UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

MASTER UNIVERSITARIO EN ANALISIS DE DATOS MASIVOS (BIG DATA)

TRABAJO FIN DE MÁSTER

ANÁLISIS DE SENTIMIENTOS MEDIANTE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

FABRICIO TORRES SAMUEZA

Dirigido por

MARIA CRUZ GAYA LOPEZ

CURSO 2024-2025

TÍTULO: ANÁLISIS DE SENTIMIENTOS MEDIANTE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

AUTOR: FABRICIO TORRES SAMUEZA

TITULACIÓN: MASTER UNIVERSITARIO EN ANÁLISIS DE DATOS MASIVOS (BIG DATA)

DIRECTOR/ES DEL PROYECTO: MARIA CRUZ GAYA LOPEZ

FECHA: SEPTIEMBRE de 2024

RESUMEN

El presente trabajo tiene por objetivo realizar una herramienta capaz de clasificar automáticamente los comentarios extraídos de Twitter [1] en categorías de positivos, negativos o neutrales. Esta herramienta se basa en técnicas de procesamiento de lenguaje natural y aprendizaje automático, permitiendo analizar grandes volúmenes de datos en tiempo real.

Los resultados del proyecto ofrecerán a las empresas una comprensión más profunda de las opiniones de los usuarios, permitiéndoles detectar no solo las emociones predominantes, sino también identificar tendencias y patrones. Está información será útil para la toma de decisiones estratégicas, ya que permitirá a las empresas responder de manera más efectiva a las necesidades y preocupaciones de sus clientes. Además, la herramienta facilitará la identificación temprana de posibles crisis de reputación, proporcionando a las empresas la oportunidad de actuar de manera preventiva

Palabras clave: Emociones, Análisis de Sentimientos, Machine Learning [26], Twitter [1], Gestión de Reputación Corporativa.

ABSTRACT

The purpose of this project is to develop a tool capable of automatically classifying comments extracted from Twitter [1] into positive, negative, or neutral categories. This tool is based on natural language processing techniques and Machine Learning [26], allowing for the real-time analysis of large volumes of data.

The results of the project will offer companies a deeper understanding of user opinions, enabling them to detect not only predominant emotions but also identify trends and patterns. This information will be valuable for strategic decision-making, as it will allow companies to respond more effectively to the needs and concerns of their customers. Additionally, the tool will facilitate the early identification of potential reputation crises, providing companies with the opportunity to act preventively.

Keywords: Emotions, Sentiment Analysis, Machine Learning [26], Twitter [1], Corporate Reputation Management.

AGRADECIMIENTOS

La primera persona a la que quiero agradecer es a mi tutora, María Cruz Gaya López, por guiarme y apoyarme a lo largo del desarrollo de este proyecto y también por sus consejos y ánimos, estando siempre disponible para cuando la he necesitado.

También quiero darle las gracias a mi familia ya que de no ser por ellos no habría podido embarcarme en la aventura que es este máster y también por todo lo que me han dado a lo largo de tantos años.

Por último, pero no por ello menos importante, muchas gracias a mis amigos y compañeros que han estado ayudándome a lo largo del máster, así como a todas aquellas personas que en algún momento han tenido que aguantarme.

TABLA RESUMEN

	DATOS
Nombre y apellidos:	Fabricio Torres Samueza
Título del proyecto:	Análisis de sentimientos mediante técnicas de aprendizaje automático
Directores del proyecto:	María Cruz Gaya López
El proyecto se ha realizado en colaboración de una empresa o a petición de una empresa:	NO
El proyecto ha implementado un producto: (esta entrada se puede marcar junto a la siguiente)	SI
El proyecto ha consistido en el desarrollo de una investigación o innovación: (esta entrada se puede marcar junto a la anterior)	NO
Objetivo general del proyecto:	Crear una herramienta capaz de clasificar comentarios para así ayudar a las empresas

Índice

Índice de Figuras

Índice de Tablas

Capítulo 1. RESUMEN DEL PROYECTO

1.1 Contexto y justificación

Hoy en día es muy importante tener en cuenta la opinión de los usuarios y las emociones expresadas en línea ya que juegan un papel muy importante en la formación de tendencias sociales, políticas y en la reputación de marca, influyendo así en la toma de decisiones de empresas, gobiernos, etc.

Por lo que el análisis de sentimientos es una herramienta imprescindible para controlar de cerca la percepción que tienen los consumidores sobre las empresas, ya que puede tener un impacto considerable sobre ellas.

Dada la importancia de analizar los sentimientos, opiniones de los usuarios de manera eficiente y precisa en medios sociales en el siglo XXI, es esencial que las empresas comprendan y utilicen herramientas de análisis de sentimientos para, medir la reputación de marca, mejorar la oferta de productos/servicios, conocer qué opinión tienen acerca de la empresa, optimizar la gestión de crisis y ser más competitivos.

1.2 Planteamiento del problema

Debido al volumen masivo de datos generados diariamente en plataformas sociales y el desafío que presentan estos para las empresas que desean entender la opinión de las personas sobre temas específicos. Este proyecto se centra en desarrollar una herramienta capaz de extraer y analizar eficazmente estas opiniones.

1.3 Objetivos del proyecto

El objetivo general del proyecto es desarrollar una herramienta para evaluar las opiniones expresadas en las redes sociales, específicamente en Twitter [1] y como objetivos específicos, extraer y procesar tweets, implementar un algoritmo de análisis de sentimiento y evaluar la precisión del modelo con métricas estándar.

1.4 Resultados obtenidos

El análisis reveló que el modelo de Regresión Logística [10] con CountVectorizer [11] obtuvo una precisión del 71.9% en la clasificación de sentimientos, logrando convertir datos cuantitativos, en un output cualitativo que indica un tweet es positivo, negativo o neutro.

1.5 Estructura de la memoria

La memoria del proyecto se estructura de la siguiente forma:

En el capítulo 1 se describe el contexto del trabajo, sus objetivos, los resultados obtenidos y la estructura del documento

En el capítulo 2 se realiza una revisión de algunas herramientas con funcionalidades similares al trabajo realizado en el proyecto, se describe la motivación para el trabajo y se realiza el planteamiento del problema.

En el capítulo 3 se plantean los objetivos generales y específicos del proyecto, así como sus beneficios.

En el capítulo 4 se describe la planificación de trabajo, la solución propuesta para el análisis de sentimientos, las herramientas, recursos empleados, se evalúa la viabilidad del proyecto considerando aspectos técnicos/económicos y se concluye con los resultados finales obtenidos

En el capítulo 5 se analizan e interpretan los resultados obtenidos, comparándolos con las expectativas de estudios previos.

En el capítulo 6 trata de las conclusiones, los objetivos que se han cumplido, los desafíos enfrentados y lo aprendido durante el proceso.

En el capítulo 7 se describe el trabajo a futuro, mejoras que se podrían realizar basándose en las limitaciones y en los resultados obtenidos.

En el capítulo 8 se incluyen todas las fuentes bibliográficas citadas a lo largo del trabajo.

En el capítulo 9 se adjunta el material adicional relevante que complementan el proyecto.

Capítulo 2. ANTECEDENTES / ESTADO DEL ARTE

2.1 Estado del arte

En el presente apartado se recogerán las características más importantes acerca del software semejante que hay actualmente en el mercado.

La herramienta implementada es una herramienta de análisis de sentimiento diseñada para explorar cómo las opiniones y emociones expresadas en línea pueden impactar en la reputación y el desempeño de una empresa. En este ámbito se han encontrado las siguientes herramientas:

2.1.1 Service Hub de HubSpot

Service Hub de Hubspor [2] es una herramienta que permite extraer el feedback de los clientes mediante la evaluación de sus respuestas a encuestas. Esta herramienta permite identificar los comentarios positivos de los negativos a través de tecnología NPS, que indica si una reseña es buena o mala y las organiza dependiendo de su sentimiento. Además, se puede extraer información sobre la satisfacción del cliente a partir de las tablas y gráficas que el sistema genera. Entre sus características destaca:

- Ofrece comentarios sobre la encuesta
- Interacción con el cliente en tiempo real
- La integración de CRM

2.1.2 Social Mention

Social Mention [3] es una herramienta orientada a rastrear menciones de marcas, productos, competidores o palabras clave en redes sociales y otros sitios web. Trabaja con el “sentimiento” de las menciones (es decir, si el comentario es positivo o negativo) y la influencia de quienes lo han publicado. Entre sus características destaca:

- Información valiosa sobre las palabras clave más vinculadas
- Números de datos de sentimiento categorizados

2.1.3 Brand24

Brand24 [4] es una herramienta orientada al monitoreo e identificación de eventos relacionados con las redes sociales. Emitiendo notificaciones instantáneas en caso de menciones, comentarios, sentimientos e interacciones en general con los usuarios, seguidores o embajadores de marca. Permite clasificar las interacciones de los usuarios y asignar

automáticamente un nivel de actitud (negativo, neutro o positivo). Brindando indicadores para comprender su respuesta ante diferentes propuestas de contenido, campañas o productos en general. Entre sus características destaca:

- Su interfaz gráfica y categórica
- Recopila datos de la mayoría de las fuentes en línea

2.1.4 Repustate

Repustate [5] es una herramienta de análisis de sentimiento de contenido premium, permite obtener comentarios de los clientes dividiendo los comentarios en categorías gramaticales. Al hacer esto, la herramienta analiza y categoriza las interacciones en positivas y negativas. Una vez obtenido los resultados estarán disponibles para su visualización en forma de gráficos, gráficos circulares y gráficos de barras. Entre sus características destaca:

- Ofrece una prueba gratuita, un plan estándar y un plan personalizado
- Ofrece varias opciones de presentación de datos gráficos
- Analiza con precisión los comentarios positivos y negativos

2.1.5 Lexalytics

Lexalyttics [6] es una herramienta de análisis de texto enfocada en explicar por qué un consumidor responde a una empresa de cierto modo. Emplea algoritmos de procesamiento de lenguaje natural, que analizan el texto y arrojan un análisis de sentimiento que deja ver la intención detrás de cada mensaje. Entre sus características destaca:

- Herramienta de detección de idioma
- Seguimiento de palabras clave
- Datos de gráfico de barras

	Service Hub	Social Mention	Brand24	Repustate	Lexalytics	MiApp
Informes y análisis	✓		✓			
Seguimiento en tiempo real		✓	✓			✓
Análisis de sentimiento		✓	✓	✓	✓	✓

Monitoreo de menciones		✓	✓			✓
Análisis multilingüe				✓	✓	
Resúmenes automáticos					✓	✓
Procesamiento de lenguaje natural (NLP)		✓	✓	✓	✓	✓
Personalización	✓		✓	✓	✓	✓
Interfaz amigable	✓	✓	✓			✓
Costoso	✓		✓	✓	✓	

Tabla 2-1. Comparación de Herramientas existentes para análisis de sentimientos

2.2 Contexto y justificación

El análisis de sentimiento es una herramienta que nos permite comprender la percepción y la opinión de las personas hacia diferentes temas, productos o entidades, puede ser fundamental para evaluar la reputación de una empresa, comprender la satisfacción del cliente, identificar áreas de mejora y anticipar tendencias del mercado.

