Detección de amenazas cercanas en la ciudad mediante el uso de redes sociales

Daniel Arnedo Nieto

Máster Universitario en Ingeniería de Telecomunicación

Smart Cities

Tutor/a de TF
David Crespo García

Profesor/a responsable de la asignatura
Carlos Monzo Sánchez

09/01/2023

Ficha del Trabajo Final

Título del trabajo:	Detección de amenazas cercanas en la ciudad mediante el uso de redes sociales
Nombre del autor/a:	Daniel Arnedo Nieto
Nombre del Tutor/a de TF:	David Crespo García
Nombre del/de la PRA:	Carlos Monzo Sánchez
Fecha de entrega:	01/2023
Titulación o programa:	Máster Universitario en Ingeniería de Telecomunicación
Área del Trabajo Final:	Smart Cities
Idioma del trabajo:	Castellano
Palabras clave	IA, ML, Twitter, Emergencias, Vigilancia, Detección.

Resumen del Trabajo

Las redes sociales son una herramienta cada vez más arraigada en la sociedad actual hasta el punto de que pueden influir notablemente en nuestras vidas ya sea para estar en contacto con personas cercanas, obtener noticias de última hora, notificaciones de celebridades o eventos o incluso para adquirir nuevos conocimientos culturales interesantes.

El presente proyecto propone diseñar un prototipo que sea capaz de extraer información útil de publicaciones de redes sociales para poder detectar emergencias, anomalías, altercados violentos o alarmantes que puedan poner en peligro a más civiles de los ya involucrados.

Para ello se escoge como red social de referencia Twitter, desde donde se extraerá información de distintas publicaciones. Dicha información se analizará mediante varios algoritmos de Machine Learning que detectarán sucesos que impliquen un cierto grado de riesgo.

Todo este proceso estará respaldado también por un análisis gráfico para facilitar la comprensión de los datos recopilados y explotados de una forma sencilla sin necesidad de requerir conocimientos complejos en la materia.

El objetivo de este diseño es poder tener un producto escalable y fácilmente integrable con otras tecnologías para poner un punto de partida a la interconexión con una gran cantidad de individuos mediante cualquier dispositivo electrónico que porten y recibir alertas de los sucesos detectados para evitar futuros riesgos.

Abstract

Social networks are an increasingly ingrained tool in today's society to the point that they can significantly influence our lives to be in touch with family members and friends, receive breaking news, notifications of celebrities or events or even to acquire new and interesting cultural information.

This project proposes to design a prototype capable of extracting useful information from social media publications to detect emergencies, anomalies, violent or alarming events that could endanger more civilians than those already involved.

For this purpose, Twitter is chosen as a social reference network, from where information will be extracted from different publications. This information will be analyzed using several Machine Learning algorithms that will detect events that involve a certain degree of risk.

This process will also be supported by graphical analysis to facilitate understanding of the data collected and exploited in a simple way without requiring complex knowledge in the Artificial Intelligence area.

The aim of this design is to obtain a scalable product and easily integrated with other technologies to make possible to interconnect with a large number of people through any electronic device they carry on and receive alerts of detected events to avoid future risks.

Índice

Lista de Figuras

Lista de Tablas

1. Introducción

1.1. Contexto y justificación del Trabajo

Es sabido que la cantidad de información que circula actualmente a través de internet es inmensa, tanto que muchas veces no se es consciente de las cifras tan altas que reflejan todo el volumen de datos. Gran parte de ese tráfico es generado por las redes sociales (RRSS), por ejemplo, Twitter en tan solo 1 minuto genera de media 350.000 tweets [1] por parte de los usuarios en un día corriente.

En mayor medida el contenido publicado por parte de RRSS sirve como medio de expresión personal o profesional a través de millones de cuentas a lo largo de todo el mundo, y a su vez también sirve como medio efectivo para que las personas estén conectadas con la actualidad, con su entorno, con lo que está ocurriendo en el momento preciso en el que se abre la aplicación para estar informado.
Una de las muchas preguntas que pueden surgir tras realizar esta observación es si de toda la ingente cantidad de información que rodea a las personas activas en redes sociales puede tener más propósitos, darle todavía más valor, como el de estar conectada a espacios ya sean públicos o privados o sistemas que tengan la capacidad de analizar, detectar y avisar de emergencias sociales o naturales al ciudadano.

En un mundo donde la transformación digital avanza con pasos agigantados y hace que el individuo sienta la necesidad de estar conectado con un entorno actualizado, inteligente y eficiente, es clave poder contar con el ciudadano como parte también de este proceso de cambio tecnológico para aprovechar la transición de las grandes ciudades a *Smart Cities*.

La propuesta de este trabajo trata de estudiar la viabilidad del uso de redes sociales como medio de detección de alarmas sociales o naturales por medio de un prototipo que pretende emular un sistema a una escala mucho mayor.

1.2. Objetivos del Trabajo

Este proyecto tiene como objetivos principales los siguientes:

- Estudiar el estado de otros trabajos similares que hayan analizado datos de distintas redes sociales.
- Aprender nuevos conceptos en el campo de la Inteligencia Artificial (IA) así como distintas técnicas de aprendizaje automático.
- Familiarización con nuevas herramientas tecnológicas para la recopilación de datos y su visualización.

- Obtener un conjunto de datos válido para su análisis. Dichos datos serán extraídos de la plataforma Twitter.
- Programar un algoritmo de *Machine Learning* (ML) que encuentre un patrón para detectar anomalías en un sector de la población, una ubicación o lugar concreto. Dichas anomalías serán altercados violentos o alarmantes que pueda poner en peligro a más civiles de los ya involucrados.
- Diseñar una infraestructura sencilla que de soporte a las pruebas a realizar. Esto comprende:
    - Construcción de entorno virtual para las pruebas. Incluyendo máquina, sistema operativo y entorno de programación.
    - Herramienta que extraiga los datos necesarios conectando a la API de Twitter.
    - Aplicación que permita la visualización de los datos extraídos.

1.3. Impacto en sostenibilidad, ético-social y de diversidad

Este trabajo en un principio no tiene un gran impacto en la dimensión de la sostenibilidad, dado que se centra en la detección de peligros inminentes más asociados a la interacción social. Si bien es cierto que en una explotación futura del prototipo se podrían abarcar algoritmos de detección de impacto medioambiental como pudiera ser incendios o contaminación excesiva en determinados lugares, pero todo ello no tiene cabida en este proyecto.

Observando desde una perspectiva ético-social, este proyecto puede tener un gran impacto y relacionarse estrechamente con el cumplimiento de la ODS[1] 16 (Paz, justicia e instituciones sólidas) de la Agenda 2030, puesto que se pretende reducir altercados violentos y evitar involucrar a un mayor número de civiles por accidente. Aunque este objetivo no es tan realizable en países donde se cuenta con menos recursos ya que una gran mayoría de la población no tiene acceso a tecnologías avanzadas que en países desarrollados está normalizado su uso en el día a día.

Dentro del marco de la diversidad, cabe mencionar que relacionado con la dimensión ético-social respecto a colectivos marginados o con acceso restringido a las últimas tecnologías, no podrán incluirse en la participación de esta iniciativa. No obstante, esta discriminación indirecta no perjudica a ningún otro tipo de colectivo más, siendo el proyecto pensado para ser explotado por cualquier persona de cualquier etnia, religión, orientación sexual, género o ideología por lo que se podría relacionar con el cumplimiento de la ODS 5 (Igualdad de género) y, en menor medida, con la ODS 10 (Reducción de las desigualdades).

[1] Objetivos y metas de desarrollo sostenible.

Por último, se tendrá en cuenta la protección de datos y privacidad de la información recopilada a través de las redes sociales no revelando en ningún momento ninguna información sensible de las personas que contribuyan activa o pasivamente a la alimentación del algoritmo de detección de alarmas.

1.4. Enfoque y método seguido

Este proyecto se centra en el desarrollo de un nuevo producto, de estudiar la viabilidad de dicho producto en un entorno controlado y simulado a partir de datos reales extraídos de la plataforma Twitter para localizar casos de alarmas sociales.
La idea de desarrollar un producto más complejo que abarque la detección de más tipos de emergencias y en un entorno más amplio queda fuera del alcance de este trabajo debido a que escapa en plazos y recursos. El resultado de esta investigación es el punto de partida para la construcción de una obra más elaborada con un fin más allá del académico.

La metodología seguida durante el transcurso de este proyecto será por una parte cuantitativa, ya que se pretende analizar el contenido de las publicaciones de la red social Twitter para identificar comportamientos, intenciones y lugares de interés. Si bien este análisis se usaría con otro fin distinto al de conocer algo más acerca de la persona de la que se extrae toda la información, es necesario este estudio sobre sus publicaciones para lograr el propósito del trabajo.
Por otra parte, también se sigue una metodología cualitativa dado que la recolección de datos parte de la observación de los distintos usuarios de la plataforma que alimentarán la muestra de datos a analizar. Estos datos seguirán llegando y se seguirán analizando independientemente de que se haya detectado ya algún tipo de alarma puesto que la situación podría no haberse solucionado todavía o podría haberse vuelto a producir y es por ello por lo que el canal de extracción de datos deberá seguir abierto y en constante funcionamiento.

1.5. Planificación del trabajo

Los recursos disponibles para el desarrollo de este proyecto son:

- Trabajos y estudios de trabajos similares publicados online.
- Páginas web especializadas, revistas y artículos electrónicos.
- Documentación oficial de las herramientas utilizadas para la construcción del entorno.
- Material docente de asignaturas cursadas a lo largo del máster.
- Biblioteca UOC.
- Consultas al tutor del TFM.

Las tareas que se llevan a cabo a lo largo del TFM se dividen principalmente en 5 grandes grupos:

1. **Definición de objetivos y alcance del proyecto**. La primera toma de contacto con el trabajo a desarrollar, se deberán tratar los puntos principales sobre los que se quiere abordar el TFM para definir tareas y objetivos.
2. **Estudio del estado del arte.** Donde se analizan otros proyectos similares al propuesto, así como el desarrollo de algoritmos de IA y arquitecturas que dan soporte estable a estos algoritmos para definir un diseño de prototipo a implementar. Se estudiarán las distintas tecnologías involucradas para dicho diseño incluyendo, además de algoritmos de IA/ML e infraestructura, herramientas de ingesta de datos y explotación por medio de la visualización de dichos datos.
3. **Diseño, implementación y análisis del prototipo.** Puesta en marcha del proyecto donde se despliega entorno de pruebas, desarrollo de algoritmo, infraestructura y se llevan a cabo pruebas de validación y análisis de la solución obtenida.
4. **Redacción de la memoria final.** Se hace un repaso de todos los apartados y se llevan a cabo correcciones de entregas anteriores además de dar un formato final al documento.
5. **Defensa del TFM.** Incluye la realización de la presentación y su ensayo para la preparación de la defensa del proyecto ante el tribunal.

La planificación del proyecto queda reflejada en el siguiente diagrama de Gantt de la Figura 1:

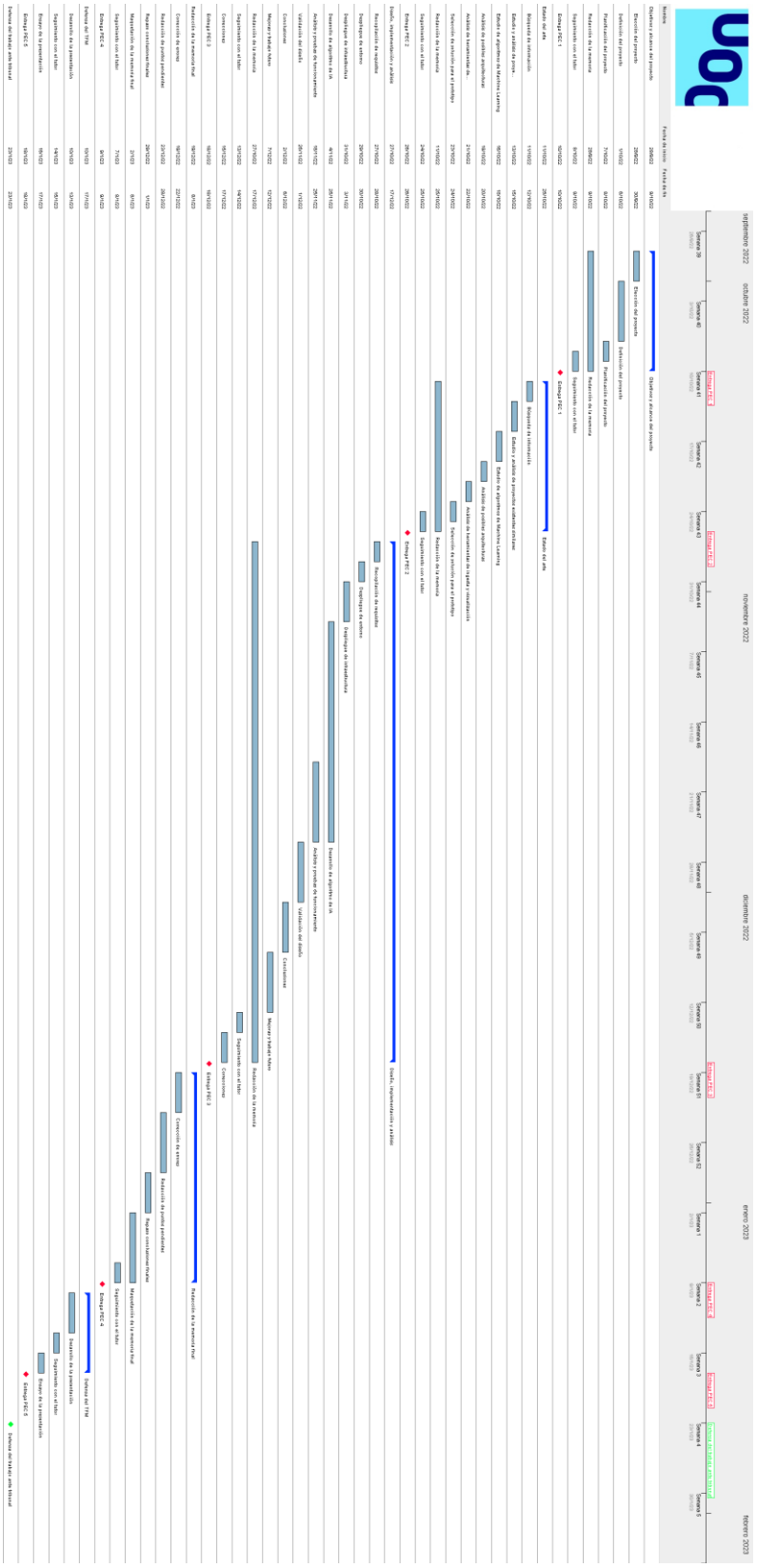


Figura 1. Diagrama de Gantt del proyecto

1.6. Breve sumario de productos obtenidos

A continuación, se listan los productos obtenidos:

- Datos reales obtenidos de distintas publicaciones de Twitter en formato JSON.
- Algoritmos basados en tecnología de *Machine Learning* realizados en lenguaje de programación Python.
- Entorno pequeño para tratar la información almacenada para su uso y explotación.
- Prototipo capaz de detectar amenazas reales y pueda ser escalable e integrable con otras tecnologías.

1.7. Breve descripción de otros capítulos de la memoria

A continuación, se incluye una pequeña descripción del resto de capítulos que componen la memoria del proyecto.

Capítulo 2 – Estado del arte
Se relata el estudio de otros proyectos similares al propuesto, así como el desarrollo de algoritmos de IA y arquitecturas que dan soporte estable a estos algoritmos para definir un diseño de prototipo a implementar. Se analizan distintas tecnologías involucradas para dicho diseño incluyendo, además de algoritmos de IA/ML e infraestructura, herramientas de ingesta de datos y explotación por medio de la visualización de dichos datos.

Capítulo 3 – Diseño del prototipo
Se describe el proceso seguido para llegar a tener un prototipo válido para extraer resultados concluyentes. Este proceso incluye la puesta a punto de un entorno de desarrollo, la extracción de información útil de Twitter, el preprocesamiento y procesamiento de los datos extraídos y set de datos de entrenamiento, la construcción de algoritmos de ML y comparación teórica de los modelos construidos.
También se opta por incluir un subapartado con conceptos importantes que facilitan la comprensión de los resultados teóricos (y más adelante prácticos).

Capítulo 4 – Análisis y resultados
Se exponen los resultados obtenidos tras la aplicación de los distintos modelos de ML construidos sobre los datos extraídos mediante el uso de tablas y gráficas útiles para el análisis.

Capítulo 5 – Conclusiones y trabajos futuros
En este capítulo final se valorará todo el trabajo realizado detallando las lecciones aprendidas, los objetivos logrados, planificación cumplida, impactos y líneas futuras desde las que continuar el desarrollo en el que queda este proyecto.

2. Estado del arte

2.1 Introducción

En este capítulo se realiza un análisis de la situación en la que se encuentra el ámbito del trabajo propuesto. De esta forma, se puede tener una base teórica para proponer un diseño del prototipo final.

A lo largo del capítulo se hablará acerca de aplicaciones de referencia que ya son operativas a día de hoy para alertar a los ciudadanos de posibles peligros, la manera en la que la API de Twitter permite extraer información de su plataforma, tecnología de IA y ML aplicada al análisis de publicaciones en Twitter, herramientas que facilitan la visualización de los datos extraídos y analizados, arquitecturas Big Data y por último, retos importantes y el impacto que tendría el desarrollo de este trabajo en el marco legal.

2.2 Aplicaciones operativas similares

Actualmente no hay un sistema estable conocido que sea capaz de avisar a la población de situaciones de peligro gracias a las redes sociales. Aunque sí es cierto que ya existen aplicaciones que alertan a los ciudadanos de peligros cercanos o inminentes de gran importancia.

Estas aplicaciones se pueden encontrar a nivel nacional como, por ejemplo:

- Radar COVID: puesta en marcha al poco tiempo del inicio de la pandemia provocada por el virus SARS-CoV-2, esta aplicación diseñada para móviles notificaba a aquellos usuarios que la tuvieran instalada y hubieran estado en reciente contacto con un positivo que también tuviera instalada la aplicación y hubiera introducido un código facilitado por el centro de salud que realizó la prueba PCR[2]. Al ingresar este código, Radar COVID procedía a enviar una alerta a todos aquellos dispositivos que habían quedado guardados en la memoria mediante conexión Bluetooth al estar en contacto cercano con el dispositivo emisor. [2]
- Public Warning System o ES-Alert: actualmente se encuentra en período de prueba y no ha sido desplegada por completo, pero ya se han enviado algunos mensajes de prueba entre la población. Se trata de un sistema de avisos SMS por medio de

[2] Siglas en inglés de 'Reacción en Cadena de la Polimerasa', es una prueba que se utiliza para el diagnóstico de detección de un patógeno determinado a partir de un fragmento de material genético de referencia.

tecnología Cell Broadcast[3] a los ciudadanos en caso de catástrofes, emergencias o accidentes industriales validadas por las autoridades nacionales correspondientes. Este sistema de alerta basado en telefonía móvil viene impuesto a nivel europeo[4] ya que todos los estados miembros cuentan con su propio sistema o están en proceso de despliegue. [5]

A nivel internacional, se pueden destacar los siguientes ejemplos:

- Alerta AMBER: se está intentando desplegar en varios países, pero tiene su origen en Estados Unidos (concretamente en la ciudad de Dallas) allá por 1996. Este sistema es capaz de alertar a la población de la desaparición de niños por numerosos medios electrónicos desde teléfonos móviles, televisión hasta vallas publicitarias o señales de tráfico. Estas alertas se emiten una vez hayan sido corroboradas por las fuerzas del orden. [6]
- Cruz Roja Americana: motivada por los desastres naturales que arremeten con frecuencia en las costas americanas, la Cruz Roja Americana puso en marcha el funcionamiento de distintas aplicaciones durante la llegada de los huracanes Harvey, Irma y María durante septiembre de 2017. Estas aplicaciones se instalan tanto en dispositivos Android como iOS y permiten alertar a los ciudadanos de la llegada de huracanes, tornados, terremotos o emergencias catastróficas, así como su monitorización en tiempo real además de aconsejar e informar a la población sobre qué hacer ante estos sucesos. [7]

2.3 Ingesta y explotación de los datos

Para abordar el problema planteado en este proyecto, uno de los primeros pasos a seguir es conseguir extraer información de la plataforma de referencia elegida para analizar publicaciones de redes sociales, es decir, Twitter. Para ello es necesario previamente tener o crear una cuenta desde su página web o aplicación móvil para luego poder habilitar dicha cuenta como perfil desarrollador de forma gratuita.

Una vez validada la cuenta, Twitter proporciona un portal de desarrollador desde donde se pueden adquirir las credenciales para acceder a la API de la plataforma [8]. Este acceso está limitado para extraer una cantidad de 500.000 tweets ya que si se quisieran más habría que elegir un plan premium y para este caso de estudio es una cantidad más que suficiente.

Cuando ya se tiene acceso a la API de Twitter el siguiente paso es formular una petición a este servicio con las credenciales que proporcionan para extraer toda la información deseada. En este trabajo se describen 3 vías o herramientas efectivas de llevarlo a cabo que

[3] Diseñada para el envío de datos a múltiples usuarios de un área seleccionada. [3]
[4] Directiva 2018/1972 [4]

pueden servir de referencia para el desarrollo del proyecto: script en Python, RapidMiner y NiFi.

