observan en la siguiente figura:



Aparece en la parte superior en azul el archivo final con terminación .csv, además se generará un archivo .txt (ambos contenidos en la carpeta data). Además, una vez terminada la etapa de limpieza se indicará en una ventana emergente opcional:

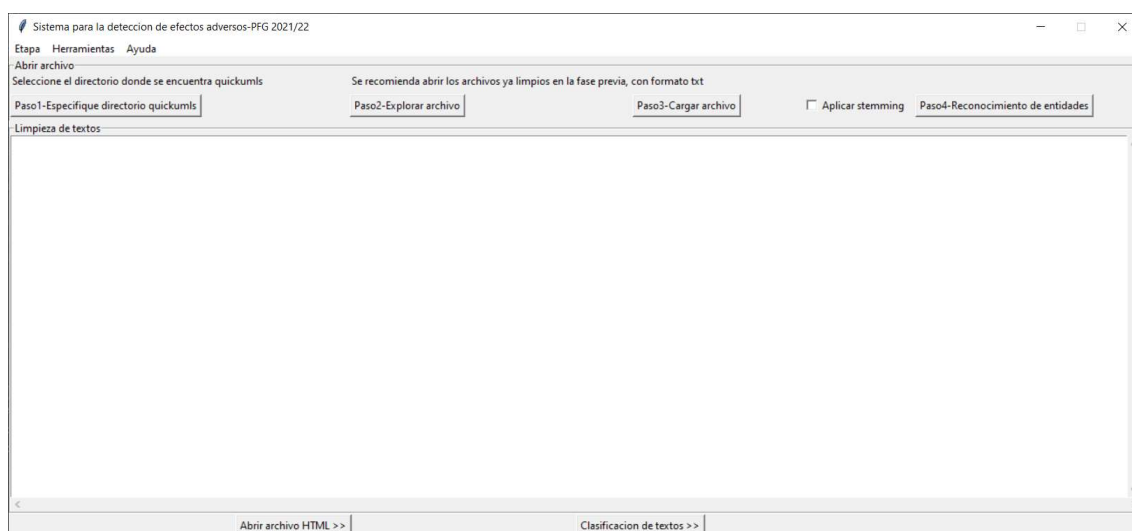


A continuación, podemos acceder a la etapa de “Reconocimiento de entidades si así se desea, presionando el botón correspondiente “Reconocimiento de entidades”.

3.-NER: en esta etapa se emplea la herramienta QuickUMLS, por lo que es necesario que instale esta herramienta para su uso en esta etapa. El archivo de instalación incorpora una carpeta denominada “quick” donde se encuentran los archivos

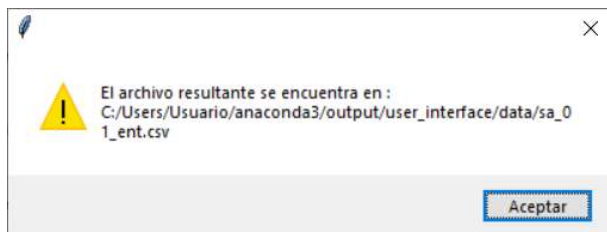
necesarios para ejecutar. No obstante, si el usuario así lo decidiese se podría utilizar otro directorio que el usuario disponga.

Debido al uso de aplicaciones externas y el empleo de grandes cantidades de texto que obligarían a un uso extensivo de memoria, se han limitados los archivos a 900000 caracteres. En el caso de que el archivo lo supere, se ha ideado unas herramientas que dividen el archivo en partes iguales y que , luego, los resultados de NER pueden concatenarse para la etapa final. Los pasos a seguir difieren un poco que, en etapas anteriores, lo primero es seleccionar donde se encuentra nuestros datos QuickUMLS que es el “Paso 1- Especifique directorio quickumls” (generará un error en el caso de que no haya sido cargado correctamente cuando se proceda al paso 4. Posteriormente, se selecciona el archivo que se desea reconocer sus entidades “Paso2-Explorar archivo” y ,una vez seleccionado, se presiona el “Paso3-Cargar archivo”