Para realizar este trabajo se va a crear una herramienta de análisis de sentimiento diseñada para explorar cómo las opiniones y emociones expresadas en línea pueden impactar en la reputación y el desempeño de una empresa. Esta herramienta permitirá recopilar, analizar, visualizar datos de manera eficiente y así poder utilizar esta información para generar insights y tomar decisiones estratégicas.

2.3 Planteamiento del problema

El análisis de sentimiento en redes sociales se ha convertido en una herramienta importante para las empresas que buscan evaluar la percepción pública sobre sus productos y servicios. El estado del arte muestra la existencia de gran cantidad de herramientas para el análisis de sentimientos, pero muchas de ellas tienen ciertas limitaciones significativas, como por ejemplo la falta de precisión y accesibilidad restringida debido a su alto costo.

Aunque existen soluciones disponibles, muchas técnicas actuales no logran capturar de manera efectiva los sentimientos. Por tanto, es necesario mejorar las herramientas disponibles para así lograr una mayor precisión en la clasificación de sentimientos y opiniones expresadas en las redes sociales.

Capítulo 3. OBJETIVOS

3.1 Objetivos generales

El objetivo principal de este proyecto es desarrollar una herramienta que permita ver las diversas opiniones clasificadas en positivas, negativas y neutras que tienen los usuarios sobre las empresas y así poder tomar decisiones en base ellas.

3.2 Objetivos específicos

El objetivo general se puede refinar en los siguientes objetivos más específicos:

- **Recopilación de Datos**: Buscar conjunto de datos en plataformas web, como por ejemplo Kaggle [7], la cual permite a los usuarios encontrar y publicar conjunto de datos, explorar y crear modelo en entorno de ciencia de datos. Se tiene que garantizar que los datos recopilados sean representativos y equilibrados (opiniones positivas, negativas y neutras) para así tener datos para poder realizar el entrenamiento del modelo.

- **Limpiar y preprocesar de Datos**: Eliminando duplicados, palabras vacías (stop words), comentarios irrelevantes, tokenizar los comentarios, convertirlos a minúsculas y realizar lematización para obtener una correcta normalización del texto que se va a utilizar.

- **Realizar un análisis exploratorio**: Para comprender la distribución de las opiniones.

- **Seleccionar un algoritmo de clasificación adecuado**: Dividir los datos en conjuntos de entrenamiento y prueba.
  Entrenar el modelo, ajustar los hiperparámetros del modelo, con el fin de construir un modelo de clasificación que funcione correctamente tanto con los datos de entrenamiento como con nuevos datos, para proporcionar predicciones precisas y fiables.

- **Evaluar el rendimiento del modelo:** Utilizando métricas como precisión, F1-score y matriz de confusión y realizar pruebas en el conjunto de prueba para asegurar la generalización del modelo para asegurar que el modelo de ML sea preciso, robusto y se pueda aplicar en situaciones reales.

- **Implementar la funcionalidad**: De procesamiento de texto y clasificación de sentimientos y desarrollar una interfaz de usuario (UI) para la herramienta que

permita realizar una búsqueda de la empresa de la cual se desee realizar el análisis de sentimiento.

- **Validar la herramienta de análisis de sentimiento**: Con conjuntos de datos adicionales para asegurar que el modelo sea preciso, robusto, fiable y se pueda aplicar en diferentes contextos y escenarios del mundo real.

- **Documentar el proceso de desarrollo**: Incluir la metodología, los algoritmos utilizados, los resultados y las conclusiones para servir como base para futuras mejoras.

- **Realizar presentación clara y concisa**: Resaltando las implicaciones empresariales y las recomendaciones derivadas del análisis de sentimiento para comunicar el valor del análisis de sentimiento.

3.3 Beneficios del proyecto

El principal beneficio del proyecto es que proporciona una herramienta precisa y accesible para el análisis de sentimientos en redes sociales, más precisamente en Twitter [1]. Al tener una buena exactitud en la clasificación de emociones y opiniones, las empresas podrán tomar mejores decisiones, lo que beneficiará en una mejor gestión de su imagen de marca y refuerza sus estrategias de marketing.

Capítulo 4. DESARROLLO DEL PROYECTO

4.1 Planificación del proyecto

Actividades	Mayo	Junio				Julio				Agosto				Septiembre
	W1	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1
Recopilación de datos y estado del arte														
Análisis Exploratorio de Datos														
Entrenamiento del Modelo de Análisis de Sentimiento														
Evaluación del Modelo														
Desarrollo de la Herramienta de Análisis de Sentimiento y validación														
Documentación y Presentación														

Figura 4-1. Planificación del proyecto

4.2 Descripción de la solución, metodologías y herramientas empleadas

4.2.1 Metodología general

El desarrollo del proyecto de estructuró en las siguientes fases:

- **Obtención de datos**: Los tweets se han obtenido de varios datasets de Kaggle [7], de GitHub [8] y de TASS (Taller de Análisis Semántico en la Sociedad Española para el Procesamiento del Lenguaje Natural) [9] en la cual existen ficheros XML con tweets en español.

- **Preprocesamiento del texto**: Consiste en eliminar inconsistencias, como valores nulos, duplicados, ruido y todo aquello que no es en sí texto, es decir, enlaces, usuarios, hashtags, caracteres no alfanuméricos, palabras sin significado o stopwords, convertir los emoticonos a texto… Se trata de normalizar el texto

- **Vectores de palabras**: Las máquinas no procesan palabras directamente para realizar clasificaciones, sino que trabajan con números.
  Para que los algoritmos de clasificación puedan interpretar el lenguaje natural, es necesario convertir las palabras en vectores numéricos, que representan la información de una manera que las máquinas puedan entender y procesar.

- **Entrenamiento del modelo**: Se entrenaron dos modelos principales para el análisis de sentimientos. El primer modelo es una red neuronal recurrente bidireccional con una capa (RNN) [12]. El segundo modelo es una Regresión Logística [10], que se entrenó utilizando características generadas a partir de un CountVectorizer [11].
  El rendimiento de los modelos se comparó utilizando la métrica de exactitud (accuracy), seleccionando el modelo de Regresión Logística [10] ya que demostró ser más efectivo.

- **Obtención de Datos de Twitter**: La extracción de datos recientes de Twitter [1] se realizó mediante Web Scraping [13], para ello se utiliza la librería Beautiful Soup [14], los tweets extraídos se someten a un preprocesamiento similar al de los datos de entrenamiento.

- **Aplicación del modelo y análisis de los resultados**: El modelo entrenado se aplicó a los tweets preprocesados para clasificar el sentimiento de cada uno y así poder realizar la visualización utilizando Dash [15], permitiendo una interpretación clara de los sentimientos.

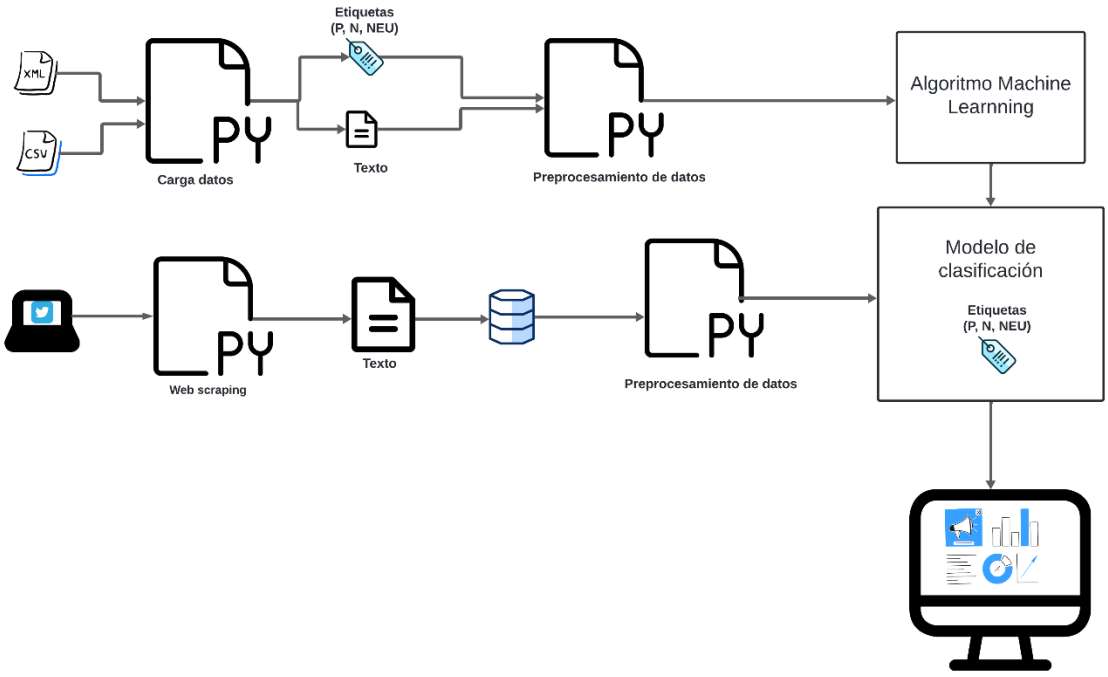


Figura 4-2. Arquitectura del proyecto

4.2.2 Recopilación y preparación de datos para el modelo de aprendizaje

4.2.2.1 Obtención de datos

El primer paso del procedimiento consiste en obtener los datos con los que se va a partir para llevar a cabo el análisis de sentimiento. En este caso se decidió que estos datos van a ser tweets de la red social Twitter [1], en la cual la gente expresa sus sentimientos y opiniones sobre cualquier tema.

El primer paso del procedimiento consiste en obtener los datos con los que se va a partir para l

Para el proyecto se decidió utilizar varios conjuntos de datos provenientes de diversas fuentes. Estas fuentes incluyen Kaggle [7], Github [8], datos.gob.es [16] y TASS [9] el cual está en formato XML.

Antes de combinar todos los conjuntos de datos en uno solo, se utilizó un código en Python [17] para procesar el fichero XML del TASS [9] y convertirlo en un Dataframe. En este proceso, se eliminaron las columnas que no son relevantes, conservando únicamente aquellas que contenían el texto y la polaridad del sentimiento. También se eliminaron las filas con valores nulos y duplicados. Una vez completado este tratamiento, se procedió a unir todos los conjuntos de datos en un único archivo CSV [18].

Finalmente, se obtuvo un fichero CSV [18] con un total de 1.087.339 tweets en español. La distribución de las polaridades fue la siguiente: aproximadamente un 71.2% de los tweets tenían polaridad neutral (774438), el 15,5% tenía polaridad positiva (168426), el 12.6% tenía polaridad negativa (136789), el 0.7% tenía polaridad mezclada (7529) y el 0.01% tenía polaridad nula (157).