Script en Python. Desde la propia página web de Twitter dedicada para desarrolladores hay varios tutoriales que indican cómo dar los primeros pasos extrayendo información usando varias librerías de Python (como por ejemplo *tweepy* o *requests*). Es una opción bastante interesante ya que es un lenguaje de programación muy utilizado en el ámbito profesional actualmente, sobre todo en el desarrollo de algoritmos de IA y además tiene una comunidad muy amplia en la que apoyarse.

RapidMiner. Se trata de una herramienta basada en flujos de bloques donde cada uno de los bloques realiza una operación determinada, desde conexiones a bases de datos hasta aplicación de modelos de ML. Tiene un conector exclusivo que permite la conexión a Twitter lo cual es bastante interesante ya que se puede realizar la conexión directamente desde esta herramienta e incluso aplicar capas de procesamiento o preprocesamiento antes de analizar más a fondo los datos extraídos.
Está orientada para el uso empresarial y educativo por lo que se puede descargar una versión gratuita sin problema. [9]

NiFi. Muy similar a RapidMiner pero de licencia Apache, por lo que su utilización está abierta al público. Tampoco posee bloques exclusivos para aplicar algoritmos de aprendizaje automático, pero sí posee dos[5] para obtener datos directamente desde la API de Twitter lo que hace que esta herramienta sea muy interesante a tener en cuenta para este TFM. [10]

2.4 Tecnología aplicada al análisis del contenido de Twitter

2.4.1 Introducción a la Inteligencia Artificial y *Machine Learning*

La Inteligencia Artificial es la manera de dar respuesta, por parte de sistemas o máquinas, a procesos imitando las funciones cognitivas de los seres humanos. Según Nils J. Nilsson, uno de los investigadores fundadores en esta ciencia, se debe basar en 4 pilares fundamentales [11]:

- Búsqueda del estado requerido en el conjunto de los estados producidos por las acciones posibles.
- Algoritmos genéticos (análogo al proceso de evolución de las cadenas de ADN).

[5] Uno de los bloques de procesamiento que utiliza (*GetTwitter*) ya no tiene la capacidad de conectarse con Twitter, ya que se conecta mediante la primera versión de la API y la red social deshabilitó el acceso a la primera versión en proyectos de desarrollador iniciados después de abril de 2022. Aun así, el bloque todavía se puede integrar en el flujo construido desde NiFi para aquellos desarrolladores que tienen credenciales de acceso anteriores, aunque se acabará eliminando en versiones posteriores de NiFi.

- Redes neuronales artificiales (análogo al funcionamiento físico del cerebro de animales y humanos).
- Razonamiento mediante una lógica formal análogo al pensamiento abstracto humano.

Dentro de esta disciplina hay 2 ramas importantes para la aplicación de algoritmos de Inteligencia Artificial, que son: *Machine Learning* ("Aprendizaje Automático") y *Deep Learning* ("Aprendizaje Profundo").

Por una parte, la rama de *Machine Learning* tiene como objetivo "permitir a las máquinas" aprender en base a los datos de información que se le otorgan y realizar predicciones certeras de datos futuros. Por otro lado, *Deep Learning* pretende llegar a un nivel más minucioso, partiendo de la base de la aplicación de ML, pero apoyándose concretamente en la aplicación de capas o redes neuronales (procesamiento de datos secuencial). La diferencia entre ambos es que en DL no es necesaria la supervisión de un ingeniero en su desarrollo, ya que los algoritmos pueden obtener resultados satisfactorios (o no) y ser "conscientes" de ello.

Dentro de esta rama tecnológica, es también importante conocer otros dos conceptos clave: aprendizaje supervisado y aprendizaje no supervisado [12] [13]. A continuación, se explican las principales características de ambos:

- **Aprendizaje supervisado**. Es aplicado cuando se tiene un conjunto de datos debidamente etiquetados gracias a los cuales se pueden entrenar algoritmos capaces de predecir resultados y obtener un porcentaje de precisión comparando los datos de salida con los de entrada etiquetados. Existen dos tipos de algoritmos principales:
    - **Algoritmos de Clasificación**. Su uso principal es para poder catalogar los datos que recibe y separarlos de otros que pudieran ser similares. Por ejemplo, su uso podría aplicarse para separar los mensajes importantes de los de spam en el correo electrónico. Algunos algoritmos dentro de este tipo son: *Support Vector Machine* (SVM), *Random Forest (RF)* o Clasificadores Lineales.
    - **Algoritmos de Regresión**. Son usados para predecir resultados de un flujo de datos en constante cambio. Un ejemplo de uso podría aplicarse para predecir el movimiento del mercado de la vivienda. Algunos algoritmos dentro de este tipo son: Regresión Lineal y Regresión Polinómica.

- **Aprendizaje no supervisado**. Es de gran utilidad para encontrar patrones, tendencias o comportamientos ocultos entre un conjunto de datos sin la necesidad de la supervisión de una persona, pues a diferencia de los algoritmos de aprendizaje supervisado, no se tienen datos etiquetados. Existen tres tipos de algoritmos principales:

- **Agrupamiento o *Clustering***. Son de ayuda para agrupar los datos en diferentes grupos según similitudes encontradas entre estos. Un uso muy adecuado de este tipo de técnica sería el de encontrar anomalías en segmentos de datos al encontrarse fuera del clúster principal. Algunos algoritmos representativos dentro de este tipo son: *K-Means* y *Gaussian Mixture Models*.
- **Asociación**. Se utilizan para encontrar relaciones entre distintas variables a medir entre los datos recopilados para así poder entender su conexión. Es muy útil para generar algoritmos que generen recomendaciones de productos comerciales a los usuarios. Uno de los algoritmos más utilizados es *Apriori Algorithm*.
- **Reducción dimensional**. Su uso es importante cuando se quieren descartar algunas características de los datos recopilados como, por ejemplo, el ruido en imágenes. Algunos de los algoritmos más usados dentro de este tipo son: *Singular Value Decomposition* (SVD) y *Linear Discriminant Analysis* (LDA). [14]

2.4.2 IA y ML aplicada a Twitter

Es muy común encontrar trabajos y estudios en el campo de la Inteligencia Artificial aplicada a redes sociales, sobre todo a la plataforma de Twitter. En este apartado se han identificado varios proyectos que pueden aportar información valiosa al estudio de la solución de este TFM y que aplican técnicas de *Machine Learning* para poder extraer información útil de las publicaciones realizadas por los usuarios.

Uno de los procesos más utilizados para analizar textos extraídos de redes sociales es el análisis de sentimientos, una rama del procesamiento del lenguaje natural que pretende dar significado a la intención del autor del texto a través de técnicas de aprendizaje automático. Este tipo de análisis generalmente trata de agrupar las palabras en positivas, negativas o neutras o también en una escala numérica de puntuaciones (de palabras menos felices a más felices) previamente evaluadas por varias personas. Una vez hecha esta agrupación se aplican técnicas de ML para conocer la intención del usuario, su opinión respecto a un tema, la forma de expresarse o incluso su estado de ánimo.

Algunos ejemplos destacados de artículos que analizan publicaciones de Twitter son:

- *Application of Text Classification and Clustering of Twitter Data for Business Analytics* de Alrence Santiago Halibas [15], donde aplica por un lado un algoritmo de árbol de decisiones y por otro un algoritmo de clustering, concretamente *K-Means*. Mediante la aplicación de un árbol de decisiones obtuvo una precisión total del 70,45% mientras que aplicando *K-Means* obtuvo una precisión del 85% en palabras negativas.
- En *Twitter Sentiment Analysis using Naïve Bayes Classifier with Mutual Information Feature Selection* Maria Arista Ulfa et al. [16] aplican un clasificador Naïve Bayes

para analizar la polaridad de los textos extraídos de Twitter, es decir, averiguar si se escribió con una intencionalidad negativa o positiva. Este trabajo obtuvo un porcentaje de acierto del 96,2%. Cabe mencionar que además de aplicar Naïve Bayes también se usó la técnica de *Mutual Information Feature Selection* (MI) que combinada con la anterior pasaba a obtener un acierto del 97,9% además de reducir el tiempo de ejecución a la mitad.

- El artículo *Sentiment Analysis on Twitter using Neural Network: Indonesian Presidential Election 2019 Dataset* publicado por Ahmad Fathan Hidayatullah et al. [17] estudia la aplicación de algoritmos de redes neuronales, concretamente *Convolutional Neural Network* (CNN), *Long Short-term Memory* (LSTM), CNN-LSTM, *Gated Recurrent Unit* (GRU) y *Bidirectional LSTM* en las opiniones publicadas en Twitter por la población de Indonesia durante las elecciones presidenciales de 2019. Este trabajo logró alcanzar el mayor porcentaje de acierto con el algoritmo *Bidirectional LSTM*, concretamente una precisión del 84,6% y es bastante interesante ya que todos los resultados obtenidos rondan el 84% de acierto además de comparar dichos resultados con otras técnicas más tradicionales de ML como *Support Vector Machine* (SVM), *Logistic Regression* (LR) o *Multinomial Naïve Bayes* (MNB) obteniendo estas 84%, 83% y 82% de precisión respectivamente.

La gran cantidad de publicaciones generada hace posible el análisis en tiempo real de aquello sobre lo que los usuarios más hablan. Este análisis es el que ha realizado Anisha P. Rodrigues et al. en su artículo *Real-Time Twitter Trend Analysis Using Big Data Analytics and Machine Learning Techniques*. [18]
Dicho análisis compara varias técnicas de *Machine Learning* para el estudio de tendencias de publicaciones en tiempo real y de publicaciones almacenadas menos recientes. Las técnicas utilizadas fueron *Latent Dirichlet Allocation* (LDA), Jaccard y *K-Means Clustering*. Las dos primeras tienen como objeto el clasificar por temas las palabras clave encontradas en las publicaciones, mientras que el algoritmo de *K-Means* es aplicado para reforzar la salida obtenida de las dos anteriores permitiendo así una mejor relación entre los temas y las palabras incluidas en los mensajes generados por los usuarios en sus publicaciones.
La precisión en los resultados obtenidos en este trabajo fue de un 74% aplicando LDA y de un 83% aplicando Jaccard.

Por último, cabe destacar el trabajo *Finding Local Events Using Twitter Data* realizado por David Chen et al. [19] donde se lleva a cabo un gran estudio de un set de datos recopilado de Twitter con el objetivo de identificar los eventos más importantes sucedidos en un área geográfica determinada. En este proyecto se utilizan varias técnicas: *K-Means*, *Hierarchical Clustering*, *Density-Based Spatial Clustering of Applications with Noise* (DBSCAN) y *High Density-Based Spatial Clustering of Applications with Noise* (HDBSCAN).
La utilización de K-Means y DBSCAN genera clústeres demasiado grandes y difusos mientras que HDBSCAN es más eficaz si se quiere obtener distribuciones de clústeres más limpias y precisas, es decir, agrupaciones de tweets más pequeñas que estén relacionados.

Después de detectar los clústeres más relevantes se pasa a encontrar el pico de tweets a lo largo del tiempo y de este pico analizar el texto contenido en ellos y valorar aquellos que tengan información más relevante por medio del análisis de las *content words*[6].
Finalmente, se logra detectar y ordenar los eventos de más a menos relevantes siendo en este caso el más destacado un concierto en Londres seguido del cumpleaños de la reina de Inglaterra.

2.4.3 Otras tecnologías

Es importante mencionar que otros proyectos con menos implicación en la parte técnica o de programación también han llevado a cabo análisis interesantes sobre aspectos sociales a través de Twitter, pero en lugar de desarrollar algoritmos, han utilizado como herramienta de apoyo Meaning Cloud.

Meaning Cloud es un software capaz de realizar un análisis de sentimientos a partir de los textos generados por usuarios de distintas redes sociales estudiando la polaridad, acuerdo, ironía o incluso objetividad de la opinión. Su funcionamiento parte de varios algoritmos basados en la teoría de la rueda de las emociones[7] y tiene una de las precisiones más bajas (en torno al 67%) con respecto a otros modelos tradicionales como Regresión Logística o softwares como Amazon Comprehend[8]. [21]

A pesar de no tener una precisión especialmente elevada, algunos proyectos que se han servido de este software son:

- *La repercusión del movimiento MOOC en las redes sociales. Un estudio computacional y estadístico en Twitter* para la revista española de pedagogía [22]. Donde se analiza el impacto del uso de los cursos tipo MOOC (*Massive Online Open Courses*) en todo el mundo mediante el análisis de los perfiles oficiales de MOOC y perfiles personales en la plataforma Twitter.
- *Redes sociales y su influencia en los jóvenes y niños: Análisis en Instagram, Twitter y YouTube* para la Revista Científica de Comunicación y Educación [23]. Este proyecto tiene como objetivo estudiar los personajes públicos más seguidos en Twitter, Instagram y YouTube a través de sus publicaciones en estas plataformas y cómo su forma de actuar es capaz de crear modelos a seguir para las generaciones más jóvenes que siguen de cerca estas figuras públicas de Internet.

[6] Traducido del inglés “palabras de contenido”, son palabras que tienen gran relevancia a la hora de otorgar significado a un texto.
[7] Defiende un modelo de análisis de las emociones mediante su clasificación y relación con otras emociones teniendo como base 8 emociones principales. El resto de las emociones parten de estas 8, que pueden combinarse entre sí para formas emociones compuestas. Por ejemplo, la combinación de alegría y confianza darían lugar al amor. [20]
[8] Servicio de Amazon para el análisis del PLN (Procesamiento del Lenguaje Natural).

2.5 Herramientas de apoyo para el análisis y la visualización de los datos

Dado que llevar a cabo el análisis de un gran número de publicaciones recopiladas de Twitter, y ya no solo de dichas publicaciones, sino de las transformaciones aplicadas sobre ellas para extraer información útil, es un trabajo que requiere esfuerzo y dedicación, es adecuado contar con herramientas visuales de apoyo que faciliten el estudio de los datos recopilados y procesados.

En este punto es interesante destacar la herramienta Jupyter Notebook, mediante la cual, a través de una sencilla interfaz web se pueden programar scripts en lenguaje de programación Python y generar salidas en bloques de código de una forma bastante ilustrativa gracias a que Python cuenta con numerosas librerías para generar salidas gráficas de gran variedad dependiendo del ámbito en el que se esté trabajando. Jupyter Notebook soporta más lenguajes de programación como R, Julia o Haskell entre muchos otros. [24]

Otras herramientas que comparten similitudes con Jupyter son Zeppelin [25] y Google Colab [26]. La primera de ellas es *Open Source* y además permite fácilmente su integración con otras tecnologías de código abierto orientadas al *Big Data* tales como HDFS (*Hadoop Distributed File System)* para el almacenamiento distribuido o Spark para el procesamiento de los datos. Es también bastante versátil ya que permite programar en varios lenguajes de programación en un mismo código gracias a lo que se conocen como intérpretes[9] dentro de la aplicación.
Por otro lado, Google Colab está basado en Jupyter, con la diferencia de que no necesita de los recursos locales del usuario al estar alojado el servicio en la nube de Google por lo que es una gran ventaja para investigadores con prestaciones limitadas.

Existen otras herramientas más orientadas a la gobernanza del dato tales como Power BI, Qlik View o Superset que ofrecen vistas más elaboradas para relacionar la información mediante consultas a tablas, bases de datos o importaciones, pero dado que este tipo de software es utilizado con un objetivo más dirigido al ámbito de negocio que al de investigación se descarta su uso y estudio en este TFM.

2.6 Arquitecturas *Big Data*

Cuando se desarrollan aplicaciones que necesitan almacenar un gran volumen de datos es importante desplegar una infraestructura que de soporte a todas las aplicaciones involucradas.

[9] Son unas extensiones que permiten transcribir el contexto necesario a nivel software por debajo de la capa del servicio de Zeppelin Notebook para utilizar un lenguaje de programación determinado elegido por el usuario. [27]

Para el estudio de información recopilada de redes sociales se espera que lleguen grandes flujos de datos constantemente por lo que se debería contar con un sistema capaz de gestionar una carga de trabajo considerable pues se deben ingestar los datos, almacenarlos, procesarlos, aplicar técnicas de IA para su análisis, almacenar los datos limpios, etc.

Estas últimas décadas la gestión de todas estas operaciones sobre un flujo masivo de datos es un orden prioritario en casi todas las empresas del sector TIC (Tecnologías de la Información y las Comunicaciones) y es muy común usar sistemas distribuidos.

Los sistemas distribuidos se componen de varias máquinas (o nodos) independientes conectadas entre sí que proporcionan una serie de características indispensables para lo descrito en los párrafos anteriores de esta sección tales como el particionado de los datos y se replicación en diferentes nodos para el acceso y recuperación de la información de forma rápida y fiable; la escalabilidad del sistema, pues un crecimiento exponencial de los datos almacenados puede conllevar una necesidad de aumentar los componentes que integran la arquitectura que da soporte a las aplicaciones; o la tolerancia a fallos pues el mal funcionamiento o parada de una de las máquinas que componen la arquitectura no afecta a la continuidad del funcionamiento de todo el clúster.

Algunos ejemplos de arquitecturas basadas en sistemas distribuidos son:

- Cloudera Data Platform
- Amazon Web Services
- Google Cloud
- Microsoft Azure

Sin embargo, en este proyecto, al trabajar con un conjunto de datos relativamente pequeño y presentar muchas cuestiones a tratar, no aplicaría el despliegue y desarrollo de arquitecturas Big Data porque se desviará demasiado el objetivo del trabajo y además solo se va a contar con una cantidad de información limitada (inferior al orden de terabytes con el que generalmente se trabaja en el ámbito profesional) que no requerirá de mucho tratamiento y coste computacional.

2.7 Retos y marco legal del proyecto

Si bien este trabajo tiene un enfoque muy tecnológico es interesante también indagar sobre otros aspectos más ligados a la idea de negocio.

2.7.1 Retos

Como se ha podido comprobar en apartados anteriores de este proyecto, conseguir precisiones de acierto superiores al 90% es complicado pues el análisis del lenguaje humano

es muy impredecible y no siempre se puede deducir correctamente la plena intencionalidad de las palabras de una persona a través de una publicación de una red social.

En este punto entra en juego el PLN, pues puede tener un gran impacto el análisis del significado real de las palabras analizadas de un texto. Sin embargo, para este trabajo interesa más una de sus subramas, que es el análisis de sentimientos, ya que detectar amenazas lleva implícito que algo malo está ocurriendo y, por lo tanto, tener una predicción que clasifique polaridades entre positivas y negativas ya es un adelanto considerable en lo que al análisis se refiere pues se descartaría mucha información inservible. Para llevar a cabo este objetivo se pretende delegar ese esfuerzo por medio del entrenamiento de algoritmos de ML sin entrar en técnicas de análisis de PLN en mucha profundidad.

Mucha de la información desechable para el resto de los procesos también se puede detectar mediante la identificación de las conocidas *content words*. El análisis para su detección tampoco es trivial, pero puede servir como soporte para generar un algoritmo más preciso.

Otro de los retos a los que se debe hacer frente es a la ubicación de un suceso. Muchos usuarios no tienen habilitada la opción que permite a Twitter obtener las coordenadas GPS (*Global Positioning System*) de su posición en el momento de publicar un tweet. Esto es un problema bastante importante, pues si no se puede ubicar un acontecimiento es imposible de avisar al usuario de lo que está sucediendo, ya que, si además la probabilidad de acierto se encuentra relativamente lejos de la deseada, se podría llegar a producir el efecto contrario del esperado, que es el de ofrecer seguridad a las personas para evitar situaciones de riesgo.

2.7.2 Marco legal

Es importante tener en cuenta la Ley Orgánica 7/2021 [28] sobre la protección de datos y el Reglamento General de Protección de Datos (RGPD) [29]. Aunque este trabajo no sea para un fin comercial sino educativo, se recogerá información de personas físicas en su mayoría y por tanto en alguna ocasión podría incluir alguna información sensible.

No se almacenará ni compartirá información privada de los usuarios en este trabajo, ni siquiera su nombre de usuario o ID asociado y en caso de detectar alguna infracción grave o que se solicitase la colaboración por parte de los cuerpos de seguridad del Estados, se cumpliría con el deber de colaboración.

3. Diseño del prototipo

En esta sección se detallan las fases seguidas para conseguir tener un prototipo que logre revisar los objetivos propuestos para el desarrollo de este proyecto. Se obtendrán los datos útiles de entrada para el entrenamiento, creación y análisis de modelos de ML, así como los datos contra los que se validará cada diseño.
También se estudiarán los resultados del entrenamiento y características de cada modelo para determinar si pueden encajar dentro de la solución final.

3.1 Flujo de trabajo

El proceso de desarrollo de esta prueba de concepto ha seguido las siguientes fases de trabajo:

1. Búsqueda de un *dataset* adecuado.
2. Extracción de tweets mediante la API de Twitter.
3. Procesamiento de los datos.
4. Entrenamiento modelos de ML.
5. Aplicación modelos entrenados a los tweets extraídos.

En los siguientes apartados se relata con más detalle cada fase mencionada.

3.1.1 Dataset

Una de las mayores dificultades que ha tenido el proyecto ha sido la localización de un conjunto de datos debidamente etiquetado.

El objetivo de búsqueda en la red social es la de sucesos negativos, eventos que puedan estar poniendo en peligro a cualquier individuo. Es por ello por lo que se ha pensado que la manera de expresar un suceso de la forma más clara o correcta es en la que se suele presentar en las noticias en los medios de comunicación y que, por tanto, encontrar un *dataset* con titulares de noticias podría ser un buen punto para poder entrenar algoritmos de ML.