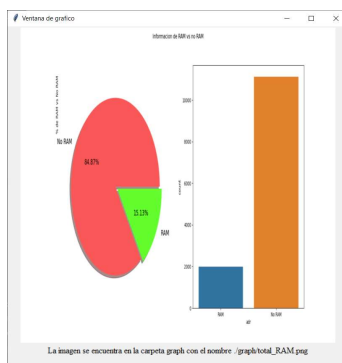
Tras este paso, se observa cómo se ha cargado en la pantalla nuestro archivo, en el caso de este manual se ha llamado “./data/txt_limpieza.txt”, debido a que este archivo tiene más de 900000 caracteres, se procede a su división en partes iguales, empleando el módulo de “Herramientas” (ver paso en el apéndice dedicado a “Herramientas”). Además, tenemos la opción de aplicar *stemming*, aunque no siempre es aconsejable (en el caso del trabajo original el *stemming* generó más problemas que beneficio, por lo que no se utilizó). Se genera como paso intermedio un archivo HTML, que se encuentra en la carpeta “data” y que se llama “NER_displacy” (ruta: “./data/NER_displacy.html”). Como advertencia al usuario, esto es un proceso lento, más lento a medida que aumenta

el número de entidades a reconocer y el tamaño del texto. Como resultado de esta etapa, se genera un archivo con un nombre acabado en “*_ent.csv”. En el ejemplo del modelo, se emplearon 18 archivos que tienen el formato “sa_”+número+”.txt” que tendrá como resultado de esta etapa un archivo llamado “sa_”+número+”_ent.csv”, disponibles en la carpeta data

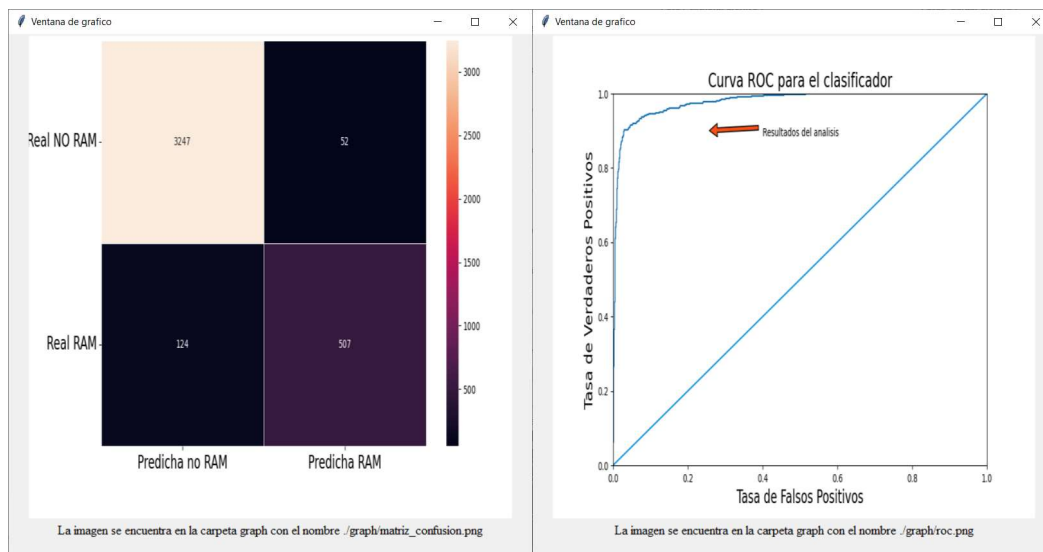


En esta etapa NER, también podemos concatenar estos archivos resultantes para el último paso, para ello se presiona el botón “Concatenar resultados NER”, el resultado final de esta etapa será un archivo final en la ruta “./data/ext_full.csv”. Otros archivos generados consisten en la lista de medicamentos encontrados (en la ruta “./data/med_data.json”), lista de entidades biomédicas encontradas (en la ruta “./data/lista_umls.txt”), así como se guardarán los componentes (pipe) del pipeline utilizado en las carpetas nlp_drug (para fármacos) y nlp_terminos (para los términos biomédicos).

4.-Clasificación de textos: Se repiten los pasos, es decir, “Paso1- Explorar archivo”, posteriormente “Paso 2-Cargar archivo” y , finalmente, “Paso3-Clasificación”. Esta es la etapa más lenta ya que se emplean algoritmos y clasificadores que enlentecen el proceso. Empleando el archivo resultante de la etapa anterior “ext_full.csv”, se obtienen varias gráficas, la primer a el número de efectos adversos detectados:

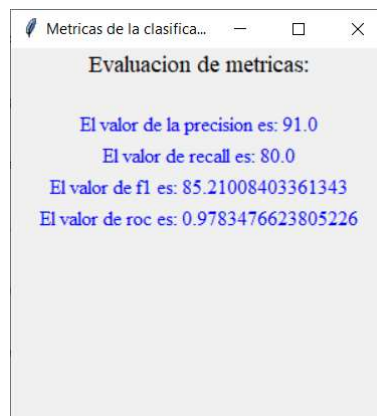


Otra gráfica obtenida sería la matriz de confusión y la curva ROC del algoritmo utilizado con esos textos:



Estas gráficas se guardarán en la carpeta llamada graph.