Para obtener un dataset más balanceado, se decidió seleccionar únicamente los primeros 100000 tweets con polaridad neutral.

```
tweet_text,sentiment
menos campaña todavía preguntar ir lugar,NEUTRAL
tener jugar él equipo,NEUTRAL
```

Figura 4-3. Muestra del formato de tweets

En este caso, al leer del fichero los tweets para pasarlos a un dataframe, las columnas que se obtienen son:

- **tweet_text**: Es el texto del tweet
- **sentiment**: La polaridad del tweet (POSITIVE, NEGATIVE, NEUTRAL, MIXED, NONE)

Dado que la columna “sentiment” contiene cinco tipos de polaridad, se decidió eliminar todas aquellas filas que tengan polaridad nula (NONE) y mezclada (MIXED).

Una muestra de los cinco primeros tweets sería la siguiente:

	tweet_text	sentiment
0	@JoseAMeadeK @miseleccionmx A pero si estás en...	NEUTRAL
1	#MarioPereyraDT\n"Tenemos que jugarle a Franci...	NEUTRAL
2	@miseleccionmx No me pidas eso mi selección, s...	NEGATIVE
3	Si llega a ser la despedida, no será la mejor....	POSITIVE
4	No se les olvide que nuestro trabajo es constr...	NEUTRAL

Figura 4-4. Muestra de tweets

La distribución de sentimientos en el dataset es la siguiente:

Figura 4-5. Distribución de sentimientos

4.2.2.2 Preprocesamiento del texto

Una vez obtenido el conjunto de tweets seleccionado, se procede a realizar la normalización del texto, realizando para ello una limpieza de este.

En la Figura 4-4 se observa que los tweets pueden estar escritos tanto en mayúsculas como en minúsculas, contener signos de puntuación, caracteres no alfanuméricos, enlaces y codificación de texto. Asimismo, hay palabras que no aportan significado relevante para diferenciar los tweets.

Por lo que, se procede a realizar un procesamiento de los tweets, para que de esta manera asegurar que todos estén normalizados de la misma forma.

Para limpiar el texto se utilizará el módulo de Python [17] re [19]. Este módulo permite eliminar texto que coincida con un patrón o sustituirlo por otro cualquiera, facilitando así la normalización del contenido

Un ejemplo de patrones que se utilizará son los siguientes:

```
# Eliminar URLs
text = re.sub(r'https?://\S+|www\.\S+', '', text)
# Eliminar palabras con @ y #
text = re.sub(r'[@#]\w+', '', text)
```

Figura 4-6. Código de patrones de la librería re

El primero busca patrones que comiencen con “http://”, “https://” o “www”, seguidos de cualquier secuencia de caracteres, eliminando así las URLs. El segundo de ellos busca patrones que comience con “@” o “#”, eliminando las menciones y los hashtags del texto.

También se convierten los emojis a su descripción textual correspondiente utilizando la biblioteca Emoji [20].

Se eliminan los signos de puntuación, conservando únicamente caracteres alfanuméricos y algunos caracteres especiales. Además, se reemplaza el carácter de subrayado “_” por un espacio en blanco para mejorar la legibilidad del texto y se sustituye los saltos de línea “\n” por espacios en blanco para evitar fragmentaciones.

Lo siguiente que se realiza es convertir el texto a minúsculas para asegurar uniformidad y evitar distinciones entre mayúsculas y minúsculas. Luego, se procesa el texto utilizando la biblioteca Spacy [21], la cual permite la lematización y el análisis de las partes del discurso.

Finalmente se realiza la lematización del texto, eliminando palabras que no aportan significado relevante (stopwords).

Solo se conservan sustantivos, adjetivos, verbos y adverbios ya que son las palabras más relevantes y facilitan una evaluación más precisa de los sentimientos, ayudando a mejorar la calidad de los resultados y la interpretación del análisis del texto.

La siguiente función recibe los tweets y realiza estas modificaciones que se comentaron anteriormente.

```
def clean(text):
    # Eliminar URLs
    text = re.sub(r'https?://\S+|www\.\S+', '', text)
    # Eliminar palabras con @ y #
    text = re.sub(r'[@#]\w+', '', text)
    # Tratar los emojis
    text = emoji.demojize(text, language='es')
    # Eliminar puntuaciones
    text = re.sub(r'[^\w\sáéíóúüñ]', ' ', text)
    # Sustituir el '_' por un espacio en blanco al tratar los emojis
    text = text.replace('_', ' ')
    # Sustituir los '\n' por espacios en blanco
    text = text.replace('\n', '')
    # Convertir a minúscula
    text = text.lower()
    # Procesar el texto con spacy
    doc = nlp(text)
    # Lematización y filtrado de stopwords
    lemmatized_tokens = [
        token.lemma_ for token in doc
        if token.lemma_ not in stop_words and token.lemma_ != '' and token.
        lemma_ not in english_words and token.pos_ in ['NOUN', 'ADJ', 'VERB',
        'ADV']]

    # Unir tokens en un string
    cleaned_text = ' '.join(lemmatized_tokens)

    return cleaned_text
```

Figura 4-7. Código para realizar la limpieza de tweets

Una vez realizado este proceso, obtenemos lo siguiente:

	tweet_text	sentiment
0	menos campaña todavía preguntar ir lugar	NEUTRAL
1	tener jugar él equipo	NEUTRAL
4	olvidar trabajo construir dictadura proletariado	NEUTRAL
5	existir partido político sólo existir	NEUTRAL
6	festejar triunfo selección mexicano	NEUTRAL

Figura 4-8. Muestra de tweets tras realizar el preprocesamiento

Por último, se muestran cuáles son las palabras más repetidas tanto en tweets con polaridad positiva (Figura 4-9 Izquierda), con polaridad negativa (Figura 4-9 Centro) y con polaridad neutral (Figura 4-9 Derecha)

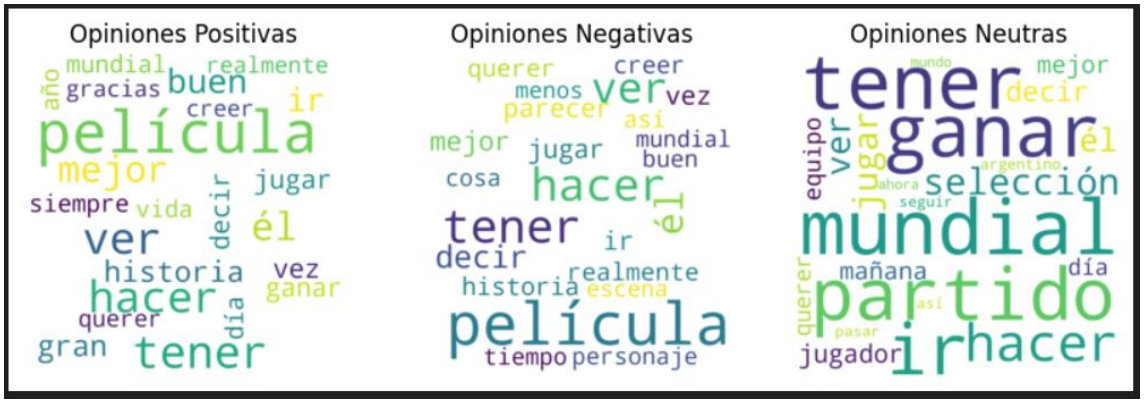


Figura 4-9. Nube de palabras de tweets con polaridad

4.2.2.3 Vectores de palabras

Word2Vec

Para realizar Word Embedding [22] con Word2Vec [23], la cual consiste en convertir palabras en representaciones numéricas (vectores) que capturan el significado semántico de las palabras, se utilizó el paquete Word2Vec [23] de la librería Gensim [24], se puede utilizar para averiguar relaciones entre las palabras de un conjunto de datos, calcular similitud entre ellas o clasificar textos. Para ello se tokeniza el corpus normalizado y luego se construye el modelo word2Vec [23].

Los parámetros principales para configurar el modelo son:

- **vector_size**: Establece el tamaño o la dimensión de los vectores de palabras y puede oscilar entre decenas y miles, determina la dimensionalidad de los embeddings. Por defecto tiene un valor de 100.

- **window**: Establece el tamaño de la ventana que especifica la longitud de la ventana de palabras que debe considerarse para que el algoritmo las tenga en cuenta como contexto al entrenar. Es decir, establece cuántas palabras por delante y por detrás de la actual se incluirán como contexto de esta. Por defecto tiene un valor de 5.

- **min_count**: Especifica la frecuencia mínima de una palabra en todo el corpus para que sea considerada en el vocabulario. Esto ayuda a eliminar palabras muy específicas que pueden no tener mucha importancia ya que aparecen muy raramente en los documentos. Por defecto tiene un valor de 5.

```
word2vec_model = Word2Vec(sentences=corpus, vector_size=100, window=5,
min_count=5, sg=0)
```

Figura 4-10. Código de la creación y entrenamiento del modelo Word2Vec

Word Embedding

Para realizar Word Embedding [22], se utiliza la librería Keras [25] para Deep Learning [27]. El proceso comienza con el uso del método Tokenizer para convertir los textos en secuencias de enteros, que son necesarios para la entrada en modelos de redes neuronales.

Primero se ajusta el tokenizer al conjunto de datos mediante fit_on_texts, creando un índice de palabras basado en la frecuencia de aparición. A continuación, se utiliza la función text_to_sequences para transformar los textos en secuencias de enteros, donde cada palabra del texto es reemplazada por su índice correspondiente.

Por último, la función pad_sequences garantiza que todas las secuencias tengan la misma longitud, rellenado con ceros o truncando según sea necesario.

```
tokenizer = Tokenizer(num_words = dim, oov_token = '<00V>')

tokenizer.fit_on_texts(df.tweet_text)


sequences = tokenizer.texts_to_sequences(df.tweet_text)
X = pad_sequences(
    sequences,
    maxlen = INPUT_LENGTH
)
y = pd.get_dummies(df.sentiment).values
```

Figura 4-11. Código para realizar Word Embedding

Vectorización del texto con CountVectorizer

Para realizar la vectorización de texto, se utiliza CountVectorizer [11] de la librería scikit-learn [28], para convertir el texto en una matriz de características basadas en conteos de palabras y n-gramas. En este caso, se consideran unigramas, bigramas y trigramas. La opción binary=true indica que se deben contar las presencias o ausencias de palabras, no las frecuencias.

```
cv = CountVectorizer(binary = True, ngram_range=(1, 3))
```

Figura 4-12. Código para realizar la vectorización del texto

Figura 4-13. N-gramas

4.2.3 Desarrollo del modelo de aprendizaje automático

4.2.3.1 Entrenamiento del modelo

Una vez obtenidos los vectores de tweets en la fase de extracción de características, se procede a implementar y evaluar diversos algoritmos de clasificación. La librería scikit-learn [28] ofrece herramientas para llevar a cabo la implementación de algoritmos de Machine Learning [26] que permite entrenar modelos y clasificar nuevos datos.