El problema de este punto de partida es que no ha sido posible encontrar ningún conjunto de datos etiquetado con esas características, ni siquiera un conjunto sin etiquetar en español. Por lo que se ha optado por utilizar un set de datos que contiene más de 1 millón de titulares de noticias del medio ABC (*Australian Broadcasting Corporation*).

Esta información ha sido extraída de la comunidad de Kaggle [30] pero al no estar etiquetados se ha tenido que realizar el etiquetado con una lógica programada directamente en Python puesto que un etiquetado manual es inviable en el alcance de este proyecto.

La lógica implementada consiste en clasificar el titular de una noticia como negativa o positiva en función del mensaje transmitido. Para ello, se busca de entre todo el texto del titular si hay contenida en él al menos una palabra de un corpus con palabras catalogadas como negativas en el campo de análisis de sentimientos. Si es así, entonces se entiende que la noticia narra un suceso o evento nocivo y el titular se etiqueta con un 1, en caso contrario la noticia se contempla como un suceso neutral o positivo y se etiqueta con un 0.

El corpus utilizado contiene más de 4700 palabras en inglés, incluyendo palabras mal escritas con sus erratas frecuentes que aparecen comúnmente en diversos medios. Dicho corpus ha sido extraído de un blog de la UIC (*University of Illinois Chicago*) [31] [32].

En un primer análisis se utilizó un corpus mucho más pequeño, concretamente de 30 palabras, donde se incluían tan solo términos relacionados con sucesos concretos y triviales como pudiera ser “asesinato”, “criminal” o “secuestro”, por ejemplo. Esta forma tan determinista de etiquetar los datos provocaba que evidentemente se detectasen muchos menos titulares como negativos. En la siguiente figura se puede ver esta comparación:

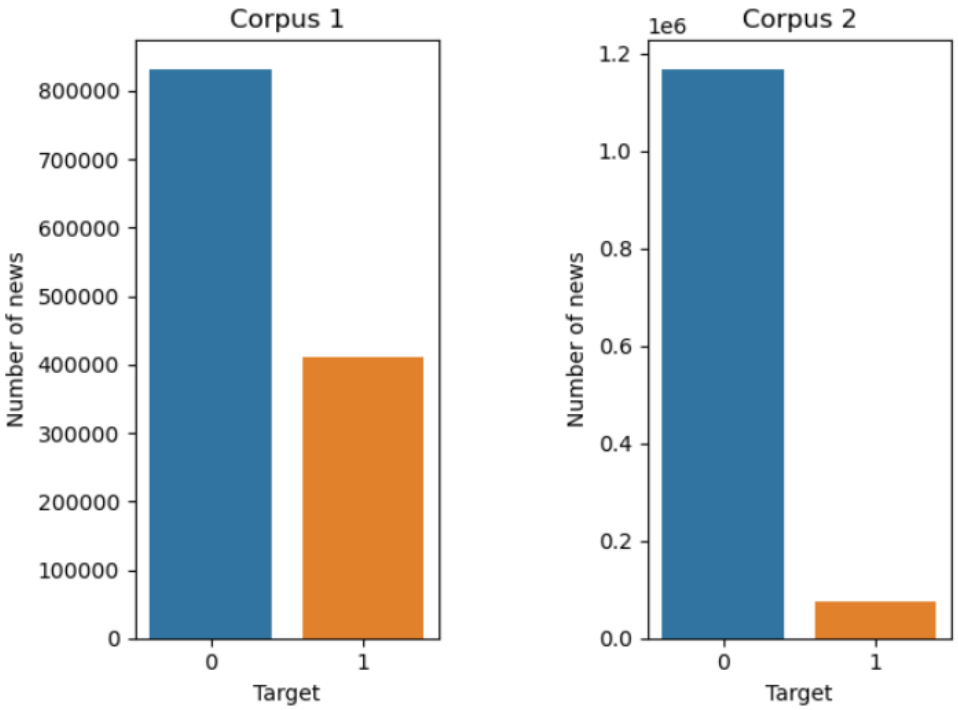


Figura 2. Comparación de etiquetado con 2 corpus distintos

Donde “Corpus 1” es aquel que contiene mayor número de términos y “Corpus 2” aquel que tan solo contiene 30 palabras.

Los resultados con los dos corpus son comparados en esta memoria en los apartados 3.3 Modelos de *Machine Learning* y 4.1 Pruebas de funcionamiento.

3.1.2 Extracción de tweets

Para tener la posibilidad de extraer información desde la API de Twitter es necesario disponer de una cuenta de desarrollador o habilitar una cuenta existente como una de desarrollador, no solo basta con tener una cuenta ordinaria en la aplicación.

Una vez hecho esto, a cada usuario, según el perfil contratado (en este caso se parte del perfil gratuito), se le otorgan una serie de claves y tokens de acceso para la conexión contra la API y que en este caso se hará desde el script de Python programado con ayuda de la librería ***tweepy***.

En un principio, el objetivo era poder llevar a cabo el prototipo extrayendo información de Twitter en español desde un área geográfica española determinada para el análisis, pues en el trabajo de investigación previo no se encontraron muchas referencias sobre ello a nivel nacional. Pero dado el problema existente con el etiquetado de datos, se ha decidido extraer los tweets de una región de habla inglesa, en este caso se ha optado por extraer tweets desde Londres en un radio de 50 km. También se filtran solo aquellos que estén indicados por Twitter que están escritos en inglés, ya que independientemente del área de extracción puede haber usuarios que escriban en distintos idiomas y eso implicaría el procesamiento de los datos en función de su idioma de entrada y elevaría la complejidad del producto sustancialmente.

Para el almacenamiento de la información recopilada, se han guardado los campos más relevantes de cada tweet en un tipo de estructura de datos muy utilizado en Python: formato DataFrame. Para así posteriormente exportarlos en formato CSV fácilmente si se quiere reutilizar el mismo conjunto de datos extraído.

3.1.3 Procesamiento de los datos

Una vez se han obtenido todos los datos a utilizar en este diseño se deberá aplicar una capa de preprocesamiento y procesamiento para poder limpiar dichos datos y llevar a cabo un análisis y entrenamiento de modelos de ML más limpio y eficiente.

Preprocesamiento
Se desarrolla una función que permita poder limpiar el texto de caracteres especiales (@, #, ^, …), espacios en blanco, letras en mayúscula, enlaces a páginas web, emoticonos, saltos de línea, números y signos de puntuación. De esta forma se libera a los algoritmos de analizar datos desconocidos o información innecesaria que pueda llevar a resultados inconclusos.

Procesamiento
Se han utilizado dos técnicas muy reconocidas en PLN y también en la subrama del análisis de sentimientos:

- **Eliminación de *stop words*.** Son palabras que no aportan ningún tipo de información a la hora de clasificar un texto o extraer de él algún análisis conductual de la persona creadora del mensaje. Para extraer estas palabras de los datos se ha utilizado el módulo ***stopwords*** de la librería ***corpus*** incluido en el paquete ***Natural Language Toolkit*** (NLTK) para Python. [33]
- **Aplicación de *stemming* (derivación en español).** Se trata de una técnica que tiene como objetivo reducir una palabra a su raíz sin sufijos o prefijos. Es similar a la técnica de lematización[10], pero en este caso las raíces no tienen por qué ser siempre palabras como base ni tampoco tienen que pertenecer a la misma clase. Ejemplo: de amamos, se extrae am- como base, en lugar de amam-. Para ser posible ejecutar esta técnica se ha hecho uso del módulo ***SnowballStemmer*** extraído también del paquete NLTK de Python. [34]

Tanto la capa de preprocesamiento como de procesamiento se aplica a los dos conjuntos de datos: noticias etiquetadas y tweets extraídos.

Cabe mencionar que el uso de la técnica de lematización se ha barajado para incluirla como parte de la solución en este trabajo, pero se descartó por implicar un alto coste computacional además de poder ofrecer resultados inesperados al ser una técnica probabilística y compleja de programar. [35]

3.1.4 Entrenamiento modelos de ML

Una vez se ha llevado a cabo todo el procesamiento de los datos y se tienen los campos de texto debidamente limpiados, se procede a entrenar varios modelos basados en algoritmos de ML para luego comparar resultados entre ellos.

Previamente al entrenamiento de cualquier modelo se deben separar los datos de entrada en conjunto de entrenamiento y conjunto de test (o prueba). Como en este caso queremos detectar o predecir la naturaleza de un tweet, este no tendrá ningún tipo de etiquetado y es por ello por lo que se deberá introducir dicho tweet como entrada al modelo una vez creado y obtener la salida de predicción correspondiente.

Por lo tanto, los datos que se tienen de los titulares de noticias etiquetados serán los que se utilicen para la creación y entrenamiento de cada modelo. Estos se dividirán en datos de entrenamiento y datos de test, en una proporción de 70% y 30%, respectivamente, para evitar sobreajuste (*overfitting*) o subajuste (*underfitting*) en los modelos[11]. El volumen de

[10] Consiste en hallar el lema de las palabras y agruparlas según raíces comunes/lexemas. En PLN se usa para tener menos variedad de palabras y agilizar las búsquedas y ejecución de algoritmos. Ejemplo: de cantar, cantaban o cantó, se extrae cant-.
[11] Tanto el sobreajuste como subajuste ocurren cuando el modelo no está siendo debidamente entrenado y de algún modo los datos que se están introduciendo como entrada provocan un desajuste al obtener un resultado de predicción excesivamente favorable o insuficiente según el caso de ajuste impreciso que se esté produciendo.

datos de entrada ha ido variando en distintas ejecuciones para poder obtener conclusiones reveladoras y válidas.

Los modelos creados para este trabajo son los siguientes:

- Naïve Bayes (NB).
- Regresión Logística (*Logistic Regression*, LR).
- Árbol de decisión (*Decision Tree*, DT).
- Bosque Aleatorio (*Random Forest*, RF).

En el apartado **3.3 Modelos de *Machine Learning*** se explica el funcionamiento de cada algoritmo con detalle.

Un aspecto importante a estudiar en la construcción de los modelos es que, dado que se están analizando fragmentos de texto, el análisis eleva su dificultad en gran medida y por ello se suele emplear un procedimiento que se conoce como vectorización.

La vectorización consiste en la conversión de datos en formato de texto a formato numérico. Para este trabajo se ha decidido usar en concreto el método de *Term Frequency-Inverse Document Frequencies* (TF-IDF) pues su uso es bastante extendido en la ciencia del análisis de sentimientos por sus buenos resultados. [36]

Este método calcula, por un lado, la frecuencia con la que un término aparece en un texto determinado y, por otro lado, el número de textos que contienen dicho término dentro de un set entero (es decir, lo común que es dicho término dentro del corpus). Estos dos cálculos son multiplicados de forma que se crea una "puntuación" para posteriormente poder construir una matriz de valores e introducirlos como entrada a cada uno de los modelos de *Machine Learning*. Para construir dicha matriz de valores se ha utilizado la función ***TfidfVectorizer*** de la librería ***Scikit-learn*** (SKLEARN) de Python.

La función ***TfidfVectorizer*** funciona del siguiente modo:

1. Se calcula la frecuencia de aparición de un término, $\boldsymbol{tf(t,d)}$, siendo $\boldsymbol{t}$ la frecuencia de aparición y $\boldsymbol{d}$ el número de veces que un término aparece en el documento[12].

2. Se calcula la frecuencia de documento inversa, $\boldsymbol{idf(t)}$, mediante la siguiente expresión:

$$\boldsymbol{idf(t) = \log \frac{1+n}{1+df(t)} + 1}$$

   Siendo $\boldsymbol{n}$ el número total de documentos y $\boldsymbol{df(t)}$ el número de documentos en el set de documentos que contiene el término $\boldsymbol{t}$.

[12] En este caso de estudio, un documento será una noticia para los datos de entrenamiento y un tweet para la validación de los modelos. Un set de documentos por tanto será el conjunto de noticias y el conjunto de tweets en cada uno de los conjuntos de datos recogidos.

3. Se calcula el producto de los dos términos hallados en los pasos 1 y 2:

$$\boldsymbol{tfidf(t,d) = tf(t,d) \cdot idf(t)}$$

4. El vector resultante es normalizado como:

$$v_n = \frac{v}{\|v\|^2} = \frac{v}{\sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}}$$

3.1.5 Aplicación modelos entrenados sobre tweets extraídos

Entrenados los modelos, se procede a validar cada uno de ellos introduciendo los tweets como entrada, también serán convertidos en vectores usando TF-IDF.

En esta validación se muestra el comportamiento real de cada modelo en comparación con su comportamiento teórico en el momento de su creación y es donde se podrá ver la verdadera precisión con la que se detectan tweets con información negativa referente a un evento desfavorable.

Todas estas fases seguidas para el diseño del prototipo de este TFM se pueden observar de forma sencilla en el siguiente esquema:

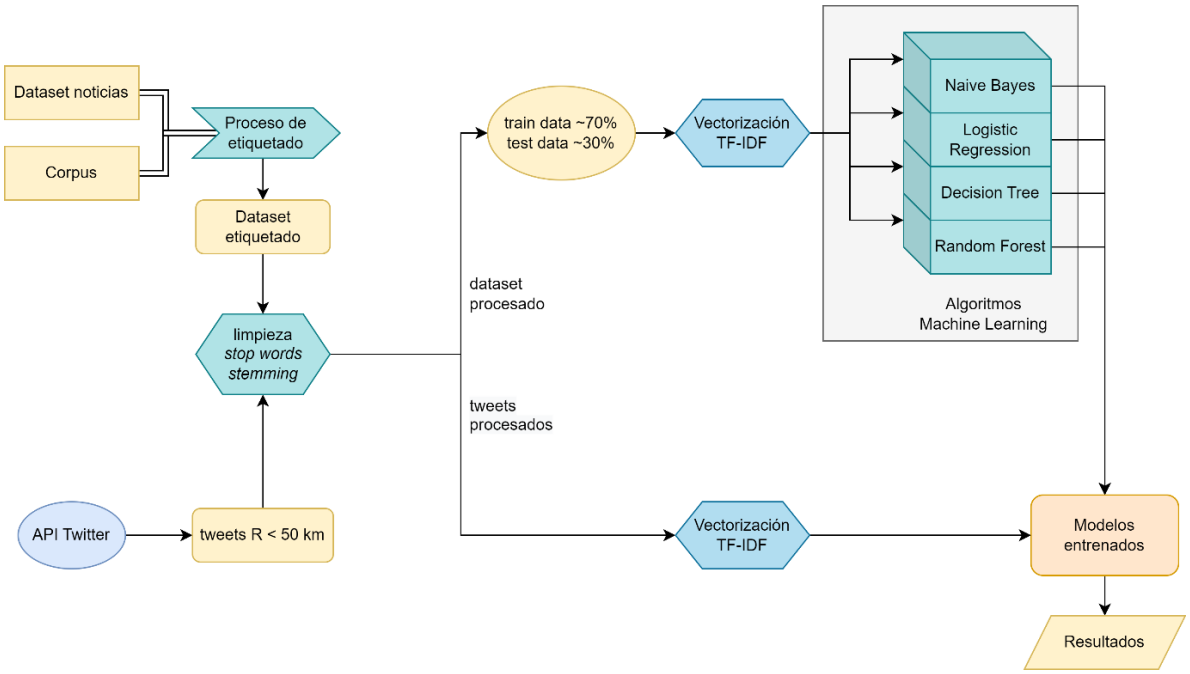


Figura 3. Esquema diseño de prototipo

3.2 Entorno de desarrollo

Todo el proceso de trabajo se ha llevado a cabo mediante la herramienta de Jupyter Notebook usando Python como lenguaje de programación.

El entorno se ha simplificado al máximo para facilitar la consecución de los objetivos y también porque, por un lado, la aplicación utilizada es bastante robusta y permite fácilmente integrar en ella lectura de ficheros de entrada para su procesamiento con el análisis en Python en sí. Por otro lado, la extracción de información desde Twitter se podía hacer de una forma muy eficiente con librerías de Python y es un lenguaje ampliamente utilizado en el campo de la Inteligencia Artificial.

Además, al ser Jupyter Notebook un entorno interactivo, se pueden mostrar resultados de forma gráfica sin necesidad de un software externo de apoyo.

La integración del prototipo con la participación de más herramientas en un entorno más complejo que incluya también almacenamiento externo se plantea como líneas de trabajo futuro pues no se dispone actualmente de la capacidad y tiempo para ello.

3.3 Modelos de *Machine Learning*

En este apartado se pretende explicar el funcionamiento de cada algoritmo aplicado, así como también comparar los resultados al entrenar y crear cada uno de los modelos.

3.3.1 Conceptos clave

Antes de explicar la mecánica de todos los modelos y sus algoritmos, conviene echar un vistazo a algunos conceptos importantes que son de utilidad para analizar la validez de cada modelo y entender mejor el porqué de los resultados logrados[13].

Matriz de confusión

Refleja los resultados de un modelo gracias a la detección de los aciertos (verdaderos positivos y verdaderos negativos) y los fallos (falsos positivos y falsos negativos). Es decir, si se analiza un texto en el que se quiere detectar si se describe un suceso producido o no, se tendrá:

- **Verdadero positivo** si ha habido un suceso y se detecta.
- **Falso positivo** si no ha habido un suceso y se detecta.
- **Falso negativo** si ha habido un suceso y no se detecta.

[13] Este apartado se decide colocar en esta sección de la memoria en lugar de en el capítulo 2 (Estado del arte) ya que facilita la lectura de la memoria, pues son conceptos muy específicos y que cobran más importancia a raíz de este capítulo 3.

- **Verdadero negativo** si no ha habido un suceso y no se detecta.

La siguiente figura muestra el ejemplo de una matriz de confusión:

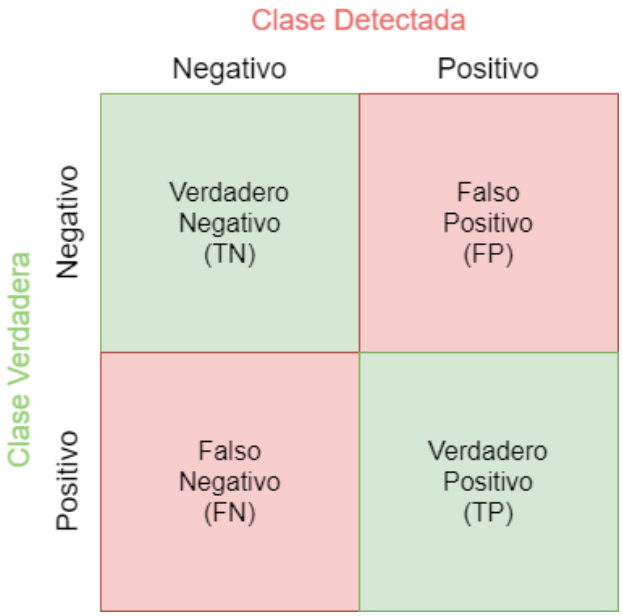


Figura 4. Matriz de confusión

Exactitud o *Accuracy*

Mide la habilidad con la que el modelo detecta las clases de interés correctamente. El modo de hallar este valor es dividiendo el número de verdaderos positivos más el número de verdaderos negativos entre el total, es decir:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Precisión

Mide la cantidad de predicciones positivas correctas. Se calcula dividiendo los verdaderos positivos entre la suma de los verdaderos positivos y los falsos positivos, o sea:

$$Precision = \frac{TP}{TP + FP}$$

Exhaustividad o *Recall*

Mide la cantidad de aciertos reales entre todos los positivos verdaderos. Es decir, la cantidad de verdaderos positivos entre la suma de los verdaderos positivos y los falsos negativos o:

$$Recall = \frac{TP}{TP + FN}$$

F1-score o puntuación F1

Es el valor que indica la puntuación total conseguida por la precisión y exhaustividad de un modelo. Es muy útil cuando se quieren comparar varios modelos teniendo en cuenta la precisión y exhaustividad total de cada modelo por igual, sin dar preferencia a ninguna de las dos medidas por ser a la vez importantes para el análisis. La expresión para calcular esta puntuación es:

$$F1\_score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Curva ROC (*Receiver Operating Characteristic*) y AUC (*Area Under Curve*)

La curva ROC mide la calidad de las predicciones de un modelo cuando hay un caso de estudio de eventos binarios, como es el objeto de este trabajo. Su representación se realiza midiendo la tasa de verdaderos positivos (sería lo mismo que la exhaustividad), TPR, frente a la tasa de falsos positivos (o la tasa de falsa alama, es decir la probabilidad de detectar falsos positivos entre todos los negativos verdaderos), FPR. Se tendría entonces:

$$TPR = \frac{TP}{TP + FN} = Recall$$

$$FPR = \frac{FP}{TN + FP}$$

En un clasificador ideal, el valor de TPR sería de 1 y el de FPR de 0 pero dado que normalmente esta situación no da lugar, estaremos ante una gráfica que a partir de cierto umbral tomará el valor de 1 o 0 y normalmente se establece este umbral en 0,5. Por encima de este valor se clasificará la predicción como 1 y por debajo de este como 0, salvo que por circunstancias específicas se deba establecer otro límite en concreto.
Si todo esto se dibuja en una gráfica de dos dimensiones, TPR frente a FPR, el objetivo será focalizarse en los comportamientos extremos, es decir, fijarse si la curva forma casi un rectángulo, pues cuanto más se asemeje, mejor modelo se habrá obtenido, al menos teóricamente.