En esta etapa, también se pueden visualizar las métricas obtenidas pulsando al botón “Opcional-Ver métricas” y se obtiene una ventana:



Finalmente, en esta etapa se puede utilizar se puede obtener el resultado final mediante un archivo en la ruta “./data/prediccion_adr.csv” que contiene el mensaje original, el efecto adverso conocido, el efecto adverso predicho y la probabilidad de predicción. Este archivo se podrá visualizar en la siguiente etapa. Además, hay un archivo intermedio denominado “./data/ext_full_adr.csv” que contiene el mensaje original y las entidades dosis, medicamentos y UMLS.

5.- Resultados finales: Mismo proceso de etapas anteriores, se procede a “Paso 1-Explorar archivo” y , luego, “Paso 2-Cargar archivo” y se visualiza en el cuadro:

Sistema para la detección de efectos adversos-PFG 2021/22

Etapas Herramientas Ayuda

Abrir archivo

C:/Users/Usuario/anaconda3/output/user_interface/data/prediccion_adr.csv

Paso1-Explorar archivo Paso2-Cargar archivo

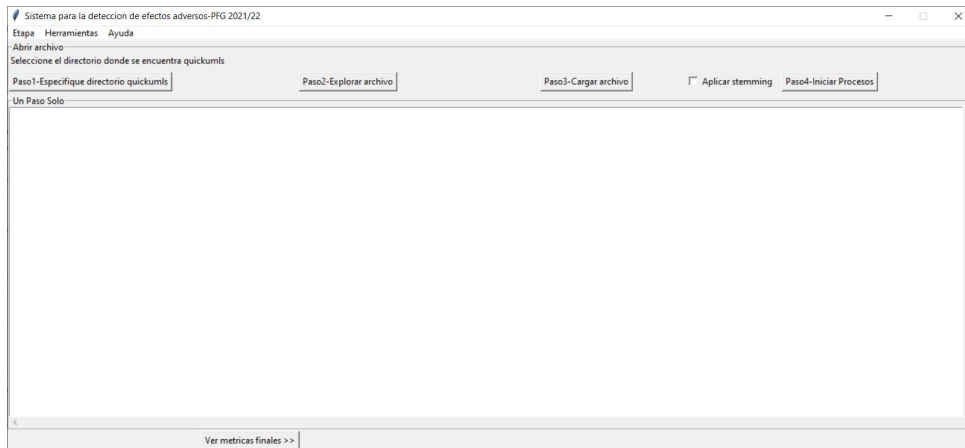
Limpieza de textos

tweets	real_RAM	predicha_RAM	probabilidad
factores asociados a sangrado mayor con uso 2000 pao riesgo de sangrado mayor: bajo peso anemia	P	N	0.004781443471108
dalay e ibuprofeno aspirina y acido acetilsalicilico: el valor que aporta la marca a los medicamentos	N	N	0.00000643991407
porque estoy bebiendo y aunque quiere morirse tampoco es plan de hacerlo rollo patético mezclando	N	N	0.002296780899183
una pregunta quien pone orden en la clinica general del norte ? desde anoche hay un paciente esperan	N	N	0.0025876669028712
que si te da covid con la vacuna lo tendras en su version light te garantiza no morir resignacion si to	N	N	0.005582638854712
reconocer pronto el expone a menos personas pero tambien reconocer rapido sintomas de gravedad	N	N	1.0274000314296935e-05
un estudio compara mortalidad por covid en paises africanos que usan ivermectina para tratar otras e	N	N	0.004027902949392
no solo me tome ibuprofeno para mi que es migrana porque no son dolores comunes así que por las t	N	N	3.90015486749185e-05

Aunque no forma parte del proceso, hay una ventana de “Herramientas” que presenta las siguientes utilidades:

- Dividir archivo: Se emplea cuando los archivos de texto son demasiado grandes para el NER. Divide el archivo en partes iguales que pueden ser procesados.
- *Web scraping* para diccionario de medicamentos y principios activos: Se ha empleado en este trabajo a partir de la página web vademecum.org, que mantiene una base de datos actualizada de medicamentos comercializados en España.
- Minado de twitter: Pulsando el botón “Cargar código de scraping de twitter” se accede al código fuente empleado para el minado de los tweets. El hecho de no incluirlo implementado se debe a que se necesitan las claves que Twitter proporciona de forma individual a cada usuario por lo que se omite en base a la confidencialidad.

6.-Un solo paso: Todas las etapas anteriores se fusionan en una sola, de forma que el sistema realiza todo en una etapa, generando archivos intermedios y visualizaciones hasta el final del proceso.



Como nota de rendimiento del programa, el programa pierde rendimiento a medida que aumenta el número de caracteres del texto. El procesamiento en un entorno Windows 10 y 32 GB de memoria RAM de 13098 tweets en la última etapa de clasificación llevan unos 360-400 segundos.