En esta etapa, se aplican varios modelos, incluyendo la Regresión Logística [10] y un modelo de Deep Learning [27] basado en una red neuronal recurrente bidireccional con una capa (RNN) [12]. El objetivo es seleccionar el modelo más eficiente según su rendimiento, evaluado mediante métricas como precisión y exactitud (accuracy). Para alcanzar este objetivo, se ajustan y determinan los parámetros relevantes en los distintos algoritmos, asegurando la mejor combinación de técnicas de extracción de características y algoritmos de clasificación supervisada.

Regresión Logística

En esta experimentación, se utilizará la Regresión Logística [10] como método de clasificación, implementado a través del módulo LogisticRegression de la librería scikit-learn [28]. La Regresión Logística [10] es una técnica de modelado que predice la probabilidad de una clase basada en una o más variables independientes, transformando la salida de una combinación lineal de las variables en una probabilidad mediante la función logística.

Para llevar a cabo el entrenamiento de este algoritmo, se explorarán diferentes configuraciones de hiperparámetros para ajustar el modelo. Los hiperparámetros que se consideran en esta experimentación son:

- **c**: Este parámetro regula la penalización por errores de clasificación en el modelo. Un valor pequeño de C indica una penalización baja por los puntos de datos mal clasificados, lo que tiende a generar un modelo con un margen más amplio, pero con mayor tolerancia a los errores de clasificación. Por el contrario, un valor grande de C implica una penalización alta por errores, lo que reduce el margen del modelo y tiende a ajustarse más a los datos, lo cual puede llevar a un modelo más complejo y propenso al sobreajuste.

- **penalty**: Define el tipo de regularización que se aplicará al modelo para evitar el sobreajuste.

- **class_weight:** Este parámetro controla como se manejan las clases desbalanceadas en el conjunto de datos, es crucial cuando se tiene un problema de clasificación con clases desbalanceadas, ya que evita que el modelo se sesgue hacia la clase mayoritaria.
- **solver:** Determina el algoritmo utilizado para optimizar la función de costo del modelo, este parámetro influye en la velocidad de entrenamiento y en la capacidad del modelo para converger a una solución óptima.

En la siguiente tabla, se presentan los diferentes valores de los hiperparámetros que se probaron durante la experimentación con el modelo de Regresión Logística [10]:

Hiperparámetro	Valores Propuestos
C	0.25, 0.5, 0.75
penalty	l1, l2
class_weignt	None, balanced
solver	saga

Tabla 4-1. Valores de hiperparámetros con los que se experimenta en Regresión Logística

El proceso de ajuste del modelo se realizó mediante GridSearchCV [29] de scikit-learn[28], el cual permite evaluar múltiples combinaciones de hiperparámetros a través de validación cruzada. Esto asegura que se seleccione el modelo con el mejor rendimiento en función de la métrica de precisión (accuracy).

En la Figura 4-14 se define un grid de parámetros a probar con todos los valores que hemos seleccionado en la Tabla 4-1. También se establece el número máximo de iteraciones para el proceso de optimización del modelo y la métrica a obtener. Finalmente, el modelo se entrena realizando la validación cruzada (cv = 5).

```
lr_cv_params = dict(C = [0.25, 0.5, 0.75],
                    penalty = ['l1', 'l2'],
                    class_weight = [None, 'balanced'],
                    solver = ['saga'])

lr_cv_grid = GridSearchCV(LogisticRegression(max_iter = 300),
                          lr_cv_params,
                          cv = 5,
                          n_jobs = -1,
                          scoring = 'accuracy')


lr_cv_grid.fit(X_train_cv, y_train)
```

Figura 4-14. Código de GridSearchCV para Regresión Logística

RNN – Bidirectional Layers

En esta experimentación, se utilizará una red neuronal recurrente bidireccional (RNN) [12] como modelo de clasificación. Esta red se implementa mediante el módulo Keras [25] de la biblioteca TensorFlow [30]. La arquitectura de la red incluye una capa de embedding, una capa bidireccional LSTM, y una capa densa con activación softmax.

Para llevar a cabo el entrenamiento de este algoritmo, se explorarán diferentes configuraciones de hiperparámetros para ajustar el modelo. Los hiperparámetros que se consideran en esta experimentación son:

- **dropout_val**: Este parámetro se utiliza en la capa LSTM para aplicar dropout, lo que ayuda a prevenir el sobreajuste al desactivar aleatoriamente un porcentaje de las unidades de la capa durante el entrenamiento.

- **batch_size**: Controla el número de muestras que se procesan antes de actualizar los pesos del modelo. Un tamaño de batch menor puede resultar en una convergencia más rápida, pero puede incrementar la variabilidad en el proceso de entrenamiento.

- **epochs**: Determina el número total de pasadas por todo el conjunto de datos durante el entrenamiento.

- **embedding_dim**: Define la dimensión del espacio de embedding para representar las palabras o entradas

- **layers**: Son los bloques básicos que componen el modelo. Cada capa procesa datos y transmite la salida a la siguiente capa

    - **Embedding**: Se utiliza una capa de embedding con una dimensión especificada por embedding_dim, que convierte las entradas en vectores densos de longitud fija.

    - **Bidirectional LSTM**: Una capa LSTM bidireccional con 20 unidades y dropout aplicado, lo que permite al modelo capturar dependencias temporales en ambas direcciones (hacia adelante y hacia atrás) en la secuencia de entrada.

    - **Dense**: Una capa densa final con 3 unidades y activación softmax, que proporciona probabilidades para cada una de las tres clases posibles.

En la siguiente tabla, se presentan los diferentes valores de los hiperparámetros que se probaron durante la experimentación con el modelo de red neuronal recurrente bidireccional (RNN) [12]:

Hiperparámetro	Valores Propuestos
dropout_val	0.6
batch_size	16
epochs	20
embedding_dim	130

Tabla 4-2. Valores de hiperparámetros con los que se experimenta RNN bidireccional

El modelo se compila utilizando el optimizador RMSprop, un algoritmo de optimización que ajusta los pesos del modelo de manera eficiente y estable, especialmente en redes neuronales recurrentes bidireccional (RNN) [12].

La función de pérdida utilizada es la entropía cruzada categórica (categorical_crossentropy), que mide la diferencia entre las predicciones del modelo y las etiquetas reales.

Para evaluar el rendimiento del modelo, se emplea la métrica de precisión (accuracy), que indica el porcentaje de predicciones correctas realizadas durante y después del entrenamiento. Además, se utilizan callbacks, como EarlyStopping y ModelCheckpoint, para detener el entrenamiento si la pérdida de validación deja de mejorar y para guardar el mejor modelo

basado en la precisión de validación, respectivamente, asegurando así un proceso de entrenamiento eficiente y un modelo final optimizado.

Para finalizar, el modelo se entrena utilizando un conjunto de datos de entrenamiento (X_train, y_train) y se valida con un conjunto de datos de prueba (X_test, y_test). El proceso de entrenamiento se lleva a cabo durante 20 épocas, con un tamaño de batch de 16, aplicando las configuraciones definidas y los callbacks mencionados en la Tabla 4-2.

```
# Compilamos modelo
bidirectional_model.compile(optimizer = 'rmsprop',
                            loss = 'categorical_crossentropy',
                            metrics = ['accuracy'])


# Creamos faststopping
callback = keras.callbacks.EarlyStopping(monitor = 'val_loss',
                                         mode = 'min',
                                         verbose = 0,
                                         patience = 3)


# Creamos checkpoint
checkpoint = keras.callbacks.ModelCheckpoint('../models/best_bidirectional_model.keras',
                                             monitor = 'val_accuracy',
                                             verbose = 1,
                                             save_best_only = True,
                                             mode = 'auto',
                                             save_weights_only = False)


# Entrenamos
history = bidirectional_model.fit( X_train,
                                   y_train,
                                   epochs = EPOCHS,
                                   validation_data = (X_test, y_test),
                                   batch_size = 16,
                                   callbacks = [callback, checkpoint])
```

Figura 4-15. Código de compilación, callbacks y entrenamiento de una RRN bidireccional

4.2.4 Aplicación y evaluación del modelo entrenado con datos reales

4.2.4.1 Obtención de Datos de Twitter [1]

Para la recolección de datos, se ha utilizado una técnica conocida como Web Scraping [13]. Este proceso implica emplear bots para extraer contenido y datos de un sitio web, incluyendo el código HTML [31]. En esencia, el Web Scraping [13] consiste en navegar automáticamente por una página web y recopilar información de ella. El software programado para realizar esta tarea se denomina bot. Los datos obtenidos, en este caso, tweets, se almacenan en formato CSV [18].

El Web Scraping [13] es legal siempre y cuando los datos recopilados estén disponibles públicamente para terceros en la web. No obstante, la legalidad de esta técnica puede variar, es fundamental considerar los derechos de propiedad intelectual de los sitios web para evitar problemas legales.

Para llevar a cabo el Web Scraping [13] o raspado de datos, se utiliza la biblioteca Selenium [32]. La versión actual de Selenium [32] se basa en HTML [31] y JavaScript [33], y permite automatizar la prueba y el registro de interacciones con aplicaciones web, así como repetir estas acciones de forma automática. Además, Selenium [32] es una herramienta de código abierto, lo que implica que no se requieren costos de licencia. Para simular las acciones del usuario en cualquier navegador, se emplea la interfaz Selenium WebDriver.

Selenium [32] puede ser utilizado con diversos navegadores. En este caso, se empleará Google Chrome [34] con la versión 128.0.6613.114 de 64 bits.

El siguiente paso es descargar e instalar el controlador del navegador con el que se va a conectar Selenium [32]. En este caso, para Google Chrome [34], el controlador necesario es ChromeDriver [35], que se puede descargar desde su página oficial en la versión compatible con el sistema. Considerando la versión del navegador, se descargará ChromeDriver 114.0.5735.90 [35].

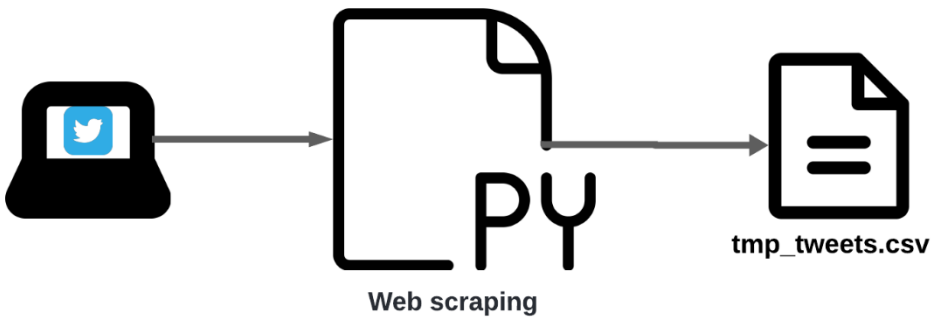


Figura 4-16. Diagrama de scraping de Twitter

4.2.4.2 Aplicación del modelo y análisis de los resultados

El modelo entrenado se aplica a los tweets preprocesados para clasificar el sentimiento de cada uno. Esta etapa consiste en pasar los datos de entrada a través del modelo para obtener predicciones sobre si los tweets expresan sentimientos positivos, negativos o neutros.