Ahora bien, el valor de AUC es el área bajo la curva ROC y dado que esta área solo puede estar comprendida entre 0 y 1, cuanto más cercano a 1 sea dicho valor, mejor modelo se estará construyendo.

3.3.2 Naive Bayes

Un clasificador basado en Naive Bayes es muy útil para catalogar datos de textos, y, por tanto, es muy útil en el análisis de sentimientos, pues como se ha podido apreciar en el estado de arte del proyecto, son varios los trabajos que han hecho utilidad de este clasificador con resultados muy favorables.

Este algoritmo está basado en el famoso Teorema de Bayes, donde se pretende estudiar cómo la probabilidad de que suceda un evento A determinado influye en la probabilidad de

que suceda otro evento B, pudiendo ser ambos eventos independientes o dependientes. La ecuación en la que está basado dicho teorema puede expresarse de la siguiente manera:

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{P(B)}$$

Ahora bien, para la construcción del clasificador se necesita calcular la clase a la que pertenecen una serie de valores. En este caso específico, se requiere calcular si un texto pertenece a la clase 1 o a la clase 0. Y así con todos los demás. La siguiente expresión combina el Teorema de Bayes con la resolución de este problema:

$$P(Y_i|X_1, X_2, \dots, X_N) = \frac{P(Y_i) \cdot P(X_1, X_2, \dots, X_N|Y_i)}{P(X_1, X_2, \dots, X_N)}, \quad i = 1{,}2;\ N = [1, n]$$

Donde $P(Y_i)$ representa la probabilidad de cada clase y $P(X_1, X_2, \dots, X_N)$ la probabilidad de producirse dichos valores. Para este algoritmo, se asume que son eventos independientes. [37]

Una vez entendido el planteamiento, se procede a programar el modelo de ML basado en este algoritmo gracias al uso de la librería de ***Scikit-learn*** de Python.

```
# Model
nb_model = MultinomialNB()
nb_model.fit(x_train_vector, y_train)

y_pred = nb_model.predict(x_test_vector)
y_prob = nb_model.predict_proba(x_test_vector)[:,1]
```

Figura 5. Creación modelo Naive Bayes Python

Primero se ajustan los vectores de las muestras de entrenamiento para luego predecir los resultados con las muestras de prueba que se recogieron del DataFrame con las noticias etiquetadas, siendo $\boldsymbol{x}$ el vector de los campos de texto e $\boldsymbol{y}$ el vector con las etiquetas.

Con estos valores podemos generar una predicción de la clasificación de las noticias (y_pred) y la probabilidad de estimación para ese vector x (x_test_vector).

Se puede apreciar que se está haciendo uso de un clasificador con distribución multinomial, esto es debido a que para clasificación de textos se ajusta mejor que una distribución de Bernoulli o Gaussiana.

Con este modelo, se obtienen los siguientes resultados utilizando el Corpus 1 con 100.000 noticias de cada clase:

Clase	Precisión	Recall	F1-score	Exactitud
0	93,08%	85,73%	89,26%	89,73%
1	86,89%	93,69%	90,16%	

Tabla 1. Resultados entrenamiento Naive Bayes con Corpus 1

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	25601 (TN)	4260 (FP)
	1903 (FN)	28236 (TP)

Tabla 2. Matriz de confusión modelo Naive Bayes con Corpus 1

Y la siguiente curva ROC:

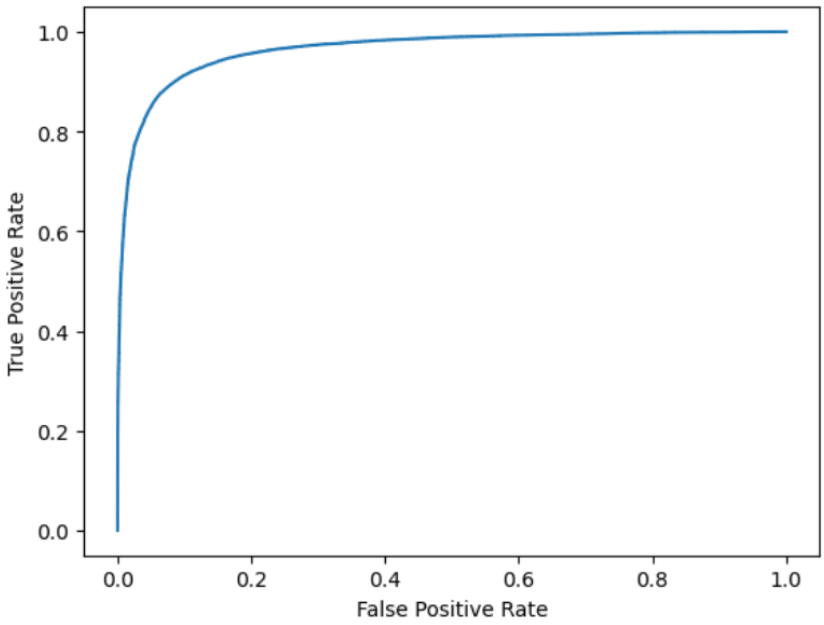


Figura 6. ROC Naive Bayes Corpus 1

Con $AUC = 0{,}9651$.

Si se utiliza el Corpus 2 con 75.000 noticias de cada clase, estos son los resultados del modelo entrenado:

Clase	Precisión	Recall	F1-score	Exactitud
0	92,78%	87,59%	90,11%	90,45%
1	88,40%	93,28%	90,77%	

Tabla 3. Resultados entrenamiento Naive Bayes con Corpus 2

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	19579 (TN)	2773 (FP)
	1523 (FN)	21125 (TP)

Tabla 4. Matriz de confusión modelo Naive Bayes con Corpus 2

Y la siguiente curva ROC:

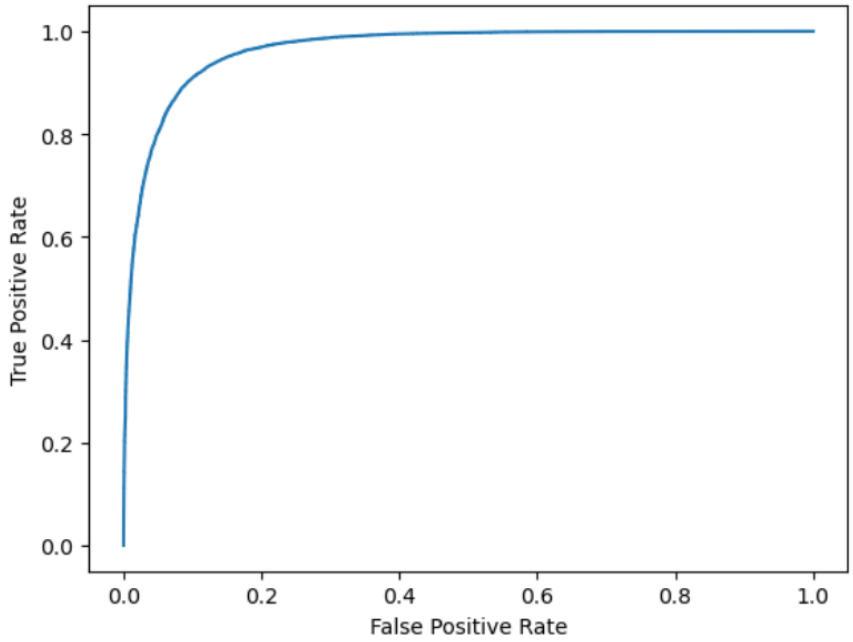


Figura 7. ROC Naive Bayes Corpus 2

Con $AUC = 0{,}9657$.

Se puede observar que este modelo obtiene resultados muy similares en su entrenamiento para los dos corpus, por lo que puede ser un indicio de que los tweets detectados para los dos corpus también sean similares entre sí o al menos guardar alguna característica común.

Por los datos obtenidos, es un buen modelo para este tipo de análisis de clasificación de textos tal y como se presuponía.

Como punto a destacar, es observable que este modelo es más preciso midiendo la exhaustividad de la clase 1 que su precisión y la precisión de la clase 0 que su exhaustividad, dado que se quieren detectar los sucesos de la clase 1, esto nos está indicando que hay un porcentaje más alto de hallar los eventos que se han producido y son detectados que los que se detectan y realmente son sucesos de interés correctos, por lo que a priori es algo muy bueno para el estudio.

3.3.3 *Logistic Regression*

El algoritmo de Regresión Logística también se emplea a modo de construir un clasificador para el aprendizaje supervisado en *Machine Learning* y es muy útil para el caso que ocupa este proyecto al tratarse de un problema de clasificación entre dos clases distintas.

El funcionamiento de este algoritmo se basa en relacionar distintos valores de $\boldsymbol{x}$ de entrada con su predicción a la salida $\boldsymbol{y}$ aplicando una función logística $\boldsymbol{\sigma}$. A los valores de $\boldsymbol{x}$ se les pueden asignar distintos pesos $\boldsymbol{w}$ mediante la multiplicación por coeficientes en función de la información que aporte cada variable de entrada, en este caso, se dejarán los valores por defecto al no tener mucha más información. [38]

Una vez se tiene la combinación lineal de los valores de $\boldsymbol{x}$ con sus pesos $\boldsymbol{w}$ se aplica la función logística $\boldsymbol{\sigma}$ y se predice la salida $\boldsymbol{y}$. Este algoritmo puede expresarse mediante la siguiente fórmula matemática:

$$Y = \sigma(Z) = \sigma(X \cdot W) = \sigma\left(\sum (x_i \cdot w_i)\right)$$

En esencia, lo que se está buscando con este algoritmo es la probabilidad de obtener 1 o 0 dado $\boldsymbol{X}$, es decir:

$$P(Y = 1|X)\ y\ P(Y = 0|X)$$

Ahora bien, para programar el modelo de ML basado en este algoritmo se usarán de nuevo funciones de la librería de ***Scikit-learn*** de Python. Y su lógica consistirá en las siguientes líneas mostradas en la Figura 8:

```
# Model
lr_model = LogisticRegression(solver = 'lbfgs', max_iter=150, C=1, penalty = 'l2')
lr_model.fit(x_train_vector, y_train)

y_pred = lr_model.predict(x_test_vector)
y_prob = lr_model.predict_proba(x_test_vector)[:,1]
```

Figura 8. Creación modelo Logistic Regression Python

La lógica seguida es la misma aplicada que para el modelo de Naive Bayes, con la salvedad de que en este caso se le asignan distintos parámetros de configuración adicionales:

- ***solver*:** método de optimización aplicado, se deja por defecto a L-BFGS.
- ***max_iter:*** número de iteraciones, siendo como mínimo 100.
- ***C:*** valor que regulariza la penalización de los parámetros optimizados en el modelo. Cuanto más pequeño sea, más robusto el resultado y menor la probabilidad de sobreajustar.
- ***penalty:*** sirve para regularizar la función de perdidas mediante una penalización que evite un sobreajuste del modelo teniendo en cuenta las magnitudes de los coeficientes.

Con este modelo, se obtienen los siguientes resultados utilizando el Corpus 1 con 100.000 noticias de cada clase:

Clase	Precisión	Recall	F1-score	Exactitud
0	91,93%	95,77%	93,81%	93,71%
1	95,63%	91,68%	93,61%	

Tabla 5. Resultados entrenamiento Logistic Regression con Corpus 1

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	28598 (TN)	1263 (FP)
	2509 (FN)	27630 (TP)

Tabla 6. Matriz de confusión modelo Logistic Regression con Corpus 1

Y la siguiente curva ROC:

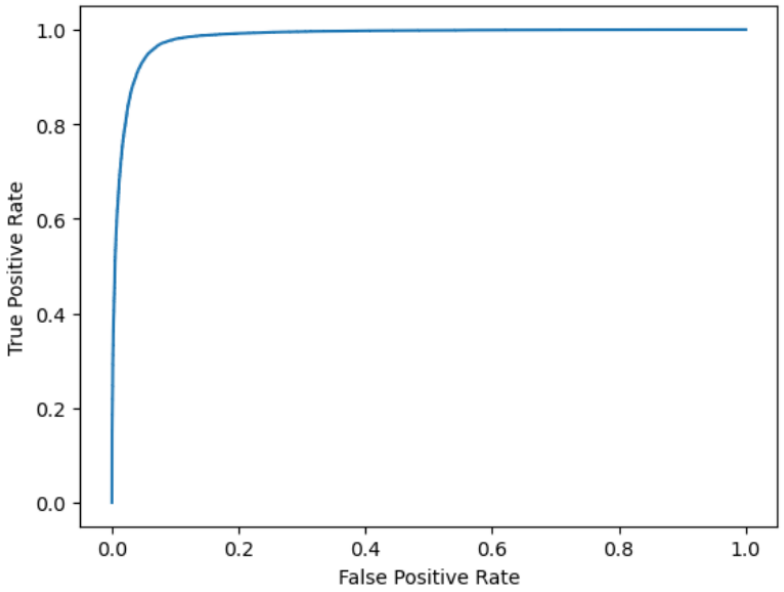


Figura 9. ROC Logistic Regression Corpus 1

Con $AUC = 0{,}9840$.

Si se utiliza el Corpus 2 con 75.000 noticias de cada clase, estos son los resultados del modelo entrenado:

Clase	Precisión	Recall	F1-score	Exactitud
0	99,82%	97,98%	98,89%	98,91%
1	98,04%	99,82%	98,93%	

Tabla 7. Resultados entrenamiento Logistic Regression con Corpus 2

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	21901 (TN)	451 (FP)
	40 (FN)	22608 (TP)

Tabla 8. Matriz de confusión modelo Logistic Regression con Corpus 2

Y la siguiente curva ROC:

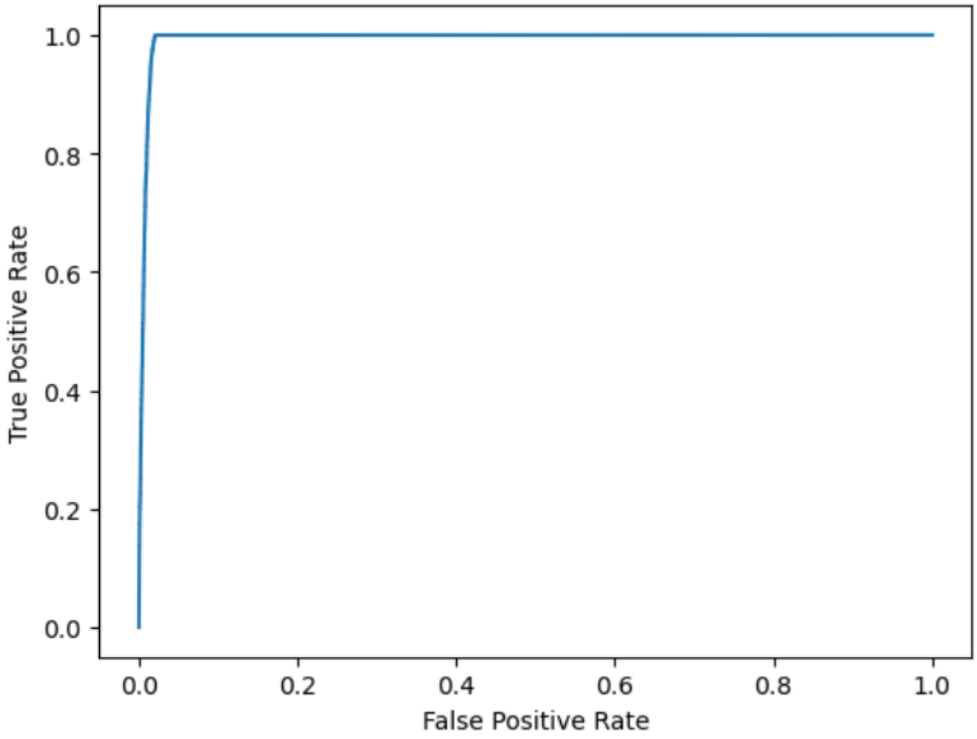


Figura 10. ROC Logistic Regression Corpus 2

Con $AUC = 0{,}9943$.

Salta a la vista que los resultados obtenidos con el segundo corpus son algo irreales o demasiado perfectos. Si se fija la atención sobre la curva ROC se ve que es prácticamente un rectángulo y su AUC está a muy pocas cifras de alcanzar la unidad, probablemente este modelo consiga unos resultados demasiado simplistas o deterministas.

Por otro lado, el modelo construido con el etiquetado del primer corpus obtiene unos resultados muy buenos también, no tan perfectos, pero sí que mejoran en parte a los obtenidos con el modelo de Naive Bayes.

3.3.4 *Decision Tree*

Un árbol de decisión consiste en recorrer de manera recursiva un proceso de clasificación en función de los datos de entrada pudiendo dividir los resultados en otras subramas de posibilidades para seguir recorriendo el árbol hasta dar con la solución.

Para este caso, se podían optar por dos formas de clasificar y dividir en subramas cada nodo del árbol: mediante el cálculo del índice de Gini o mediante el cálculo de la entropía. Dado que durante el estudio del modelo las pruebas que se hicieron obtuvieron resultados casi

idénticos, se optó por utilizar el primer método al obtener ligeramente unas décimas más de precisión.

Este método calcula las probabilidades de que cada clase se haya clasificado incorrectamente y por lo tanto cada rama del árbol terminará con un resultado de éxito o fracaso hasta el final del recorrido de forma que se minimice lo máximo posible el valor del resultado, es decir, cuando se aproxime lo máximo posible a 0 el índice de Gini y por consiguiente, la probabilidad de clasificación incorrecta. Se puede expresar con la siguiente fórmula:

$$G_{index} = \sum_{i=1}^{N} P(i) \cdot (1 - P(i))$$

En un primer momento, el uso de un algoritmo basado en árboles de decisión no estaba contemplado en el trabajo, pero al estar obteniendo resultados con precisiones tan altas cabía la posibilidad de tener en algún punto del desarrollo del prototipo algún tipo de sobreajuste que no se estaba logrando detectar. Es por ello por lo que se consideró como buena idea implementar este modelo para ver la evolución de los valores de entrenamiento y de prueba según se incrementa el parámetro de profundidad, $\boldsymbol{max\_depth}$, hasta el que llega el árbol. Con cada iteración y división del problema, lo habitual es que el algoritmo se haga más preciso, pero hay que tener cuidado con no caer en el sobreentrenamiento. [39]

Observando la gráfica para el primer corpus:

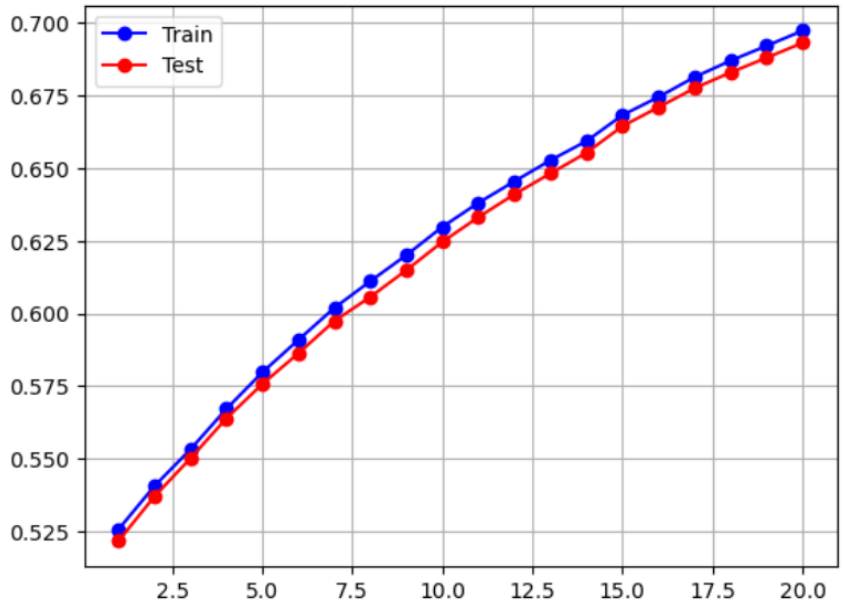


Figura 11. Comparación puntuaciones obtenidas modelo Decision Tree Corpus 1

Y para el segundo corpus:

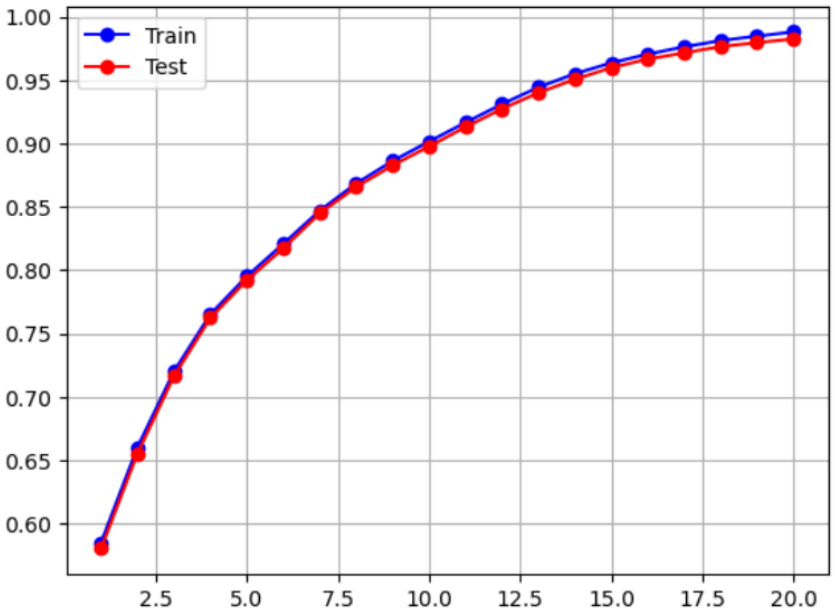


Figura 12. Comparación puntuaciones obtenidas modelo Decision Tree Corpus 2

Se ve que aparentemente (o teóricamente) no está habiendo ningún tipo de *overfitting* o *underfitting* (aunque para el segundo corpus las puntuaciones son demasiado buenas), por lo que parece que el planteamiento ejecutado hasta ahora es correcto. Ahora bien, los resultados logrados con este modelo no son tan buenos en comparación con el resto como se comentará más adelante.

Para programar el modelo de ML basado en este algoritmo se usarán de nuevo funciones de la librería de ***Scikit-learn*** de Python. Y su lógica consistirá en las siguientes líneas mostradas en la Figura 13:

```
# Create model with best parameters
# Model
dt_model = DecisionTreeClassifier(criterion='gini', max_depth=20) # With best max_depth value in performance before
dt_model.fit(x_train_vector, y_train)

y_pred = dt_model.predict(x_test_vector)
y_prob = dt_model.predict_proba(x_test_vector)[:,1]
```

Figura 13. Creación modelo Decision Tree Python

Donde se tendrá en cuenta el valor del parámetro $\boldsymbol{max\_depth}$ que haya logrado un mejor rendimiento.

Utilizando el Corpus 1 con 100.000 noticias de cada clase se tiene:

Clase	Precisión	Recall	F1-score	Exactitud
0	62,07%	98,55%	76,16%	69,30%
1	96,55%	40,33%	56,89%	

Tabla 9. Resultados entrenamiento Decision Tree con Corpus 1

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	29427 (TN)	434 (FP)
	17984 (FN)	12155 (TP)

Tabla 10. Matriz de confusión modelo Decision Tree con Corpus 1

Y la siguiente curva ROC:

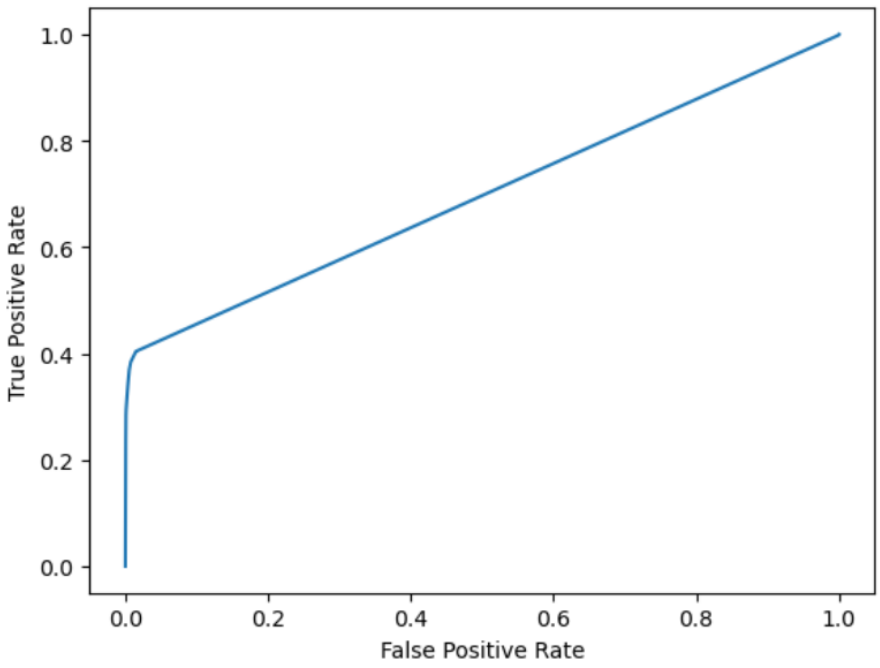


Figura 14. ROC Decision Tree Corpus 1

Con $AUC = 0.6963$.

Si se utiliza el Corpus 2 con 75.000 noticias de cada clase, estos son los resultados del modelo entrenado:

Clase	Precisión	Recall	F1-score	Exactitud
0	98,25%	98,30%	98,27%	98,28%
1	98,32%	98,27%	98,30%	

Tabla 11. Resultados entrenamiento Decision Tree con Corpus 2

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	21971 (TN)	381 (FP)
	391 (FN)	22257 (TP)

Tabla 12. Matriz de confusión modelo Decision Tree con Corpus 2

Y la siguiente curva ROC:

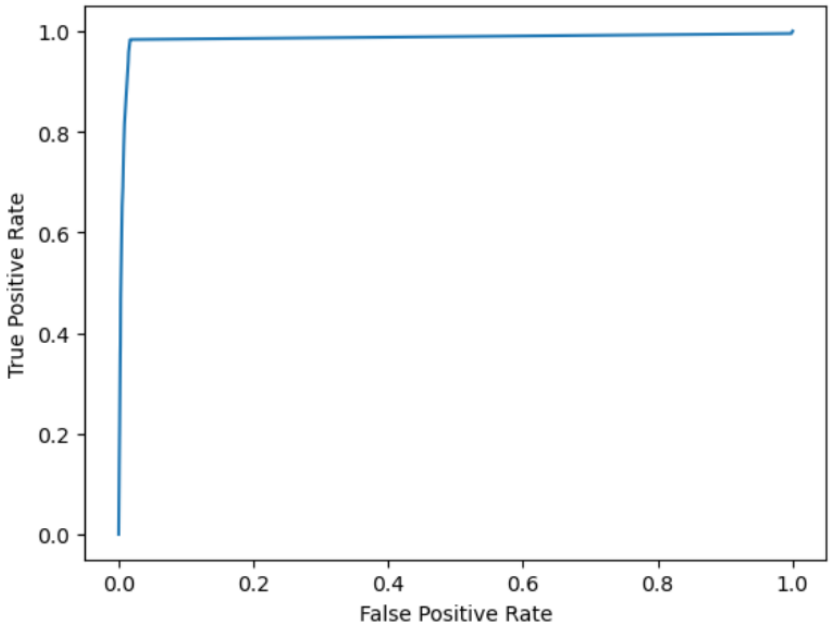


Figura 15. ROC Decision Tree Corpus 2

Con $AUC = 0{,}9845$.

Este modelo refleja perfectamente el comportamiento de un modelo con resultados aleatorios, pues con un corpus más completo y que debería enriquecer el entrenamiento de

dicho modelo se obtienen unos resultados muy malos, aunque la exactitud final roce el 70% el resto de los valores reflejan que su respuesta en la predicción de resultados puede ser totalmente impredecible (una exhaustividad para la clase 1 del 40%, por ejemplo). La curva dibujada de la ROC también muestra este comportamiento aleatorio al ser prácticamente una recta.

Respecto al modelo generado con un corpus más reducido, se ve que en comparación los resultados rozan la perfección, muy similar a lo ocurrido con LR, pero en esta ocasión hay mucha diferencia entre cada modelo entrenado y creado. Definitivamente este modelo no es una buena elección en el análisis y los tweets detectados serán poco significativos e impredecibles para el problema a resolver.

3.3.5 *Random Forest*

Este algoritmo está basado en árboles de decisión, la diferencia principal es que este algoritmo no solo construye un solo árbol sino varios con el mismo criterio de decisión. De este modo, en cada árbol se terminará por obtener un resultado para cada predicción y es al llegar al final de cada árbol donde se elegirá la clase que más veces haya salido como predicción. La siguiente gráfica muestra este proceso de una forma sencilla:

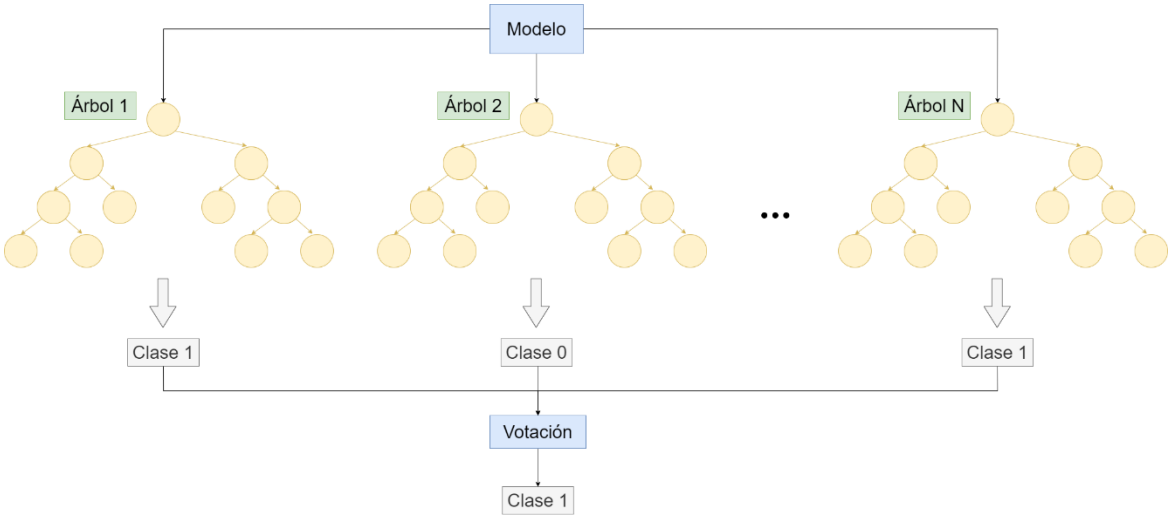


Figura 16. Funcionamiento algoritmo Random Forest

Para la construcción de este modelo, se ha seguido un planteamiento muy similar al de la elaboración del modelo de un solo árbol de decisión. Es decir, se pretende observar si el modelo está siendo sobreentrenado de alguna forma conforme la profundidad de los árboles (pues en este caso serán varios) va aumentando para así escoger el último valor en el que no está produciéndose ningún tipo de sobreajuste o subajuste en el modelo si es que lo hubiera.
Pero como en este caso hay más de un árbol, se puede elegir el número de estos involucrados a la hora de construir el modelo. Gracias al estimador ***GridSearchCV*** de la

librería ***Scikit-learn*** de Python se pueden aproximar los mejores parámetros para este modelo en función de los datos de entrada. [40] [41]

Para ello, se van a estimar dos parámetros clave: $\boldsymbol{max\_depth}$ y $\boldsymbol{n\_estimators}$, es decir, como se comentaba, la profundidad de cada árbol y el número de árboles. La manera en la que se hará sigue la lógica siguiente:

```
# Params to estimate
params_grid = {
    'max_depth': [i for i in range(1, 21)],
    'n_estimators': [100, 200, 300, 500]
}

# Create model
rf_model = RandomForestClassifier()

# Create the grid search model
grid_search = GridSearchCV(estimator=rf_model, params_grid=param_grid, cv=3, n_jobs=-1)
grid_search.fit(x_train_vector, y_train)
best_params = grid_search.best_params_
best_params
```

Figura 17. Estimación parámetros modelo Random Forest

Los dos valores adicionales que se pasan como parámetros de la función indican lo siguiente:

- $\boldsymbol{cv}$: parámetro que utiliza el método de validación cruzada por debajo para separar los datos de entrenamiento en distintos grupos según el número indicado.
- $\boldsymbol{n\_jobs}$: indica el número de procesos que corren en paralelo, en este caso el -1 indica que se están ejecutando el número máximo de procesos posible a la vez.

Con los datos etiquetados del primer corpus se tiene que los parámetros óptimos son los siguientes:

- $\boldsymbol{max\_depth = 20}$
- $\boldsymbol{n\_estimators = 300}$

Y si se dibuja la gráfica en función de la profundidad del árbol construido obtenemos:

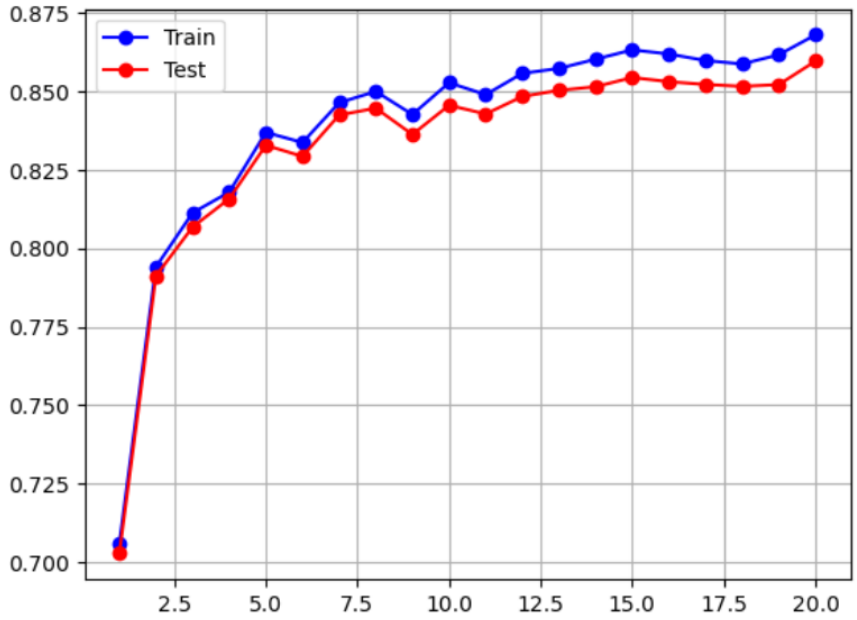


Figura 18. Comparación puntuaciones obtenidas modelo Random Forest Corpus 1

Se puede apreciar que, aunque la precisión del modelo va al alza en general, con árboles con una profundidad mayor de 5, los valores empiezan a tener máximos y mínimos nuevos por lo que cabe la posibilidad de que se esté produciendo algún tipo de *overfitting* y para prevenirlo, se escogerá el último valor antes de que eso ocurra de forma frecuente. En este caso será una profundidad de 8, así se deja de margen algún mínimo y se alcanza un punto donde apenas mejoran los datos del modelo y se estabiliza antes de volver a tener precisiones bajando y subiendo constantemente.

Si se hace lo mismo para el segundo corpus se obtienen los siguientes parámetros:

- $\boldsymbol{max\_depth = 18}$
- $\boldsymbol{n\_estimators = 500}$

Y la siguiente gráfica de evolución en función de la profundidad:

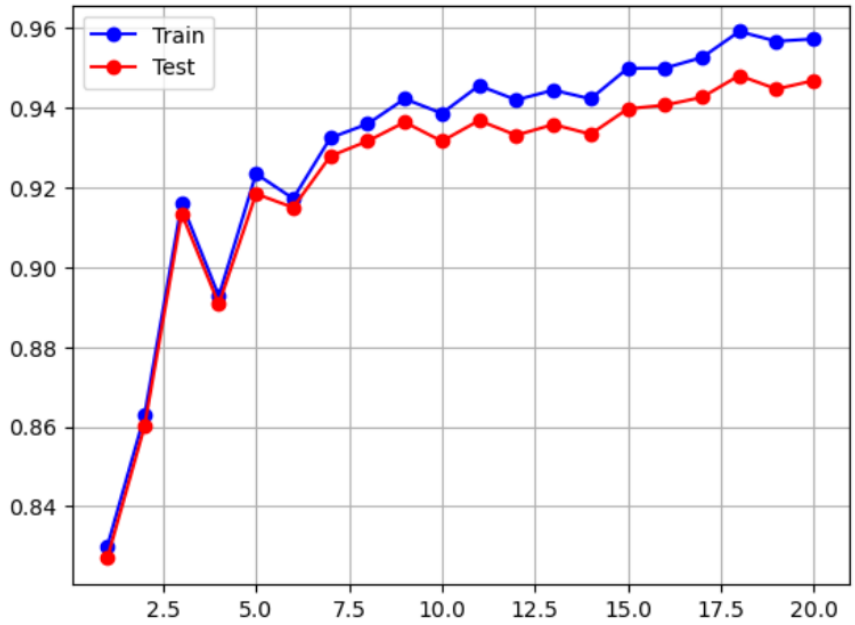


Figura 19. Comparación puntuaciones obtenidas modelo Random Forest Corpus 2

Donde se puede observar que se tiene un comportamiento parecido, en este caso se escoge un valor de profundidad de 9 para prevenir o evitar *overfitting*.

La construcción del modelo final se hará por tanto teniendo en cuenta los parámetros óptimos:

```
# Create model with best parameters
# Model with best max_depth and n_estimators values in performance before
rf_model = RandomForestClassifier(max_depth=8,n_estimators=best_params['n_estimators'])
rf_model.fit(x_train_vector, y_train)

y_pred = rf_model.predict(x_test_vector)
y_prob = rf_model.predict_proba(x_test_vector)[:,1]
```

Figura 20. Creación modelo Random Forest Python (caso Corpus 1)

Utilizando el Corpus 1 con 100.000 noticias de cada clase se tiene:

Clase	Precisión	Recall	F1-score	Exactitud
0	77,46%	96,24%	85,83%	84,19%
1	95,09%	72,25%	82,11%	

Tabla 13. Resultados entrenamiento Random Forest con Corpus 1

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	28737 (TN)	1124 (FP)
	8364 (FN)	21775 (TP)

Tabla 14. Matriz de confusión modelo Random Forest con Corpus 1

Y la siguiente curva ROC:

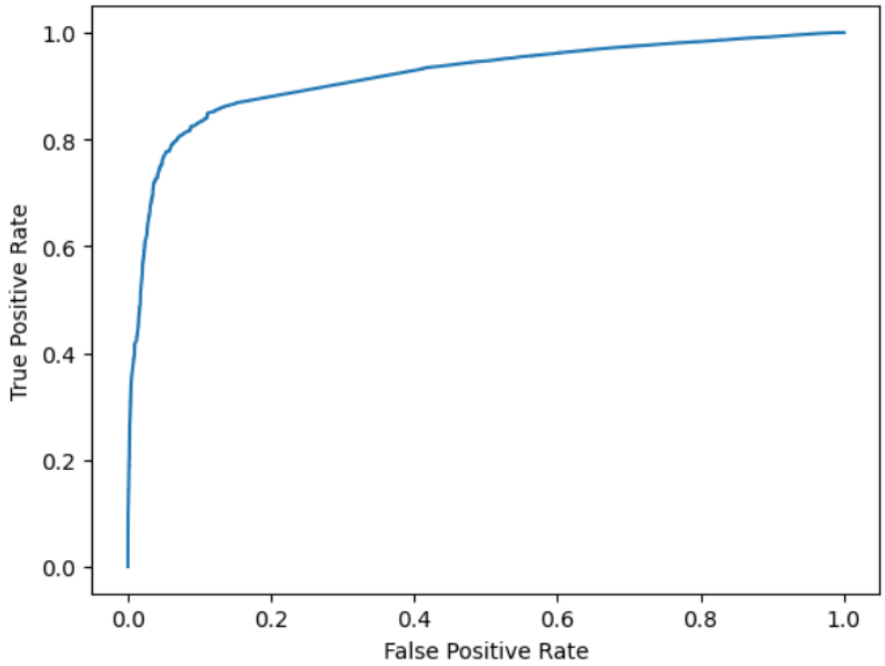


Figura 21. ROC Random Forest Corpus 1

Con $AUC = 0{,}9184$.

Si se utiliza el Corpus 2 con 75.000 noticias de cada clase, estos son los resultados del modelo entrenado:

Clase	Precisión	Recall	F1-score	Exactitud
0	91,32%	94,20%	92,74%	92,67%
1	94,09%	91,16%	92,60%	

Tabla 15. Resultados entrenamiento Random Forest con Corpus 2

Con la siguiente matriz de confusión:

	Clase detectada	
Clase verdadera	21055 (TN)	1297 (FP)
	2001 (FN)	20647 (TP)

Tabla 16. Matriz de confusión modelo Random Forest con Corpus 2

Y la siguiente curva ROC:

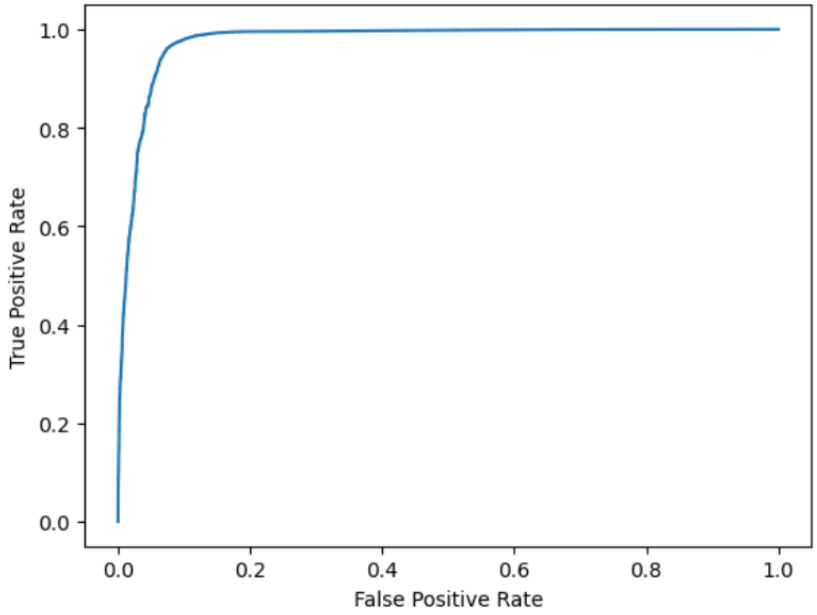


Figura 22. ROC Random Forest Corpus 2

Con $AUC = 0{,}9768$.