Los resultados del análisis de sentimientos se visualizaron utilizando Dash [15], una biblioteca de Python [17] para la creación de aplicaciones web interactivas, permite mostrar de manera efectiva las tendencias de sentimientos en Twitter [1] en un navegador, proporcionando una interfaz gráfica intuitiva para explorar y analizar los datos. Las visualizaciones generadas incluyen gráficos y tablas que facilitan la interpretación de los resultados y la identificación de patrones en los sentimientos expresados en los tweets.

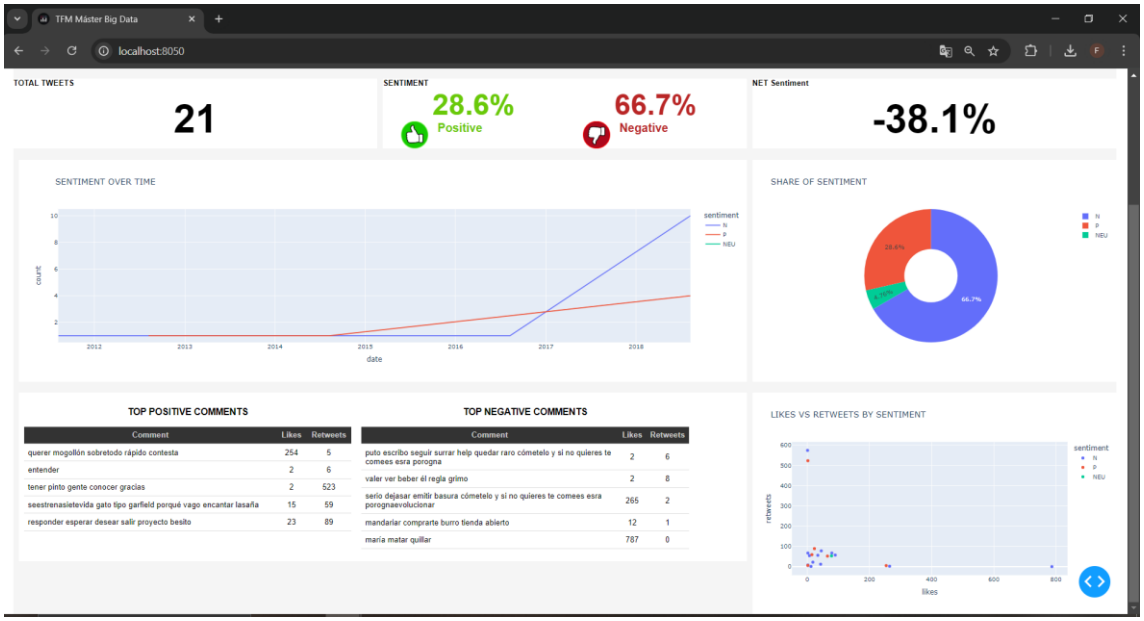


Figura 4-17. Visualización de los resultados obtenidos

4.2.5 Herramientas y Tecnologías Empleadas

Para el desarrollo y ejecución del proyecto se emplearon las siguientes herramientas y tecnologías:

- **Python**: [17] Lenguaje de programación principal para el desarrollo de scripts de scraping, el entrenamiento del modelo de aprendizaje automático y el análisis de datos.

- **Selenium**: Utilizado para la recolección de datos de Twitter [1] mediante Web Scraping [13]. Selenium [32] automatiza la navegación web y la extracción de tweets de manera eficiente.

- **ChromeDriver**: [35] Controlador del navegador Google Chrome [34] necesario para interactuar con la interfaz web de Twitter [1] a través de Selenium [32].

- **Pandas**: [36] Biblioteca para la manipulación y análisis de datos, proporcionando estructuras de datos flexibles y herramientas para leer y escribir en formatos como CSV [18].

- **NumPy**: [37] Biblioteca fundamental para la computación numérica, ofreciendo soporte para arrays multidimensionales y funciones matemáticas de alto rendimiento.

- **Matplotlib y Seaborn**: Matplotlib [38] y Seaborn [39] son bibliotecas de visualización de datos en Python [17]. Se utilizan para generar gráficos y tablas que facilitan la interpretación visual de los resultados del análisis de sentimientos.

- **TensorFlow y Keras**: Frameworks de aprendizaje automático utilizados para la construcción y entrenamiento de modelos de Deep Learning [27]. TensorFlow [30] proporciona la infraestructura, mientras que Keras [25] simplifica la construcción de modelos a través de una interfaz de alto nivel. Incluye herramientas para el preprocesamiento de texto como Tokenizer y pad_sequences.

- **Scikit-learn**: [28] Biblioteca para aprendizaje automático que incluye varios algoritmos y herramientas para la clasificación, regresión, análisis de grupos y evaluación de modelos. Incluye:

  - **LogisticRegression**: Algoritmo de Regresión Logística [10] para clasificación.

  - **GridSearchCV**: [29] Herramienta para la búsqueda de hiperparámetros.

  - **CountVectorizer**: [11] Para la extracción de características a partir de texto.

  - **train_test_split**: Para dividir conjuntos de datos en entrenamiento y prueba.

  - **classification_report y confusion_matrix**: Para evaluar el rendimiento del modelo.

- **NLTK**: [40] Biblioteca para el procesamiento del lenguaje natural, ofreciendo herramientas para la tokenización, lematización y análisis lingüístico. Incluye:

  - **nltk.tokenize**: Herramientas para la tokenización de texto.

  - **nltk.stem**: WordNetLemmatizer, se utiliza para la lematización de palabras.
  - **nltk.probability**: FreqDist para calcular la frecuencia de distribución de palabras.

- **Spacy**: [21] Biblioteca para el procesamiento del lenguaje natural, que ofrece herramientas avanzadas para el análisis lingüístico y el reconocimiento de entidades nombradas.

- **Beautiful Soup**: [14] Biblioteca de Python [17] para analizar documentos HTML [31], útil para realizar Web Scraping [13] y extraer información de sitios web.

- **Gensim**: [24] Biblioteca de código abierto para el procesamiento de textos y la construcción de modelos temáticos y de vectores de palabras como Word2Vec [23] y FastText.

- **HTML5 y CSS3**: HTML5 [31] es la última versión del lenguaje de marcado para el desarrollo web, y CSS3 [41] se utiliza para describir la presentación y el diseño de las páginas web.

- **Visual Studio Code**: [42] Editor de código fuente desarrollado por Microsoft, compatible con Windows, Linux y macOS. Ofrece soporte para depuración, control de versiones, resaltado de sintaxis y es altamente personalizable.

- **Jupyter Notebook**: [43] Interfaz web de código abierto que permite la inclusión de texto, vídeos, audio, imágenes y la ejecución de código en múltiples lenguajes a través del navegador, facilitando la exploración de datos y la creación de documentos reproducibles.

- **XML.etree.ElementTree**: [44] Módulo para trabajar con datos en formato XML, útil para el procesamiento y manipulación de documentos XML.

- **Emoji**: [20] Biblioteca para manejar y procesar emojis en texto, permitiendo su detección y manipulación.

- **rcParams**: Se importa de Matplotlib [38] para ajustar parámetros globales de configuración en las visualizaciones.

4.2.6 Dispositivos y Recursos Computacionales

El proyecto se desarrolló en una estación de trabajo con las siguientes especificaciones técnicas:

- **Procesador**: Intel(R) Core (TM) i7-8550U CPU a 1.80 GHz, con 4 núcleos físicos y 8 hilos lógicos. Este procesador proporciona un rendimiento adecuado para la ejecución de algoritmos complejos y el procesamiento intensivo de datos.

- **Memoria RAM**: 16 GB. Esta cantidad de memoria permite manejar grandes volúmenes de datos y ejecutar procesos simultáneos de manera eficiente, lo que es crucial para el entrenamiento de modelos de aprendizaje automático y el análisis de datos.
- **Almacenamiento**:
  - HDD: 118 GB. Utilizado para almacenamiento adicional de datos y archivos secundarios.

  - SSD: 915 GB. Proporciona un acceso rápido a los datos y una gestión eficiente del almacenamiento, lo que mejora significativamente la velocidad de lectura y escritura durante el procesamiento de grandes conjuntos de datos.

- **Modelo del Sistema**: HP Pavilion Laptop 15-cs0xxx. Este modelo de laptop combina portabilidad con rendimiento adecuado para tareas de desarrollo y análisis de datos.

Estas especificaciones garantizan un entorno de trabajo eficiente y un manejo fluido de grandes volúmenes de datos, facilitando el desarrollo y la ejecución del proyecto de manera óptima.

4.2.7 Modelos y Algoritmos Aplicados

Para el análisis de sentimientos, se aplicaron los siguientes modelos y algoritmos de aprendizaje automático:

4.2.7.1 Modelos de Clasificación

- **Regresión Logística**: [10] Este modelo se utilizó para la clasificación de sentimientos. Es adecuada para problemas de clasificación binaria y estima la probabilidad de que un tweet pertenezca a una clase específica en función de sus características. Su simplicidad y eficiencia permiten una rápida implementación y evaluación.

- **Red neuronal recurrente bidireccional (RNN)**: Se implementó una red neuronal recurrente bidireccional (RNN) [12] para capturar la información contextual del texto en ambas direcciones (hacia adelante y hacia atrás). Este enfoque permite al modelo captar mejor la relación y el contexto de las palabras en una secuencia, mejorando la precisión del análisis de sentimientos en comparación con métodos más simples.

4.2.7.2 Evaluación del Modelo

- **Precisión (Accuracy)**: La principal métrica utilizada para evaluar el rendimiento de los modelos fue la precisión. La precisión mide la proporción de clasificaciones correctas realizadas por el modelo sobre el total de clasificaciones. Esta métrica proporciona una visión general de la efectividad del modelo en la tarea de clasificación de sentimientos.

4.2.8 Análisis de Datos

Los resultados del análisis de los datos de Twitter [1], utilizando los modelos entrenados, fueron visualizados y presentados gráficamente. Para ello, se emplearon herramientas como Matplotlib [38] y Seaborn [39] para generar gráficos que permitieran una interpretación clara y efectiva de los datos. Las visualizaciones incluyeron gráficos de barras, histogramas y nubes de palabras, que facilitaron la identificación de patrones y tendencias en los sentimientos expresados en los tweets.