Al contrario de lo que sucedía con un solo árbol de decisión, ahora sí se obtienen resultados satisfactorios pues la precisión alcanzada es muy buena y es indicativo de que puede ser capaz de predecir eventos en tweets a la hora de validar el diseño. Tanto con un corpus amplio como con uno reducido está logrando conseguir buena puntuación global en precisión y exhaustividad, o sea si nos fijamos en el valor de f1-score. En el primer modelo, en global, se están consiguiendo unos resultados más realistas o "reservados" en detección.

Cabe mencionar que en comparación al resto de modelos se están produciendo un gran número de falsos negativos lo que indica que probablemente no se detecten tweets de sucesos a pesar de deber ser catalogados como tal.

4. Análisis y resultados

4.1 Pruebas de funcionamiento

En este apartado se expondrán los resultados obtenidos para cada modelo con los tweets extraídos desde la API de Twitter. Se han extraído un total de 100.000 tweets para la validación de las pruebas finales.

Para facilitar el visionado de los resultados se dividirá este apartado en más subapartados para organizar las pruebas según el modelo que se esté validando. Solo se recogerán los resultados y se indicará a qué corresponde cada uno de ellos debidamente y será en la próxima sección 4.2 Validación del diseño donde se extraerán conclusiones de todos los resultados.

4.1.1 Resultados con modelo Naive Bayes

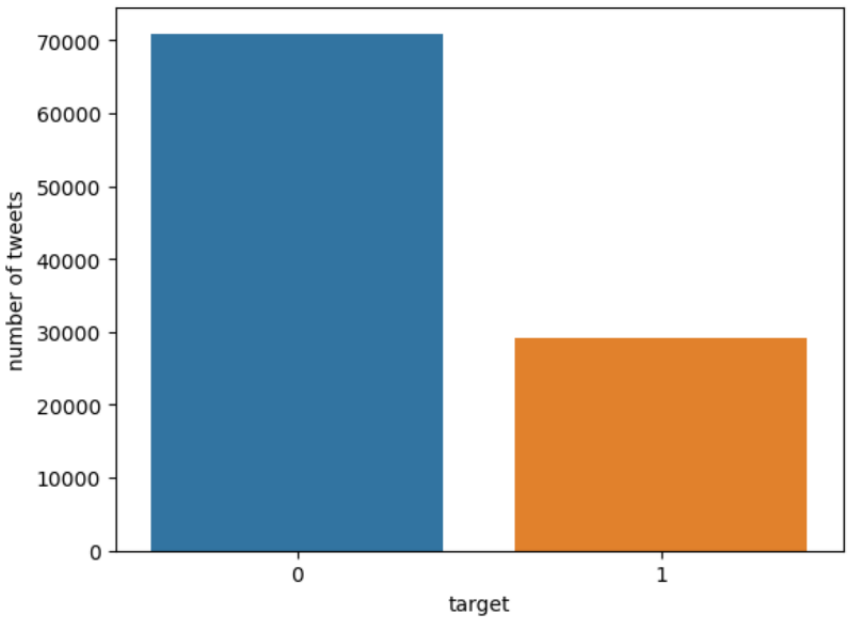
4.1.1.1 Con corpus 1

Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto	Probabilidad
1	lol and vaxxxed die b the crash bs again	99,99%
2	as more people die by suicide in the uk during this cost of living crisis will we simply say they were mentally ill or will we question if they died at the hands of racism poverty and violent austerity	99,91%
3	joblessness and poverty is killing my ppl forreal	99,91%
4	crash we s a l u t e y o u	99,87%
5	die hard	99,87%
6	how bad did the devils hurt you	99,81%
7	you re rejecting me	99,81%
8	died	99,81%
9	he s so fluffy i m gonna die	99,81%
10	i am dying	99,81%

Tabla 17. 10 primeros tweets Naive Bayes Corpus 1

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    70910
1    29073
Name: target, dtype: int64
```

Figura 23. Distribución Naive Bayes Corpus 1

Histograma del número de tweets detectado según sus probabilidades de predicción:

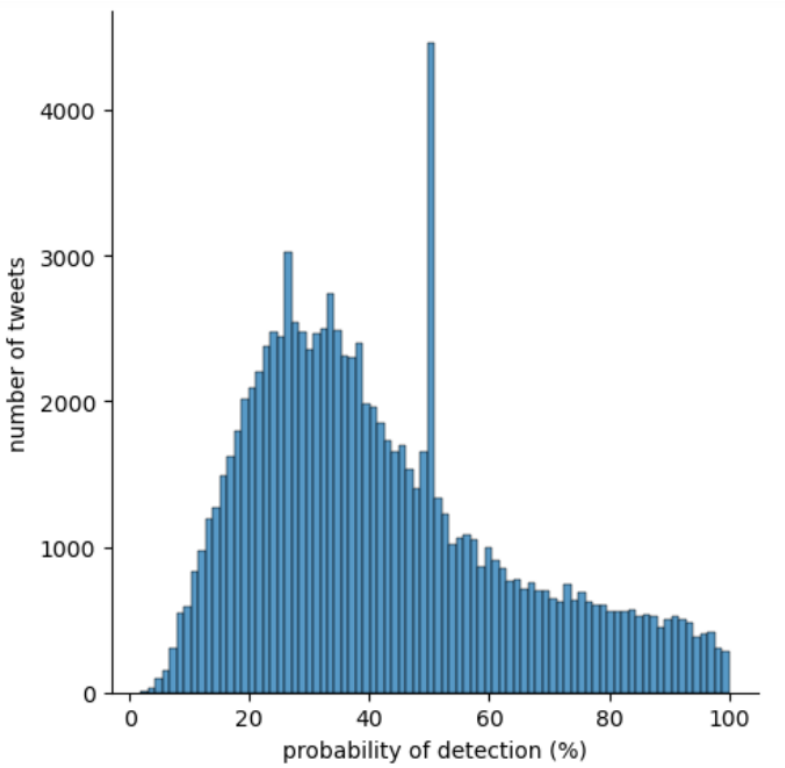


Figura 24. Histograma probabilidades Naive Bayes Corpus 1

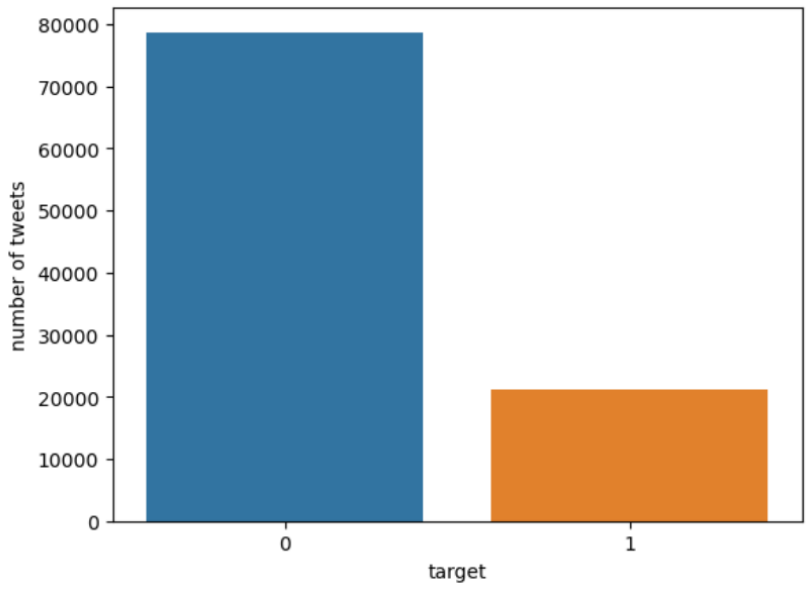
4.1.1.2 Con corpus 2

Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto (preprocesado)	Probabilidad
1	a monger of war for congrats	99,67%
2	the tylenol murders	99,51%
3	tomatoes r on their wsy to murdering me	99,51%
4	jail for the rapist death sentence for killers	99,40%
5	some more of your sick brothers uncles and murdering father zionist militias confessing their war crimes	99,27%
6	murderers not obama obv	99,24%
7	have they been convicted of said war crimes	99,19%
8	lompoc man pleads guilty to murder of ex girlfriend	99,19%
9	huhhuhuu i though he was dead	99,07%
10	g i m dead	99,07%

Tabla 18. 10 primeros tweets Naive Bayes Corpus 2

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    78703
1    21280
Name: target, dtype: int64
```

Figura 25. Distribución Naive Bayes Corpus 2

Histograma del número de tweets detectado según sus probabilidades de predicción:

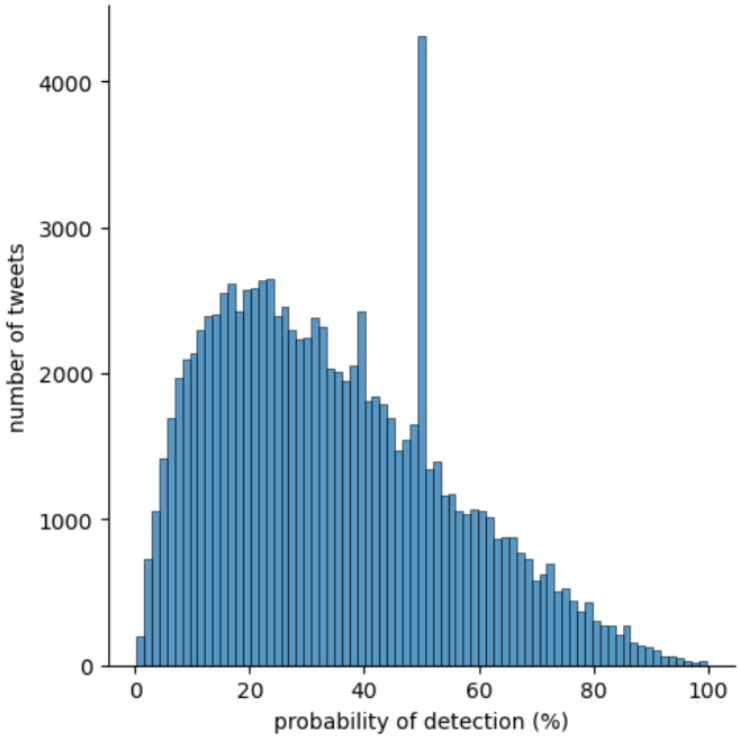


Figura 26. Histograma probabilidades Naive Bayes Corpus 2

4.1.2 Resultados con modelo *Logistic Regression*

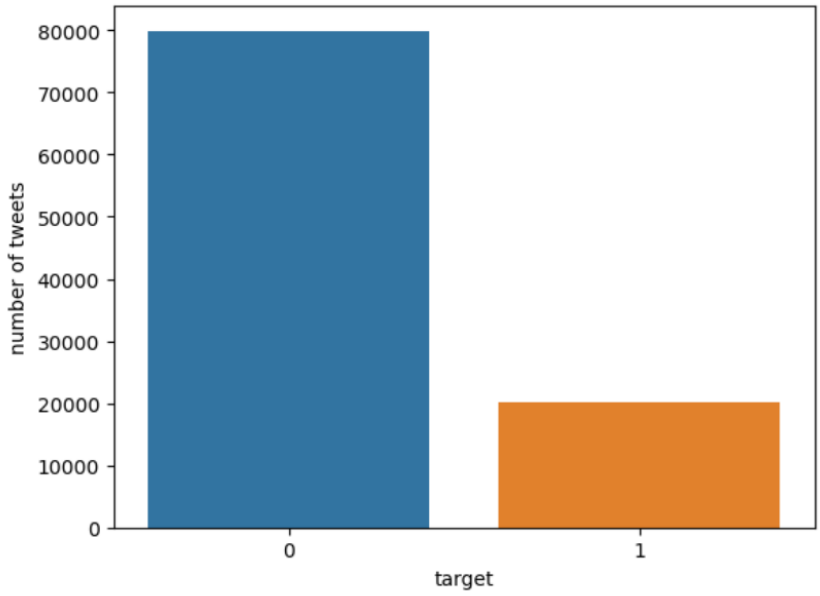
4.1.2.1 Con corpus 1

Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto	Probabilidad
1	lol and vaxxxed die b the crash bs again	99,99%
2	yep my fomo will kill me	99,99%
3	kill billl	99,99%
4	you re killing me	99,99%
5	kill him then	99,99%
6	she s gonna kill you	99,99%
7	killing myself	99,99%
8	this has killed me	99,99%
9	u fucking kill me looooool	99,99%
10	this is killed me ahaha	99,99%

Tabla 19. 10 primeros tweets Logistic Regression Corpus 1

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    79917
1    20066
Name: target, dtype: int64
```

Figura 27. Distribución Logistic Regression Corpus 1

Histograma del número de tweets detectado según sus probabilidades de predicción:

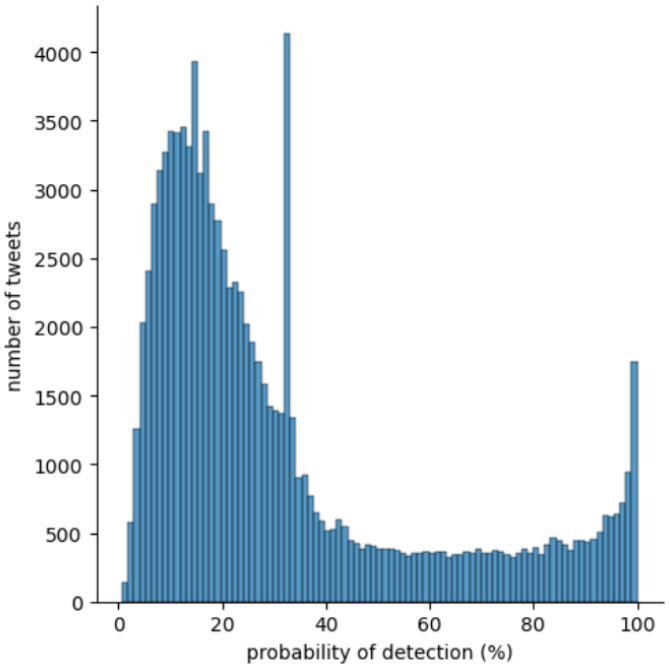


Figura 28. Histograma probabilidades Logistic Regression Corpus 1

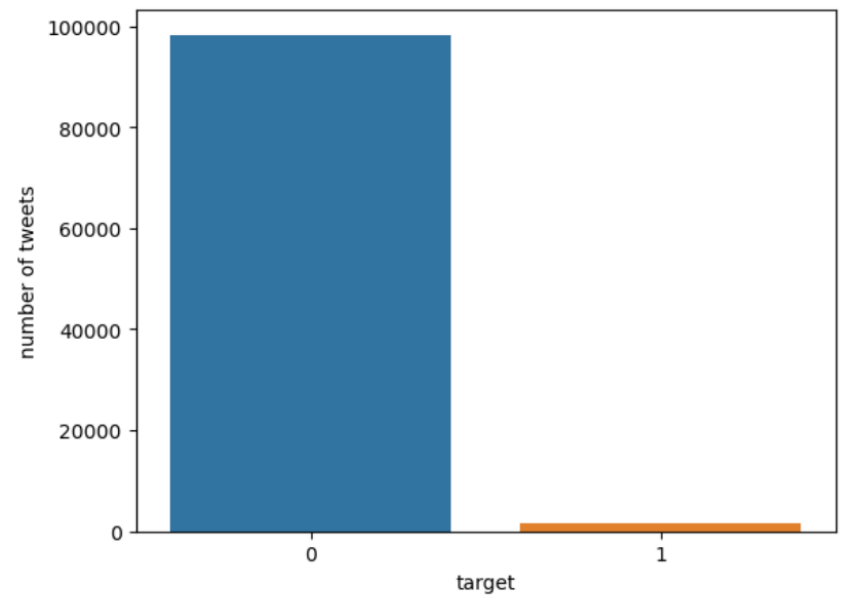
4.1.2.2 Con corpus 2

Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto (preprocesado)	Probabilidad
1	the tylenol murders	99,99%
2	tomatoes r on their wsy to murdering me	99,99%
3	a monger of war for congrats	99,99%
4	huhhuhuu i though he was dead	99,99%
5	is dead	99,99%
6	johan djourou he was too dead	99,99%
7	dead	99,99%
8	g i m dead	99,99%
9	you ll pay for these war crimes	99,99%
10	kill billl	99,99%

Tabla 20. 10 primeros tweets Logistic Regression Corpus 2

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    98290
1     1693
Name: target, dtype: int64
```

Figura 29. Distribución Logistic Regression Corpus 2

Histograma del número de tweets detectado según sus probabilidades de predicción:

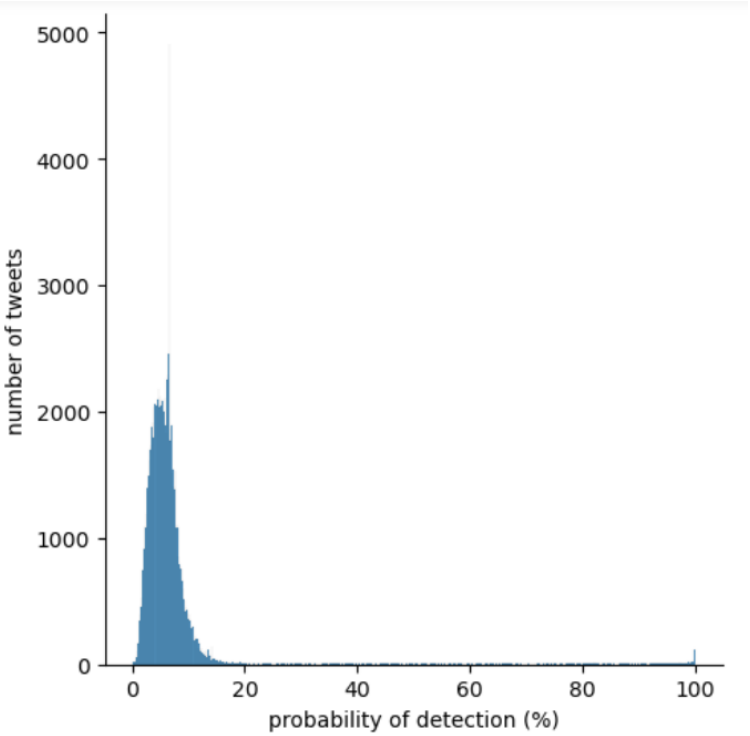


Figura 30. Histograma probabilidades Logistic Regression Corpus 2

4.1.3 Resultados con modelo *Decision Tree*

4.1.3.1 Con corpus 1

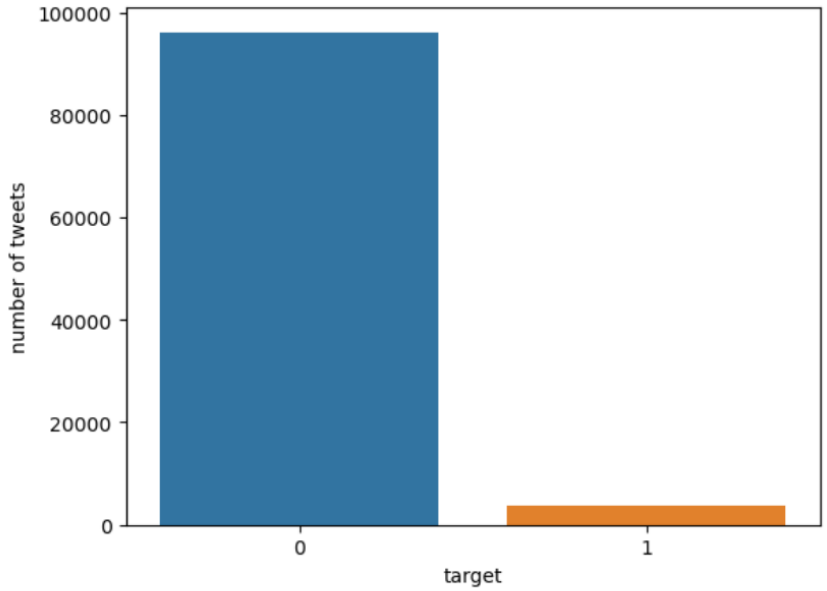
Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto	Probabilidad
1	its a hill i m willing to die on	100%
2	online sales fell in november outstripping the fall in retail sales for the month as the cost of living crisis hit consumers and inflation rises gripped the high street	100%
3	hi david thanks for the message we do not give refunds for delays outside our control including strikes thanks jl	100%
4	losing him will be felt	100%
5	biggestttttt i already fear crowds because air does not travel down i don t think people understand when you re small it is so scary	100%
6	it s a nightmare i ve started booking phone appointments so my bipolar ass doesn t lose its temper which isn t pretty i would rather stay polite so	100%

	don t put myself in a position where it happens hope the funeral goes well this is stuff you really don t need today hugs xx	
7	district line severe delays between tower hill and ealing broadway minor delays between turnham green and richmond while we fix a signal failure at gloucester road good service on the rest of the line tickets are being accepted on london buses	100%
8	so nobody actually died because that has been troubling me praying for the people in hospital	100%
9	i wonder when professional clowns will join the complaining crowd to protest against coulrophobia fear of clowns maybe it is time for another irrelevant socio political issue	100%
10	the bon appÃ©tit test kitchen bubble bursting was bigger than any economic crash	100%