4.3 Recursos requeridos

Para la ejecución del proyecto se utilizaron los siguientes recursos:

- **Recursos Técnicos**
  - Hardware

    Procesador: Intel(R) Core (TM) i7-8550U CPU a 1.80 GHz.

    Memoria RAM: 16 GB.

    Almacenamiento: SSD de 915 GB y HDD de 118 GB.

    Modelo del Sistema: HP Pavilion Laptop 15-cs0xxx.

  - Software

    Sistema Operativo: Windows.

    Entorno de Desarrollo: Jupyter Notebook [43], Visual Studio Code [42].

    Librerías y Paquetes:

Python [17]: Pandas [36], NumPy [37], Matplotlib [38], Seaborn [39], TensorFlow [30], Keras [25], scikit-learn [28], NLTK [40], SpaCy [21], Beautiful Soup [14], Gensim [24].

Otros: Selenium [32], Dash [15], re [19], Emoji [20], xml.etree.ElementTree [44].

- Herramientas de Desarrollo

  Selenium: [32] Para la automatización de la extracción de datos de Twitter [1], Google Chrome [34] y ChromeDriver [35].

- **Dispositivos**

  - Computadora de Desarrollo: HP Pavilion Laptop 15-cs0xxx.

- **Material de Laboratorio**

  - No se requirió material de laboratorio físico.

- **Asistencia de Expertos**

  - Consultas y Asesoramiento: Reuniones periódicas con la tutora del TFM para recibir orientación y feedback en el desarrollo del proyecto. Estas reuniones fueron fundamentales para guiar el avance del proyecto y resolver dudas técnicas y metodológicas.

4.4 Presupuesto

El presupuesto mostrado en la Tabla 4-3 proporciona una visión general de los costes asociados con el desarrollo y ejecución de proyecto, incluye tanto los recursos materiales como el tiempo invertido.

Tipo de coste	Valor	Comentarios
Horas de trabajo en el proyecto	120 horas * 16 €/hora = 1920 €	Tiempo dedicado al desarrollo, análisis de datos y reuniones con la tutora.
Equipo técnico utilizado	1200 €	Valor aproximado de una HP Pavilion Laptop 15-cs0xxx similar en el mercado.

Software utilizado	0 €	Herramientas de código abierto: Python [17], TensorFlow [30], Keras [25], Selenium [32], Dash [15], etc.
Estudios e informes	0 €	No se ha adquirido ningún informe adicional o suscripciones.
Materiales empleados	0 €	No se requirieron materiales físicos o de laboratorio.
Asistencia de expertos	0 €	Reuniones de asesoramiento con la tutora del TFM incluidas dentro del tiempo del proyecto
Licencias o Servicios adicionales	0 €	Ningún software comercial adicional fue necesario.

Tabla 4-3. Costes de desarrollo del proyecto

4.5 Viabilidad

4.5.1 Análisis de viabilidad económica

El análisis de viabilidad económica del proyecto considera la relación coste/beneficio, evaluando si los beneficios potenciales justifican los costos que tiene el proyecto.

4.5.1.1 Costes

- **Horas de trabajo**: El principal coste asociado es el tiempo invertido en el desarrollo del proyecto.

- **Equipo técnico**: Incluye el valor aproximado del equipo utilizado en este proyecto.

- **Software y Herramientas**: En su mayoría, el software utilizado es de código abierto o gratuito, lo que minimiza los costos adicionales. Las herramientas como Visual Studio Code [42], Jupyter Notebook [43] y librerías de Python [17] no tienen un coste asociado

4.5.1.2 Beneficios

- **Innovación y Aprendizaje**: El proyecto ofrece un valioso aprendizaje en el uso de modelos de aprendizaje automático y procesamiento de lenguaje natural, útil a nivel académico y profesional.

- **Aplicación Práctica**: El análisis de sentimientos de tweets tiene aplicaciones prácticas en diversos campos como el marketing, la investigación de opinión pública y la mejora de servicios, facilitando la toma de decisiones basadas en datos de las empresas u organizaciones.

- **Ahorro en Costes de Software**: El uso de herramientas gratuitas y de código abierto hace que el proyecto sea económicamente viable.

4.5.2 Análisis de sostenibilidad a futuro

- **Mantenimiento**: Los modelos de aprendizaje automático pueden requerir ajustes y reentrenamiento con nuevos datos para mantener su precisión. La infraestructura actual permite escalar este proceso.

- **Escalabilidad**: El uso de herramientas y librerías modernas asegura que el proyecto pueda adaptarse a futuras actualizaciones tecnológicas sin necesidad de cambios significativos.

- **Relevancia**: El análisis de sentimientos seguirá siendo útil dado el crecimiento de los datos en redes sociales.

- **Economía**: El uso de herramientas y librerías de código abierto minimiza costos, lo que facilita el mantenimiento a largo plazo sin grandes gastos adicionales

4.6 Resultados del proyecto

En este apartado se presentan los resultados obtenidos tras la implementación de los modelos. Se realizará una comparación entre ellos y se seleccionará el mejor modelo basado en la métrica de evaluación utilizada.

Las métricas de validación que se han utilizado son las siguientes:

- **Accuracy (Exactitud)**: Es la métrica de evaluación más sencilla y ampliamente utilizada. Se calcula dividiendo el número de predicciones correctas entre el número total de predicciones realizadas. Proporciona una visión general rápida del rendimiento de un clasificador, siendo más útil cuando el número de ejemplos en cada clase es aproximadamente equivalente. En este contexto, cuanto más se acerque al valor 1, mayor será la exactitud del modelo.

$$\boldsymbol{accuracy} = \frac{\boldsymbol{nº\ predicciones\ correctas}}{\boldsymbol{nº\ total\ de\ predicciones}} = \frac{\boldsymbol{TP + TN}}{\boldsymbol{TP + TN + FP + FN}}$$

- **F1-score**: Es una métrica que combina la precisión y la sensibilidad (recall) en una sola medida. El F1-score alcanza su valor óptimo (1) solo si tanto la precisión como la exhaustividad son del 100%. Si una de estas métricas es igual a 0, el F1-score también será 0

$$\boldsymbol{F1} = \boldsymbol{2} \times \frac{\boldsymbol{Recall\ \times Precision}}{\boldsymbol{Recall + Precison}}$$

- **Precisión (Precision)**: Es una métrica que evalúa la exactitud de las predicciones positivas realizadas por el modelo. Una alta precisión indica que el modelo tiene un bajo número de falsos positivos.

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

- **Recall**: También conocida como sensibilidad, es una métrica de evaluación utilizada en problemas de clasificación que mide la proporción de casos positivos identificados correctamente por el modelo sobre el total de casos positivos presentes en los datos.

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

- **Matriz de confusión**: Es una herramienta utilizada en problemas de clasificación para evaluar el rendimiento de un modelo de aprendizaje automático. La matriz de confusión muestra la cantidad de casos que el modelo ha clasificado correcta e incorrectamente en cada clase. Las predicciones correctas se encuentran en una línea diagonal que se desplaza de la parte superior izquierda a la inferior derecha.

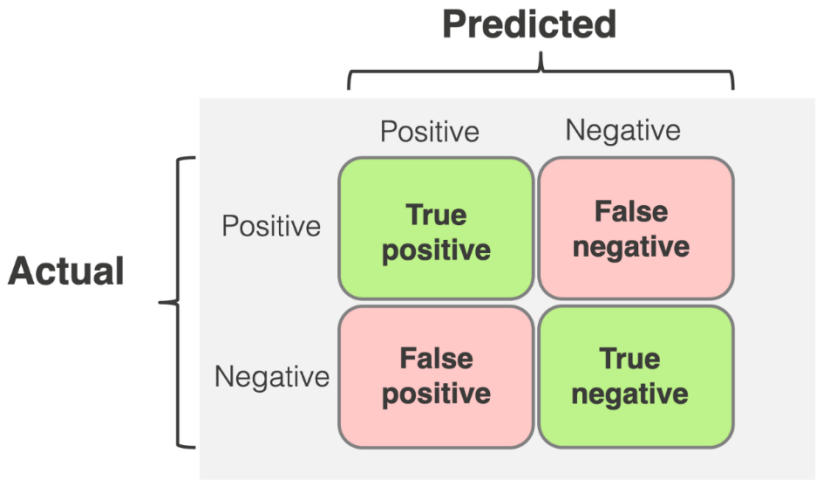


Figura 4-18. Estructura de la matriz de confusión

En ella se define las siguientes celdas:

- **True Positive (TP) - Verdaderos Positivos**: Número de instancias que fueron clasificadas correctamente como positivas por el modelo.
- **True Negative (TN) - Verdaderos Negativos**: Número de instancias que fueron clasificadas correctamente como negativas por el modelo.
- **False Positive (FP) - Falsos Positivos**: Número de instancias que fueron incorrectamente clasificadas como positivas cuando en realidad son negativas.
- **False Negative (FN) - Falsos Negativos**: Número de instancias que fueron incorrectamente clasificadas como negativas cuando en realidad son positivas.

4.6.1 Modelos de Aprendizaje Automático

4.6.1.1 Evaluación del Modelo de Regresión Logística

En este apartado se evaluarán los resultados de la experimentación con el modelo de Regresión Logística con los hiperparámetros descritos en la Tabla 4-1. El experimento se realizó utilizando un conjunto de datos compuesto por 386,600 tweets en español. Este conjunto de datos se distribuye de la siguiente manera:

- **Tweets con polaridad negativa**: 134,706

- **Tweets con polaridad positiva**: 159,600

- **Tweets con polaridad neutral**: 92,294

Evaluación con Word2Vec

Con Word2Vec [23], el modelo de Regresión logística [10] obtuvo una exactitud (accuracy) de 0.6610. Al evaluar este modelo con el conjunto de datos de prueba, los resultados fueron los siguientes:

- **Accuracy**: La Figura 4-19 indica que el modelo clasifica los tweets correctamente en un 66.1% de los casos

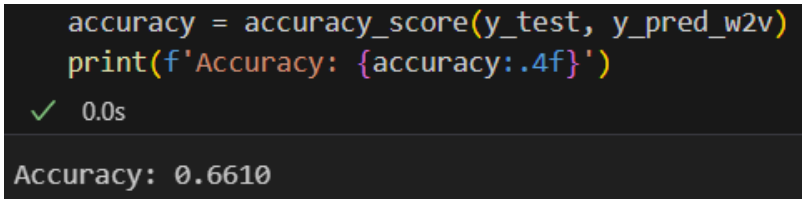


Figura 4-19. Código de accuracy de Regresión Logística con Word2Vec

- **Matriz de confusión**: En la Figura 4-20 se puede notar un mejor rendimiento en la clasificación de la clase negativa (0), con una alta tasa de verdaderos negativos (0.726942) y una menor cantidad de falsos positivos. Sin embargo, el modelo presenta dificultades en la identificación de la clase neutral (1), evidenciada por un aumento en los falsos positivos y falsos negativos.

  Por otro lado, la clase positiva (2) muestra una cantidad considerable de verdaderos positivos (0.752457), lo que indica una mayor precisión al clasificar tweets con sentimientos positivos.

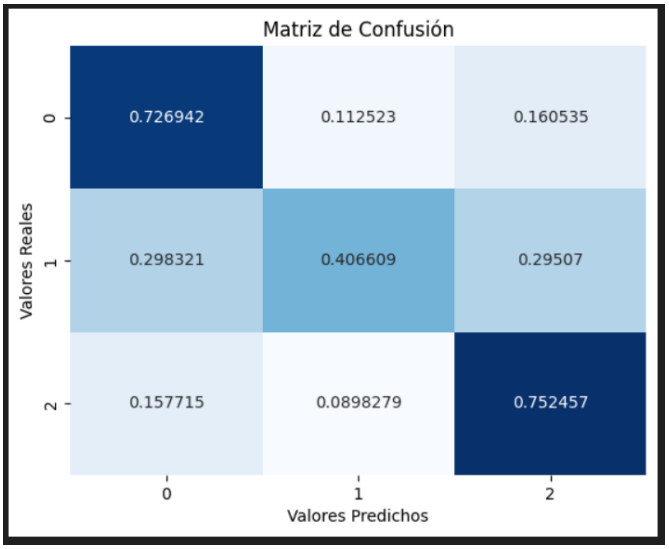


Figura 4-20. Matriz de confusión de Regresión Logística con Word2Vec

- **Informe de Clasificación**: En la Figura 4-21 se observa una precisión general (accuracy) de 0.66, lo que indica que el modelo clasifica correctamente el 66% de los tweets.

  La clase positiva (1) muestra el mejor rendimiento, con una precisión de 0.71, un recall de 0.75 y un f1-score de 0.73, lo que refleja un buen equilibrio entre precisión (precisión) y sensibilidad (recall).

  En la clase negativa (-1), se obtiene una precisión de 0.65 y un recall de 0.73, lo que sugiere que el modelo clasifica correctamente una mayor proporción de tweets negativos, aunque con algunos falsos positivos.

  La clase neutral (0), presenta el rendimiento más bajo, con una precisión de 0.56 y un recall de 0.41, lo que indica dificultades en la correcta clasificación de esta clase.
  En términos generales, el modelo logra una media ponderada del f1-score de 0.65, lo que sugiere un desempeño aceptable pero mejorable, especialmente en la clase neutral.

	precision	recall	f1-score	support
-1	0.65	0.73	0.69	26910
0	0.56	0.41	0.47	18460
1	0.71	0.75	0.73	31950
accuracy			0.66	77320
macro avg	0.64	0.63	0.63	77320
weighted avg	0.65	0.66	0.65	77320

Figura 4-21. Informe de clasificación de Regresión Logística con Word2Vec

Evaluación con CountVectorizer

Con CountVectorizer [11], el modelo de Regresión Logística [10] obtuvo una exactitud (accuracy) de 0.71994. Al evaluar este modelo con el conjunto de datos de prueba, los resultados fueron los siguientes:

- **Accuracy**: La Figura 4-22 indica que el modelo clasifica los tweets correctamente en un 71.9% de los casos.

```
accuracy_lr_cv = accuracy_score(y_test, y_pred_log_reg_cv)
accuracy_lr_cv
```

```
0.719943093636834
```

Figura 4-22. Código de accuracy de Regresión Logística con CountVectorizer

- **Matriz de confusión**: En la Figura 4-23 se puede notar un mejor rendimiento en la clasificación de la clase negativa (0), con una alta tasa de verdaderos negativos (0.724266) y una menor cantidad de falsos positivos. Además, el modelo presenta un menor rendimiento en la identificación de la clase neutral (1), evidenciada por un aumento en los falsos positivos y falsos negativos.

  Por otro lado, la clase positiva (2) muestra una cantidad considerable de verdaderos positivos (0.775962), lo que indica una mayor precisión al clasificar tweets con sentimientos positivos.

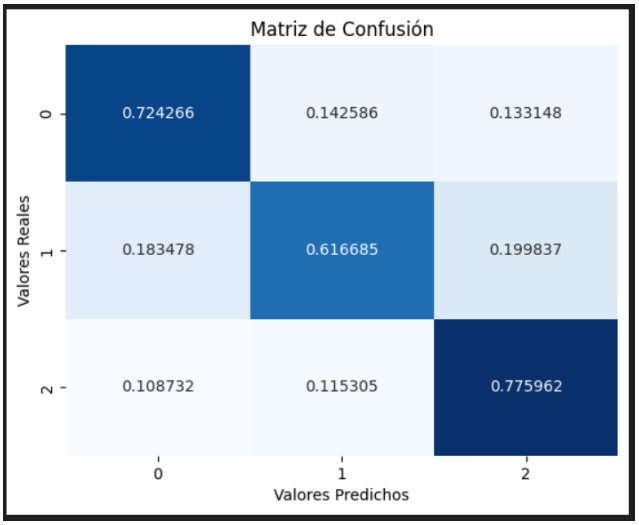


Figura 4-23. Matriz de confusión de Regresión Logística con CountVectorizer

- **Informe de Clasificación**: En la Figura 4-24 se observa una precisión general (accuracy) de 0.72, lo que indica que el modelo clasifica correctamente el 72% de los tweets.

  La clase positiva (1) es la que presenta el mejor rendimiento con una precisión de 0.77, un recall de 0.78 y un f1-score de 0.77, lo que indica un excelente equilibrio entre precisión y exhaustividad para esta clase.

  En la clase negativa (-1), se obtiene una precisión de 0.74 y un recall de 0.72, lo que refleja un buen desempeño en la identificación correcta de tweets negativos, aunque aún existe un pequeño margen de error en la clasificación de tweets negativos como positivos.

  La clase neutral (0), con una precisión de 0.60 y un recall de 0.62, es la más desafiante para el modelo, lo que sugiere que le resulta más difícil diferenciar los tweets neutrales de los de otras polaridades.

  En términos generales, se observa una media macro de f1-score de 0.71, lo que sugiere un rendimiento equilibrado entre todas las clases. El promedio ponderado del f1-score es de 0.72, destacando el rendimiento aceptable del modelo, aunque se podría mejorar la clasificación en la clase neutral para obtener un mejor equilibrio en los resultados.

	precision	recall	f1-score	support
-1	0.74	0.72	0.73	26910
0	0.60	0.62	0.61	18460
1	0.77	0.78	0.77	31950
accuracy			0.72	77320
macro avg	0.71	0.71	0.71	77320
weighted avg	0.72	0.72	0.72	77320

Figura 4-24. Informe de clasificación de Regresión Logística con CountVectorizer

4.6.1.2 Evaluación del Modelo con una RNN bidireccional

En este apartado se evaluarán los resultados de la experimentación con el modelo con una red neuronal recurrente bidireccional (RNN) [12] con los hiperparámetros descritos en la Tabla 4-2.

Este modelo obtuvo una exactitud (accuracy) de 0.7106. Al evaluar el modelo con el conjunto de datos de prueba, los resultados fueron los siguientes:

- **Accuracy**: La Figura 4-25 indica que el modelo clasifica los tweets correctamente en un 71.06% de los casos.

```
accuracy_bi = bidirectional_model.evaluate(X_test, y_test)
accuracy_bi
```

```
2417/2417 ━━━━━━━━━━━━━━━━━━━━ 38s 16ms/step - accuracy: 0.7106 - loss: 0.6783

[0.6809850335121155, 0.7083807587623596]
```

Figura 4-25. Código de accuracy de RNN bidireccional

- **Matriz de confusión**: En la Figura 4-26 se puede notar un mejor rendimiento en la clasificación de la clase negativa (0), con una alta tasa de verdaderos negativos (0.758194) y una menor cantidad de falsos positivos. Además, el modelo presenta un menor rendimiento en la identificación de la clase neutral (1), evidenciada por un aumento en los falsos positivos y falsos negativos.

Por otro lado, la clase positiva (2) muestra una cantidad considerable de verdaderos positivos (0.777089), lo que indica una mayor precisión al clasificar tweets con sentimientos positivos.

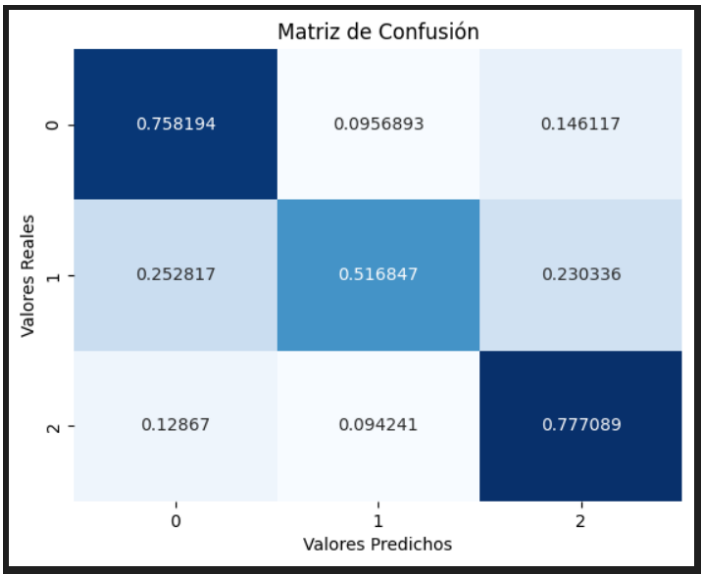


Figura 4-26. Matriz de confusión de RNN bidireccional

- **Informe de Clasificación:** En la Figura 4-27 se observa una precisión general (accuracy) de 0.71, lo que indica que el modelo clasifica correctamente el 71% de los tweets.
  La clase positiva (2) es la que presenta el mejor rendimiento con una precisión de 0.75, un recall de 0.78 y un f1-score de 0.76, lo que indica que el modelo identifica correctamente la mayoría de los tweets positivos, con un buen equilibrio entre falsos positivos y falsos negativos.

  En la clase negativa (0), se obtiene una precisión de 0.70 y un recall de 0.76, lo que refleja un buen rendimiento en la clasificación de tweets negativos. Sin embargo, algunos tweets de esta clase se clasifican incorrectamente en otras categorías.

  La clase neutral (1), con una precisión de 0.63 y un recall de 0.52, es la más desafiante para el modelo, lo que sugiere que le resulta más difícil identificar correctamente los tweets neutros, clasificándolos erróneamente como negativos o positivos en una mayor proporción.