Tabla 21. 10 primeros tweets Decision Tree Corpus 1

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    96132
1     3851
Name: target, dtype: int64
```

Figura 31. Distribución Decision Tree Corpus 1

Histograma del número de tweets detectado según sus probabilidades de predicción:

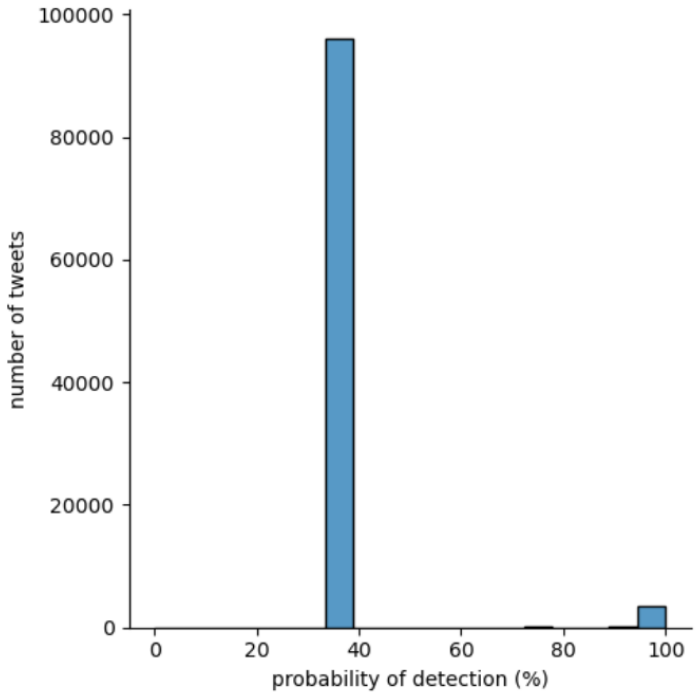


Figura 32. Histograma probabilidades Decision Tree Corpus 1

4.1.3.2 Con corpus 2

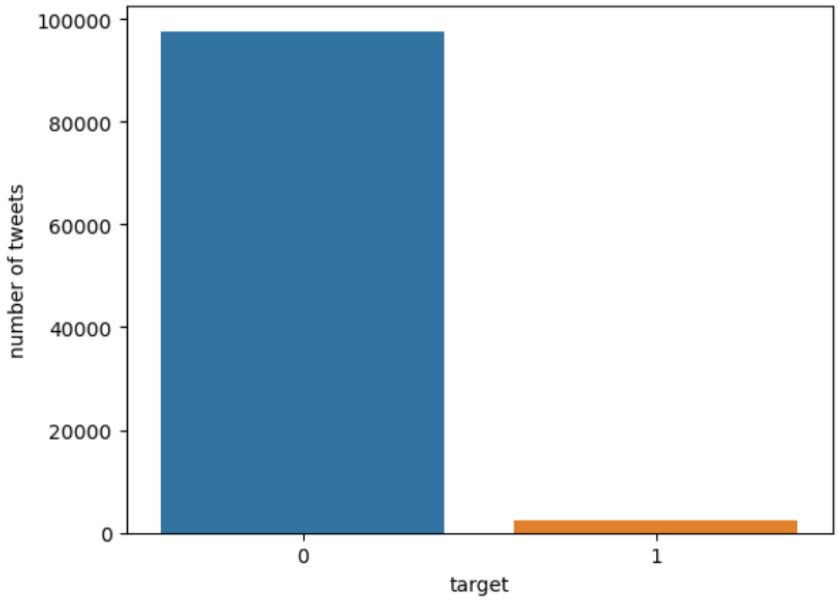
Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto (preprocesado)	Probabilidad
1	i m half italian half irish we lovingly debate each other over dinner for sport in our family always a laugh never ends in tears why does everyone on here think it s war	100%
2	belarus strongly supported by russia has been waging a hybrid war against numerous neighbours including poland and ukraine for years	100%
3	yeah and start a nuclear war genius idea	100%
4	goal to have endless war not successful war	100%
5	russia ukraine war at a glance what we know on day of the invasion	100%
6	/ the russian constructions follow traditional military plans for entrenchment largely unchanged since the second world war such constructions are	100%

	likely to be vulnerable to modern precision indirect strikes	
7	excited to discuss naved bakali and s important new edited collection the rise of global islamophobia in the war on terror on a panel inc contributors amp and chaired by pm jan zoom register	100%
8	news on pandemics war and politics will be always covered in lies never trust a thing you hear on tv	100%
9	on march th we warned of the stalinisation of russia under putin s war in political and social terms it may be necessary to go back almost a century to find a parallel to when stalin liquidated the entrepreneurial class to consolidate his power	100%
10	no climate urgentist is credible if they are not talking about preparations likely we need huge defence spending mobility impoverishment make oil expensive protect asset rights invasions and crime coming any breakdown in society will mean there is no stopping	100%

Tabla 22. 10 primeros tweets Decision Tree Corpus 2

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    97623
1     2360
Name: target, dtype: int64
```

Figura 33. Distribución Decision Tree Corpus 2

Histograma del número de tweets detectado según sus probabilidades de predicción:

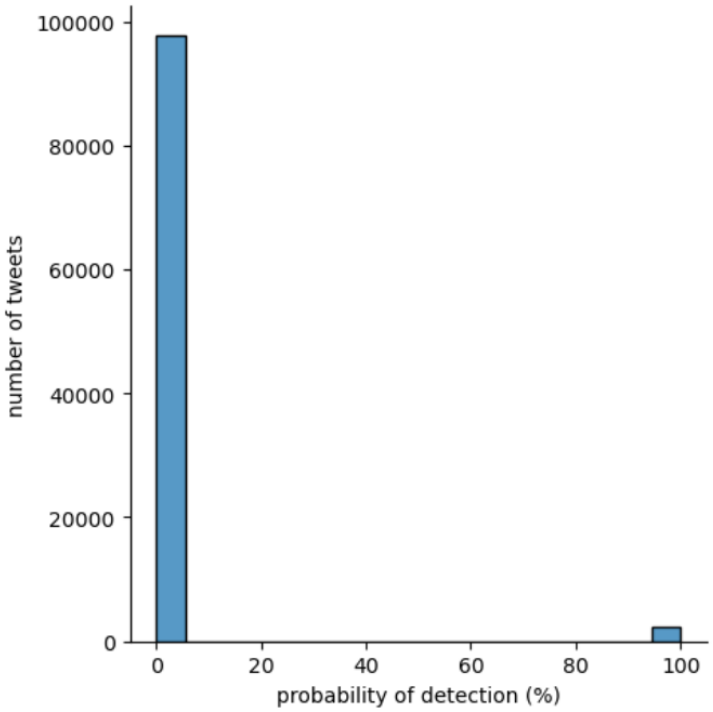


Figura 34. Histograma probabilidades Decision Tree Corpus 2

4.1.4 Resultados con modelo *Random Forest*

4.1.4.1 Con corpus 1

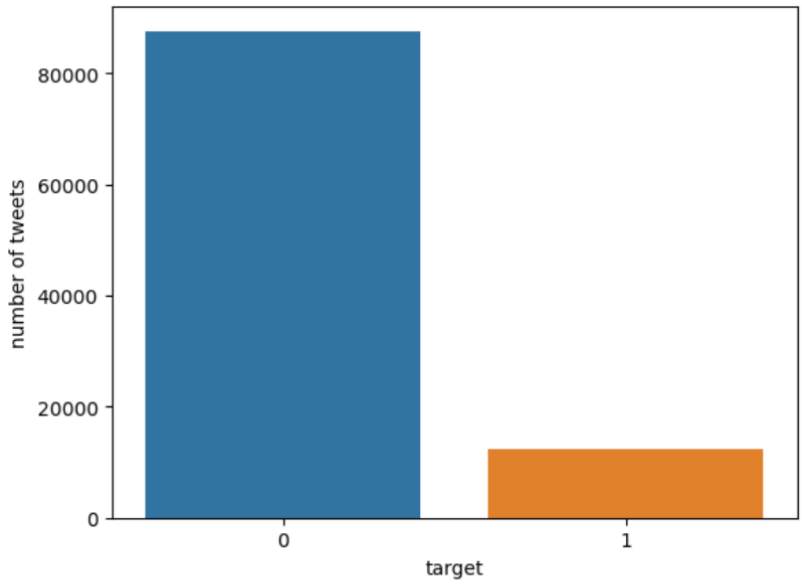
Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto	Probabilidad
1	respected concerned departments this woman seeking justice as she is suffering from physical assualt and threats of honor killing she presented docked medical but police didn t register fir against culprits due to corruption and mal administration plz take legal action regards	60,41%
2	it s hard to think that one of the most toxic proteins ever created is circulating in your blood and might kill you or cause cancer anytime soon i ve never seen in the scientific literature another toxin with so many potential mechanisms of disease or death	59,38%
3	excess cancer deaths due to delays in diagnostics and treatment since march people dying unnecessarily because of conservative failure to invest	58,97%

4	fears grow in african union for future of crisis wracked seperatist failed state off european coast as extremist far right regime tries to starve and bully nurses rail workers and postal staff into accepting poverty wages	58,72%
5	such is the pitiful state of the con party that in spite of johnson s failings his lying incompetence and corruption this kleptocrat has the support of why do they see something of themselves in him or do they simply lack the self respect required to reject a liar	58,56%
6	more people have died to left wing extremism in history of you want to go there socialist regimes led by mao hitler and stalin killed hundreds of millions and your ad hominem fails too because i am not american	58,42%
7	you know if this was the other way round it would be shocking revelation of aggressive harry attacking stunned william harry keeps guilty silence and refuses to deny his atrocious behaviour	58,21%
8	madam vice president it s also important to keep an eye on the continuous abuse of human rights young men and women are tortured abducted others even get killed for no crimes committed there only crime is to raise up their political views	58,09%
9	someone is on the verge of death but all you care about is to take your time to read his whatsapp chat how are we sure you re even telling the truth what if you legit killed him after you found out he cheated on you you stood there amp watch a human die without remorse	57,99%
10	why is it so hard to believe that what harry said is false the rf tell lies all the time the biggest one was over qe death according to giles brandeth who claims the queen died of cancer william spoke out before harry and lied saying it was a joint statement	57,90%

Tabla 23. 10 primeros tweets Random Forest Corpus 1

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    87594
1    12389
Name: target, dtype: int64
```

Figura 35. Distribución Random Forest Corpus 1

Histograma del número de tweets detectado según sus probabilidades de predicción:

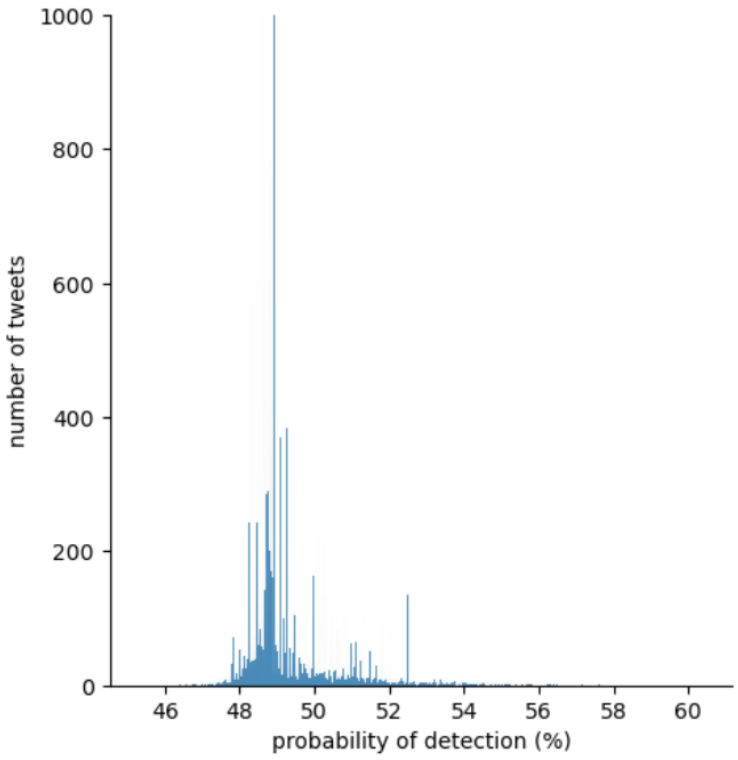


Figura 36. Histograma probabilidades Random Forest Corpus 1

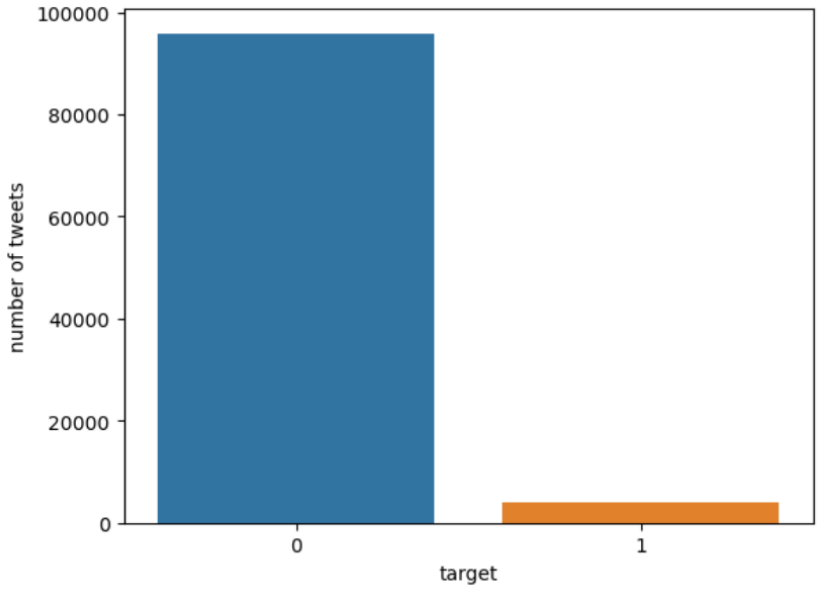
4.1.4.2 Con corpus 2

Los 10 primeros tweets detectados ordenados por probabilidad de predicción descendiente:

Orden	Texto	Probabilidad
1	officers have been charged in the deadly arrest of a black man in louisiana these are the first charges to emerge from a death authorities initially blamed on a car crash long suppressed body camera video showed white officers beating stunning	62,40%
2	i was just listening to murder case where the defense attorney argued that a video of a man the accused choking a woman the murder victim was no big deal because stuff like that is all over movies and television	61,36%
3	isn t amitabh bachhan the same man who said khoon ka badla khoon after indira gandhi s death just before the riots in which thousands of innocent sikhs were killed by murderous crowds of congis fascism lol you haven t seen the real face of it	59,63%
4	and the people the al huwaiti driven off their land one shot dead three sentenced to death by specialised criminal court	59,13%
5	some more of your sick brothers uncles and murdering father zionist militias confessing their war crimes	58,30%
6	at least two people are injured after two cars crash into each other and into a house	58,11%
7	no arrests have been made after a woman was found stabbed to death at her home in stoke newington	58,02%
8	no arrests have been made after a woman was found stabbed to death at her home in stoke newington	58,02%
9	ah yes that fine tradition of kneecapping bombs disappearing animal kidnapping bank robberies and child killing	58,00%
10	university of idaho students kaylee goncalves maddie mogen and xana kernodle and xana s boyfriend ethan chapin were stabbed to death as they slept four weeks later there are no named suspects murder weapon or motive	57,79%

Tabla 24. 10 primeros tweets Random Forest Corpus 2

Distribución de los resultados según la clase a la que pertenece cada tweet detectado:




```
0    95878
1     4105
Name: target, dtype: int64
```

Figura 37. Distribución Random Forest Corpus 2

Histograma del número de tweets detectado según sus probabilidades de predicción:

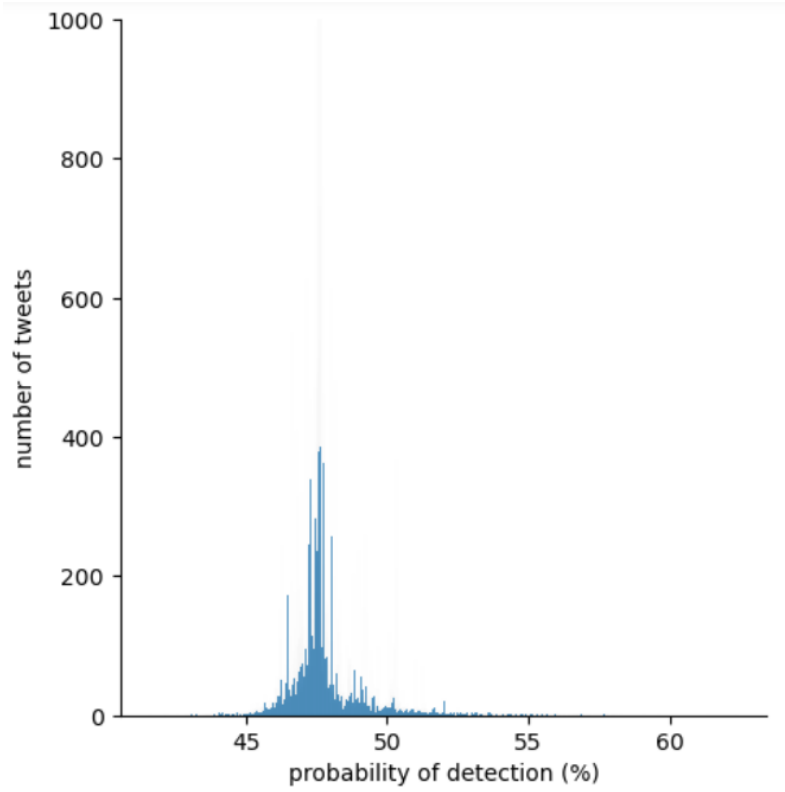


Figura 38. Histograma probabilidades Random Forest Corpus 2

4.2 Validación del diseño

Una vez vistos los resultados obtenidos para cada modelo y cada corpus se procede a realizar una validación del prototipo en función de estos.

Lo que más llama la atención en los resultados es la exagerada probabilidad de acierto que tienen los modelos de NB, LR o DT. Por encima del 99% en los 10 primeros tweets extraídos, lo que debería indicar que esos tweets están narrando algún suceso casi con total seguridad y quitando un par de sucesos como la condena de un crimen y hechos sobre la guerra entre Ucrania y Rusia el resto de los tweets son todos sobre opiniones, en ocasiones sobre un debate aparentemente importante y otras ocasiones frases con un perceptible tono irónico.

Otro detalle que destaca es la fijación del algoritmo LR con la palabra "*kill*" pues parece que en esta predicción la ha tomado como principal referencia y en parte puede ser producido este efecto por el método utilizado para la vectorización, TF-IDF, pues cabe recordar que se basaba en la búsqueda de frecuencias de aparición de los términos en las entradas de la función.

Si se presta atención en las distribuciones no se logra detectar en el mejor de los casos como positivos más de un ~29% de los 100.000 tweets en la entrada (~29.000 con el modelo de Naive Bayes para el primer corpus). Siendo por tanto enorme la diferencia de proporción de no detección (*target* a 0) frente a la de detección (*target* a 1) para todos los casos.

Observando los histogramas de NB, LR o DT, se puede apreciar como la mayor cantidad de tweets que no pasan el umbral del 50% de probabilidad se concentran en probabilidades de detección muy bajas (por debajo del 40%). En Naive Bayes se producen picos cercanos a la probabilidad del 50% lo que indica un cierto grado de aleatoriedad en su predicción con la mayoría de las detecciones, pues a partir del 50% se considera como positivo un suceso; LR produce máximos por debajo del 50% y DT toma un comportamiento binario en sus histogramas, algo que cuadra con el funcionamiento interno del algoritmo, solo que son resultados o muy malos o demasiado buenos por lo que realmente no está aportando mucha información.

Por otro lado, es llamativo cómo los modelos de árboles responden con tweets más largos en la predicción. Esto puede ser provocado por el nivel de detalle al dividir en cada nivel de profundidad del árbol y tratar de especificar más la búsqueda, pues se está clasificando un texto y a medida que se recorren los nodos de un árbol se van buscando más coincidencias dividiendo cada vez más el problema y en este caso el texto, por lo que a cierto nivel de profundidad para hacer ese análisis más largo y exhaustivo se necesitará de un campo de texto medianamente extenso para poder dividirlo y analizarlo.
NB y LR simplifican mucho más el resultado y priorizan tweets muy cortos, en ocasiones de una sola palabra (teniendo en cuenta que se está leyendo el texto preprocesado y pudiera

incluir algún emoticono, un tweet citado o referencia a otra cuenta, por ejemplo, pero eso no tiene cabida en el estudio).

Centrando la mirada en el modelo de Random Forest, parece que logra los resultados más lógicos y realistas pues pese a tener una exactitud de aproximadamente el 90% con cualquiera de los dos corpus en el entrenamiento del modelo, es en esta validación donde no consigue superar el 60% de probabilidad de acierto. Sin embargo, la mayoría de los tweets detectados son sobre hechos ocurridos que implican un suceso negativo y se han escrito de forma parecida al artículo de una noticia, es probable que sea de un artículo online o bien de alguna persona comentando lo sucedido al igual que otros tweets de opinión acerca de un suceso, pero es un indicativo de que el modelo de RF sí logra detectar sucesos y se ajusta mucho a la estructura predefinida para hacerlo a pesar de tener picos de detección con probabilidades por debajo del 50% en ambos histogramas y una proporción de detección frente a no detección del 14% y 4% con el corpus 1 y corpus 2, respectivamente.

Sin duda, los resultados dependen en gran medida del momento de ejecución, pues dependiendo de la hora a la que se ejecute, el número de tweets, el filtro para reducir búsqueda de tweets, entre otros factores, provocan distintos resultados, pero aun así deben guardar una relación. El factor más clave será por tanto ejecutar la prueba justo en el momento que se narre un suceso y alguien hable de ello prácticamente al instante.

Por último, cabe añadir que es evidente que los resultados están fuertemente condicionados por la forma en la que fue concebido el set de datos para la construcción de los modelos. Dada la problemática de no encontrar un conjunto de datos ya etiquetado se optó por automatizar un etiquetado determinista y simplista (tal y como queda explicado en el punto 3.1.1 Dataset de la memoria), cosa que influye directamente en los resultados obtenidos por cada modelo, donde los modelos más simples obtienen peores resultados y es probable que con un set de datos distinto o más completo y mejor, su comportamiento pueda variar enormemente.

5. Conclusiones y trabajos futuros

5.1 Lecciones aprendidas

El objetivo de este proyecto era diseñar un prototipo a modo de prueba de concepto para estudiar la viabilidad de la idea. Habiendo analizado y validado los resultados, se llega a la conclusión de que la idea puede llegar a ser viable si se desarrolla con más complejidad.

Si bien se ha visto que claramente de entre todos los modelos diseñados, solo 1 (RF) ha conseguido estar más cerca de los objetivos y ser más realista con las predicciones, no se podrían descartar del todo los diseños si se tiene en cuenta un nivel de desarrollo más detallado, con la construcción desde cero de cada modelo y personalizar cada parámetro y cada configuración para el caso de estudio.

Por otro lado, se han visto las carencias de no enfocar el diseño plenamente con un procesamiento del lenguaje natural. Es un campo que necesita de mucho aprendizaje y conocimiento del lenguaje humano a un gran nivel.
Si bien se han empleado técnicas propias de PLN como eliminación de *stop words* o *stemming* e incluso un filtrado para el etiquetado con un corpus con palabras negativas que implicaría un análisis de sentimientos implícitamente, no se han tenido en cuenta conductas propias de los seres humanos a la hora de expresarse. Por ejemplo, si una frase tiene intencionalidad de informar, es irónica o directamente es falsa, algo que se encuentra en abundancia en redes sociales y es difícil de detectar. Sobre todo, al vectorizar cada uno de los textos, pues el método TF-IDF no tiene en cuenta el orden relativo en el que aparece cada palabra.

El dataset utilizado ha sido etiquetado de manera automatizada, de un modo determinista dada la imposibilidad de encontrar un mejor conjunto de datos y debidamente etiquetado a mano. Esto hace que los modelos construidos estén muy condicionados por este factor y se extraigan resultados simples, sobre todo los resultados de los modelos más sencillos.

Además, el contar únicamente como datos de entrenamiento titulares de noticias es un hecho que también acota demasiado los resultados obtenidos. Si se quisiera partir con otro conjunto de datos más variado y no solo conteniendo artículos de noticias, sino que además se incluyeran textos de distintos tipos de publicaciones (noticias, comunicados falsos, bromas…) y procesarlos en varias capas del algoritmo, sería interesante estudiar las consecuencias. El problema de ese desarrollo es que puede ser similar al de este estudio que nos ocupa y caer en resultados simplistas, deterministas o con cierta aleatoriedad.

Es por todo lo anterior expuesto que se concluye que la viabilidad de la detección de suceso a través de redes sociales es posible, pero requiere de mucha especialización del caso o

casos concretos a detectar y un gran nivel de conocimiento en PLN e IA para poder personalizar cada parte del prototipo que se requiera diseñar.

5.2 Objetivos obtenidos

En relación con los objetivos planteados al inicio de esta memoria, se han logrado los siguientes puntos:

- Analizar la situación actual del problema planteado, donde se ha visto que el análisis de redes sociales es algo muy estudiado pero que sin embargo no está enfocado a predecir eventos de emergencia que puedan ayudar a poner a salvo a una persona.
- Se ha aprendido a construir varios modelos de ML de distintas formas con ayuda de librerías y paquetes de Python.
- Uso de herramientas ampliamente usadas a nivel profesional como puede ser el entorno de Jupyter Notebook para la programación del prototipo o el portal de desarrollador de Twitter para la extracción de tweets y gestión de aplicaciones desarrolladas.
- Extracción de un gran número de publicaciones estableciendo y configurando la conexión a través de la API de Twitter de las cuales se han estudiado sus campos y se han dejado los útiles para el análisis.
- Visualización de los datos y resultados concluyentes mediante gráficas reveladoras desde la propia herramienta de análisis, Jupyter Notebook.

5.3 Seguimiento de la planificación y metodología

Se ha cumplido con la planificación y metodología descrita en el primer apartado de esta memoria, ya que los hitos principales y entregas siempre se han realizado a tiempo, aunque la ejecución de alguna subtarea haya requerido más dedicación que otra dependiendo de la complejidad y falta de conocimientos.

Además, se ha podido compaginar la planificación del TFM con el curso de otra asignatura del plan de estudios y trabajo profesional del estudiante. Lo que es otro indicativo de que el planteamiento de organización ha sido el correcto.

5.4 Impactos previstos e imprevistos

Uno de los mayores problemas imprevistos que ha tenido este proyecto es la localización de un set de datos debidamente etiquetado y preferiblemente en español, ya que hay pocos estudios de publicaciones de redes sociales para el análisis de sentimientos o detección de eventos en este idioma y habría aportado más valor al campo de análisis.

Además, se contaba con la posibilidad de ubicar los eventos más destacados en los resultados que se obtuvieran, pero se ha podido observar que la mayoría de los usuarios no tienen habilitada la publicación de la ubicación a la hora de enviar un mensaje, lo cual es entendible por asuntos de privacidad, pero dificulta la localización de una emergencia detectada.

Por otro lado, se prevenía obtener unos resultados algo dispares o con cierta aleatoriedad si no se profundizaba en el procesamiento del lenguaje natural puro y como se ha podido apreciar en los resultados del apartado 4 de la memoria, la mayoría de ellos han sido muy poco relevantes o adecuados para predecir ningún tipo de suceso.

Por último, los impactos descritos en la sección 1.3 Impacto en sostenibilidad, ético-social y de diversidad de la memoria se han logrado dentro del marco de prototipo o prueba de concepto. En un estudio más completo podrán llegar a cumplirse con mayor grado de satisfacción.

5.5 Líneas de trabajo futuro

Para poder seguir estudiando la viabilidad de la idea se debería poder contar un equipo especializado principalmente en los siguientes campos:

- Científicos de datos para elaboración de algoritmos de IA totalmente personalizados desde cero y no depender de librerías limitadas de ningún lenguaje de programación.
- Expertos en PLN.
- Analistas del lenguaje humano para analizar las principales casuísticas en una publicación de cualquier red social.

Es un requisito exigente, pero se ha podido comprobar que no especializar el desarrollo conduce a análisis algo triviales y con carencia de significado verdadero en la mayoría de los casos.

Además, se debería plantear desarrollar en paralelo una infraestructura que pudiera dar soporte al análisis de grandes volúmenes de datos, una arquitectura propia de un entorno dedicado al Big Data puesto que una aplicación desarrollada y conectada con entornos inteligentes debe tener una gran capacidad de gestionar numerosos ficheros y sus metadatos.
En continuidad a este punto, se debería integrar a la plataforma una herramienta capaz del procesamiento de información en *streaming* o al menos en micro *batches* o lotes de información para poder analizar en tiempo real las publicaciones de entrada.

Otro punto a destacar, ya mencionado en apartados anteriores, es la importancia de contar con un conjunto de datos debidamente etiquetado, a ser posible realizado manualmente por un experto para no caer en resultados tan deterministas o simples.

Si bien todo lo descrito en este apartado requiere de un gran conocimiento en varios campos, el trabajo inmediato que se podría llevar a cabo para dar continuidad a la prueba de concepto y otorgar nuevas vías de exploración sería la investigación de la utilidad de aplicar algoritmos de redes neuronales a un conjunto de datos más acorde al caso de estudio.

6. Glosario

En este apartado se incluye la definición de los términos y acrónimos más relevantes utilizados en la memoria, es decir, aquellos que se han usado en más de una ocasión o constantemente. Se describen por orden alfabético en la tabla siguiente:

Término/Acrónimo	Definición
API	*Application Programming Interface* (Interfaz de Programación de Aplicaciones)
AUC	*Area Under Curve* (Área Bajo la Curva)
Big Data	Se atribuye este término al manejo de grandes volúmenes de datos en el ámbito informático. El uso de esta palabra suele incluir implícitamente la mención de arquitecturas o herramientas que dan soporte al procesamiento y almacenamiento de dicha cantidad de información.
Corpus	Agrupación de textos o palabras comúnmente usados para el análisis dentro del ámbito científico.
DataFrame	Estructura de datos en formato tabla ampliamente utilizado dentro del lenguaje de programación Python.
Dataset	Conjunto de datos.
DL	*Deep Learning* (Aprendizaje Profundo)
DT	*Decision Tree* (Árbol de Decisión)
IA	Inteligencia Artificial
JSON	*JavaScript Object Notation* (Notación de Objeto de JavaScript). Formato de texto ampliamente utilizado para el intercambio y análisis de datos.
LR	*Logistic Regression* (Regresión Logística)
ML	*Machine Learning* (Aprendizaje Automático)
NB	Naive Bayes
ODS	Objetivos y metas de Desarrollo Sostenible dentro del marco de la Agenda 2030.
Open Source	Código Abierto. Se refiere al tipo de licencia de una tecnología, en este caso, al ser de código abierto, significa que es de acceso gratuito.
Overfitting	Sobreajuste de las muestras de datos. Este suceso da lugar cuando el modelo entrenado detecta demasiados casos con probabilidad alta.
PLN	Procesamiento del Lenguaje Natural
RF	*Random Forest* (Bosque Aleatorio)
ROC	*Receiver Operating Characteristic* (Característica Operativa del Receptor)
RRSS	Redes Sociales
Script	Se trata de un documento que recoge una secuencia ordenada de comandos u algoritmos para la automatización de procesos en un lenguaje de programación determinado.

TF-IDF	*Term Frequency-Inverse Document Frequencies* (Frecuencia de Término - Frecuencias Inversas de Documento). Técnica utilizada en el PLN para dar pesos y vectorizar un término o varios dentro de una cadena de texto.
Tweet	Nombre que recibe una publicación por parte de un usuario dentro de la plataforma de Twitter.
Underfitting	Infrajuste de las muestras de datos. Este suceso da lugar cuando el modelo entrenado no es capaz de detectar un número mínimo de casos con una probabilidad de detección fiable.

Tabla 25. Listado de términos y acrónimos

7. Bibliografía

[1] DSAYCE. Última fecha de consulta: 3 de octubre de 2022. https://www.dsayce.com/social-media/tweets-day/

[2] Radar COVID. Última fecha de consulta: 21 de octubre de 2022. https://radarcovid.gob.es/home

[3] techopedia. Última fecha de consulta: 21 de octubre de 2022. https://www.techopedia.com/definition/6405/cell-broadcast-cb

[4] BOE. Última fecha de consulta: 21 de octubre de 2022. https://www.boe.es/doue/2018/321/L00036-00214.pdf

[5] Dirección General de Protección Civil y Emergencias. Última fecha de consulta: 21 de octubre de 2022. https://www.proteccioncivil.es/coordinacion/redes/ran/public-warning-system#:~:text=El%20protocolo%20adaptado%20para%20Espa%C3%B1a,proceso%20de%20despliegue%20en%20Espa%C3%B1a.

[6] AMBER Alert. Última fecha de consulta: 21 de octubre de 2022. https://amberalert.ojp.gov/about/faqs#faq1

[7] American Red Cross. Última fecha de consulta: 21 de octubre de 2022. https://www.redcross.org/get-help/how-to-prepare-for-emergencies/mobile-apps.html

[8] Twitter Developer Platform. Última fecha de consulta: 21 de octubre de 2022. https://developer.twitter.com/en/docs/platform-overview

[9] RapidMiner Documentation. Última fecha de consulta: 21 de octubre de 2022. https://docs.rapidminer.com/latest/studio/connect/cloud/twitter.html

[10] NiFi Documentation. Última fecha de consulta: 21 de octubre de 2022. https://nifi.apache.org/docs/nifi-docs/components/org.apache.nifi/nifi-social-media-nar/1.5.0/org.apache.nifi.processors.twitter.ConsumeTwitter/index.html

[11] Nilsson, Nils John (2000). *Inteligencia Artificial: Una Nueva Síntesis*. McGraw-Hill.

[12] Seldon Documentation. Última fecha de consulta: 21 de octubre de 2022. https://www.seldon.io/supervised-vs-unsupervised-learning-explained#:~:text=The%20main%20difference%20between%20supervised,processes%20unlabelled%20or%20raw%20data.

[13] IBM. Última fecha de consulta: 21 de octubre de 2022. https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning

[14] Brownlee, Jason. *6 Dimensionality Reduction Algorithms With Python*. Machine Learning Mastery. 10 de julio de 2020. Última fecha de consulta: 21 de octubre de 2022. https://machinelearningmastery.com/dimensionality-reduction-algorithms-with-python/

[15] Santiago Halibas, Alrence et al. *Application of Text Classification and Clustering of Twitter Data for Business Analytics*. IEEE. 24 de mayo de 2018. Última fecha de consulta: 21 de octubre de 2022. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8363162

[16] Arista Ulfa, Maria et al. *Twitter Sentiment Analysis using Naïve Bayes Classifier with Mutual Information Feature Selection*. J-COSINE, Vol. 2, No. 2, diciembre de 2018. Última fecha de consulta: 21 de octubre de 2022. https://jcosine.if.unram.ac.id/index.php/jcosine/article/download/120/26/

[17] Ahmad Fathan Hidayatullah et al. 2021 IOP Conf. Ser.: Mater. Sci. Eng. 1077 012001. *Sentiment Analysis on Twitter using Neural Network: Indonesian Presidential Election 2019 Dataset*. Última fecha de consulta: 21 de octubre de 2022. https://iopscience.iop.org/article/10.1088/1757-899X/1077/1/012001/pdf

[18] P. Rodrigues, Anisha. *Real-Time Twitter Trend Analysis Using Big Data Analytics and Machine Learning Techniques*. Hindawi, Wireless Communications and Mobile Computing. Volume 2021, Article ID 3920325. Última fecha de consulta: 21 de octubre de 2022. https://downloads.hindawi.com/journals/wcmc/2021/3920325.pdf

[19] Krish, Shruthi. *Finding Local Events Using Twitter Data*. Towards Data Science. 16 de diciembre de 2018. Última fecha de consulta: 21 de octubre de 2022. https://towardsdatascience.com/finding-local-events-using-twitter-data-18298f03ead6

[20] Alabau, Irene. *La rueda de las emociones de Robert Plutchik*. Psicología-Online. Actualizado: 8 de noviembre de 2021. Última fecha de consulta: 21 de octubre de 2022. https://www.psicologia-online.com/la-rueda-de-las-emociones-de-robert-plutchik-4707.html#anchor_0

[21] MeaningCloud Documentation. Última fecha de consulta: 21 de octubre de 2022. https://www.meaningcloud.com/blog/accuracy-in-sentiment-analysis

[22] Dr. Vázquez-Cano, Esteban et al. *La repercusión del movimiento MOOC en las redes sociales. Un estudio computacional y estadístico en Twitter*. Revista Española de Pedagogía, año LXXV, nº 266, enero-abril 2017, 47-64. Última fecha de consulta: 21 de octubre de 2022. https://reunir.unir.net/bitstream/handle/123456789/6198/La-repercusion-del-movimiento-MOOC.pdf?sequence=1&isAllowed=y

[23] Dra. Lozano-Blasco, Raquel. *Redes sociales y su influencia en los jóvenes y niños: Análisis en Instagram, Twitter y YouTube*. Comunicar. Revista Científica de Comunicación y Educación, octubre de 2022, DOI: 10.3916/C74-2023-10. Última fecha de consulta: 21 de octubre de 2022. https://www.researchgate.net/profile/Marta-Aladren/publication/363925117_Redes_sociales_y_su_influencia_en_los_jovenes_y_ninos_Analisis_en_Instagram_Twitter_y_YouTube/links/63356177ff870c55cee7e05c/Redes-sociales-y-su-influencia-en-los-jovenes-y-ninos-Analisis-en-Instagram-Twitter-y-YouTube.pdf

[24] Jupyter Documentation. Última fecha de consulta: 21 de octubre de 2022. https://jupyter.org/about

[25] Zeppelin Documentation. Última fecha de consulta: 21 de octubre de 2022. https://zeppelin.apache.org/docs/0.10.1/

[26] Google Colab Documentation. *FAQ*. Última fecha de consulta: 21 de octubre de 2022. https://research.google.com/colaboratory/intl/es/faq.html

[27] Zeppelin Documentation. *Interpreter in Apache Zeppelin*. Última fecha de consulta: 21 de octubre de 2022. https://zeppelin.apache.org/docs/0.10.1/usage/interpreter/overview.html

[28] BOE. Última fecha de consulta: 21 de octubre de 2022. https://www.boe.es/buscar/act.php?id=BOE-A-2021-8806

[29] Comisión Europea. *La protección de datos en la UE*. Última fecha de consulta: 21 de octubre de 2022. https://ec.europa.eu/info/law/law-topic/data-protection/data-protection-eu_es#:~:text=El%20Reglamento%20general%20de%20protecci%C3%B3n%20de%20datos%20(RGPD)&text=El%20Reglamento%20es%20una%20medida,en%20el%20mercado%20%C3%BAnico%20digital.

[30] Rohit Kulkarni. *A Million News Headlines*. Kaggle. Extraídos originalmente del medio Australian Broadcasting Corporation. Última fecha de consulta: 10 de diciembre de 2022. https://www.kaggle.com/datasets/therohk/million-headlines

[31] Bing Liu. *Opinion Mining, Sentiment Analysis, and Opinion Spam Detection*. Última fecha de consulta: 10 de diciembre de 2022. https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

[32] Minqing Hu and Bing Liu. "*Mining and Summarizing Customer Reviews*". Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Agosto 22-25, 2004, Seattle, Washington, Estados Unidos.

[33] Natural Language Toolkit. *Corpus package*. Última fecha de consulta: 10 de diciembre de 2022. https://www.nltk.org/search.html?q=stopwords&check_keywords=yes&area=default

[34] Natural Language Toolkit. *SnowballStemmer module*. Última fecha de consulta: 10 de diciembre de 2022. https://www.nltk.org/api/nltk.stem.snowball.html

[35] Urdaneta Fernández, Lino Alberto. *Reducir el número de palabras de un texto: lematización y radicalización (stemming) con Python*. 4 de mayo de 2019. Última fecha de consulta: 10 de diciembre de 2022. https://medium.com/qu4nt/reducir-el-n%C3%BAmero-de-palabras-de-un-texto-lematizaci%C3%B3n-y-radicalizaci%C3%B3n-stemming-con-python-965bfd0c69fa

[36] Mamata Das, Selvakumar Kamalanathan and P.J.A. Alphonse. *A Comparative Study on TF-IDF Feature Weighting Method and its Analysis using Unstructured Dataset*. 5th International Conference on Computational Linguistics and Intelligent Systems, April 22–23, 2021, Kharkiv, Ukraine. Última fecha de consulta: 10 de diciembre de 2022. https://ceur-ws.org/Vol-2870/paper10.pdf

[37] Soner Yıldırım. *Naive Bayes Classifier — Explained*. 14 de febrero de 2022. Última fecha de consulta: 10 de diciembre de 2022. https://towardsdatascience.com/naive-bayes-classifier-explained-50f9723571ed

[38] Khandelwal, Renu. *Quick and Easy Explanation of Logistic Regression*. 7 de abril de 2020. Última fecha de consulta: 10 de diciembre de 2022. https://towardsdatascience.com/quick-and-easy-explanation-of-logistics-regression-709df5cc3f1e

[39] Brownlee, Jason. *How to Identify Overfitting Machine Learning Models in Scikit-Learn*. Machine Learning Mastery. 27 de noviembre de 2020. Última fecha de consulta: 10 de diciembre de 2022. https://machinelearningmastery.com/overfitting-machine-learning-models/

[40] Scikit learn. *RandomForestClassifier*. Última fecha de consulta: 10 de diciembre de 2022. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[41] Scikit learn. *GridSearchCV*. Última fecha de consulta: 10 de diciembre de 2022. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

8. Anexos

Ficheros adjuntos con la memoria:

- Fichero "***extract-data-from-twitter.ipynb***". Script con el código Python en formato Notebook donde se extraen tweets desde Twitter.
- Fichero "***extract-data-from-twitter-output.html***". Salida exportada en formato HTML del Notebook anterior: "***extract-data-from-twitter.ipynb***".
- Fichero "***get-train-data.ipynb***". Script con el código Python en formato Notebook donde se conforman los dos corpus para el entrenamiento de los modelos a partir de un CSV de entrada.
- Fichero "***project-tfm-analysis.ipynb***". Script con el código Python en formato Notebook donde se entrenan y validan los modelos en función del corpus seleccionado.
- Fichero "***project-tfm-analysis-corpus1-output.html***". Salida exportada en formato HTML del Notebook "***project-tfm-analysis.ipynb***" usando el corpus 1.
- Fichero "***project-tfm-analysis-corpus2-output.html***". Salida exportada en formato HTML del Notebook "***project-tfm-analysis.ipynb***" usando el corpus 2.

Se opta por adjuntar en un ZIP todos los ficheros anteriores junto a la memoria dado su formato de origen y la dificultad de añadirlos de una forma clara y legible dentro de un documento Word.