  En resumen, el modelo tiene un f1-score ponderado de 0.70, lo que indica un rendimiento global equilibrado, aunque con margen de mejora en la clasificación de los tweets neutros.

```
2417/2417 ━━━━━━━━━━━━━━━━━━━━ 37s 15ms/step
              precision    recall  f1-score   support

           0       0.70      0.76      0.73     26910
           1       0.63      0.52      0.57     18460
           2       0.75      0.78      0.76     31950

    accuracy                           0.71     77320
   macro avg       0.69      0.68      0.69     77320
weighted avg       0.70      0.71      0.70     77320
```

Figura 4-27. Informe de clasificación de RNN bidireccional

De los tres modelos presentados, el modelo de Regresión Logística [10] con CountVectorizer [11] se considera el más adecuado para su utilización. Este modelo presenta el mayor f1-score ponderado (0,72) y la mayor exactitud (72%), lo que indica un buen rendimiento general.

Además, se destaca por su simplicidad en términos de implementación y entrenamiento, en comparación con modelos más complejos, como las redes neuronales recurrentes bidireccionales (RNN). Por estas razones, el modelo de Regresión Logística [10] con CountVectorizer [11] es la opción más eficiente y práctica para esta tarea.

Métrica	RNN (Red Neuronal Recurrente)	Regresión Logística (CountVectorizer)	Regresión Logística (Word2Vec)
Accuracy	**71 %**	**72 %**	**66 %**
F1-Score	**70**	**72 %**	**65 %**
Precision	**70**	**72 %**	**66 %**
Recall	**71**	**72 %**	**65 %**

Tabla 4-4. Resultados obtenidos en la evaluación

4.6.2 Análisis de sentimientos

Al modelo entrenado seleccionado se le aplicó un conjunto de datos (tweets) preprocesados. Los resultados mostraron una distribución aceptable de sentimientos (positivos, negativos, neutral). Para la visualización de los resultados, se utilizó Dash [15] para crear paneles

interactivos. Los gráficos y visualización incluyen información como el total de tweets obtenidos, el porcentaje de tweets negativos y positivos, los tweets tanto negativos como positivos más relevantes y otros más.

4.6.3 Cambios durante el proyecto

El proyecto inicialmente incluía modelos como máquinas de soporte vectorial (SVM) y bosques aleatorios, pero se decidió centrarse en la Regresión Logística [10] y en el modelo de red neuronal recurrente bidireccional (RNN) [12] por su eficacia en el análisis de sentimientos. Además, se optimizaron los hiperparámetros para mejorar el rendimiento y prevenir el sobreajuste. También se implementaron técnicas avanzadas de preprocesamiento para elevar la calidad de los datos y, por ende, el rendimiento de los modelos.

4.6.4 Desarrollo de actividades

Durante el desarrollo del proyecto, se llevaron a cabo reuniones periódicas con la tutora del TFM para recibir orientación y ajustar la metodología, mejorando así el enfoque del proyecto.

Se emplearon herramientas y librerías para el procesamiento de datos, el desarrollo de modelos y la visualización de resultados, facilitando tanto la implementación como la presentación. Además, se realizó una evaluación y validación de los modelos, utilizando visualizaciones interactivas en Dash [15] para una mejor interpretación de los resultados obtenidos a partir de los datos analizados.

Capítulo 5. DISCUSIÓN

Una vez concluido el trabajo, es esencial revisar los pasos y decisiones tomadas durante la construcción del sistema de clasificación de sentimiento basado en su polaridad negativa, positiva y neutral.

- **Resultados principales**: Los resultados indicaron que el modelo de Regresión Logística [10] con CountVectorizer [11] fue el más adecuado para el análisis de sentimientos, destacando por su alto F1-score ponderado (0,72) y una exactitud del 72%. Este modelo demostró ser efectivo para manejar secuencias de texto y ofreció una solución más sencilla en comparación con modelos más complejos.

- **Limitaciones del estudio**: Aunque los resultados son positivos, el estudio tiene limitaciones. La principal es la posible falta de generalización del modelo a otros conjuntos de datos debido a la especificidad del conjunto de datos de tweets. Además, la metodología empleada no captura completamente la complejidad del lenguaje natural.

- **Limitaciones de la tecnología empleada**: Las herramientas y librerías utilizadas, fueron fundamentales para el desarrollo del proyecto, pero presentan limitaciones. CountVectorizer [11], por ejemplo, puede no captar toda la semántica y el contexto de las palabras, lo que podría impactar la precisión en análisis más complejos.

- **Cambios respecto a objetivos planteados inicialmente**: El proyecto inicialmente contemplaba una variedad más amplia de modelos de clasificación, como máquinas de soporte vectorial y bosques aleatorios. Sin embargo, se optó por concentrarse en la Regresión Logística [10] y la red neuronal recurrente bidireccional (RNN) [12] debido a su mejor adecuación para el análisis de sentimientos, lo que permitió una alineación más precisa con los objetivos del proyecto y una implementación más efectiva.

- **Adaptación a cambios y Metodología**: Aunque la metodología inicial fue útil, se realizaron ajustes significativos a lo largo del proyecto. La optimización de hiperparámetros y la incorporación de técnicas avanzadas de preprocesamiento, como la lematización y la eliminación de stopwords, fueron cruciales para mejorar el rendimiento del modelo.

- **Impacto de los Resultados**: Los resultados proporcionaron una visión valiosa del análisis de sentimientos en tweets, demostrando la efectividad de los enfoques seleccionados. La visualización interactiva en Dash [15] facilitó una interpretación más profunda y accesible de los resultados obtenidos, mejorando la evaluación de los datos analizados.

En resumen, el proyecto logró cumplir sus objetivos al implementar y evaluar modelos de aprendizaje automático para el análisis de sentimientos en tweets, proporcionando una herramienta útil para la interpretación de datos y el análisis de opiniones en redes sociales.

Capítulo 6. CONCLUSIONES

6.1 Conclusiones del trabajo

En este trabajo se estudiaron e implementaron técnicas de Machine Learning [26] y Deep Learning [27] para llevar a cabo el análisis de sentimientos en redes sociales (Twitter [1]). Además, se pudo ver la teoría que hay detrás de estos algoritmos, los cuales trabajan y aprenden de forma diferente.

A través del análisis de trabajos previos, se demostró la viabilidad de construir un sistema de reconocimiento de sentimientos con un nivel de calidad adecuado para un Trabajo de Fin de Máster. Durante el desarrollo del TFM, se detalló cada fase del proceso de manera ilustrativa y didáctica, facilitando la comprensión de los procedimientos para lectores nuevos en la materia.

El trabajo se elaboró mayoritariamente en Python [17], aprovechando su versatilidad y las potentes librerías como NLTK [40] para el procesamiento del lenguaje, Selenium [32] para la extracción de datos de Twitter [1], y librerías comúnmente utilizadas como scikit-learn[28], Matplotlib [38] y Pandas [36].

El sistema de clasificación de sentimientos basado en CountVectorizer [11] con un clasificador de Regresión Logística [10] demostró ser el más eficaz, logrando una medida F1 ponderada del 72%, lo que refuerza la idea de que el trabajo se completó con éxito. Este proyecto permitió aplicar y profundizar técnicas aprendidas en asignaturas como Aprendizaje Automático, Procesamiento de datos.

A pesar de las restricciones temporales y del carácter introductorio del trabajo, se plantean posibles ampliaciones para mejorar los resultados y aumentar la competitividad del sistema construido.

6.2 Conclusiones personales

El desarrollo de este proyecto fue una experiencia enriquecedora y educativa que permitió aplicar en profundidad las técnicas aprendidas durante el máster, resultando gratificante tanto a nivel profesional como personal.

La oportunidad de trabajar con herramientas avanzadas y enfrentar los desafíos del procesamiento de lenguaje natural amplió la comprensión y habilidades en el campo. Además, la realización del proyecto subrayó la importancia de la adaptabilidad y la resolución de problemas en proyectos complejos.

Las lecciones aprendidas y los conocimientos adquiridos tienen una relevancia significativa para una futura carrera profesional, proporcionando una base sólida para abordar proyectos más ambiciosos en el ámbito del aprendizaje automático y el análisis de datos

Capítulo 7. FUTURAS LÍNEAS DE TRABAJO

El trabajo se centró en el análisis de sentimientos en textos provenientes de Twitter [1], una red social específica. Este enfoque incluyó el preprocesamiento adaptado a esta plataforma en particular. Sin embargo, modificando estas fases, el análisis podría extrapolarse a otras formas de comunicación textual, como textos de opinión, libros, artículos de crítica, entradas en blogs y cartas. Aunque Twitter [1] fue el principal foco, se podrían analizar otras redes sociales donde la comunicación textual es relevante, como Facebook o Telegram.

Con respecto al idioma, se tomó la decisión de trabajar con textos en castellano, pero sería interesante ampliar el trabajo hacia otros idiomas y así poder probar la influencia que la lengua tiene en los análisis, como para poder probar técnicas de traducción o transferencia léxica.

Además, se identificaron limitaciones relacionadas con la potencia computacional disponible. Ampliar el número de modelos probados y el conjunto de datos podría mejorar el rendimiento del sistema. También se propone investigar técnicas semisupervisadas que partan de pequeños conjuntos de datos etiquetados manualmente para luego expandirlos, lo que podría validar y mejorar el sistema de etiquetado inicial.

Por último, durante el proceso de recolección de datos a través de Web Scraping [13] en Twitter [1], se encontraron varias limitaciones técnicas que obstaculizan la automatización a largo plazo. Uno de los principales problemas fue la aparición de restricciones tras realizar varias sesiones de scraping. En este punto, la plataforma solicita una verificación adicional, lo que introduce una barrera significativa para la automatización. Además, dependiendo de si la cuenta es verificada o no, el usuario puede acceder a un número limitado de tweets diarios. Como alternativa viable se propone, el uso de la API de Twitter [50], no evita los bloqueos, sino que permite el acceso estructurado y legal a los datos de la plataforma. Aunque esta opción requiere un plan de pago en función del volumen de datos necesarios ya que tiene un plan gratuito, pero no permite la recolección de datos.

Capítulo 8. REFERENCIAS

Federico Pascual. “Getting Started with Sentiment Analysis using Python”. [En línea]. Disponible: https://huggingface.co/blog/sentiment-analysis-python. [Citado: 16 se septiembre de 2024]

Praveenr. “CountVectorizer vs TFIDF - Logistic Regression”. [En línea]. Disponible: https://dev.to/praveenr2998/countvectorizer-vs-tfidf-logistic-regression-3heb. [Citado: 16 se septiembre de 2024]

Samhita Alla. “Advanced Recurrent Neural Networks: Bidirectional RNNs”. [En línea]. Disponible: https://blog.paperspace.com/bidirectional-rnn-keras/. [Citado: 16 se septiembre de 2024]

Plotly. “Dash in 20 Minutes”. [En línea]. Disponible: https://dash.plotly.com/tutorial. [Citado: 16 de septiembre de 2024]

M. Al-Mulhem, Advances in Deep Learning for Biomedical Engineering. Cham, Switzerland: Springer, 2023. [En línea]. Disponible: https://link.springer.com/book/10.1007/978-3-031-02145-9. [Citado: 16 de septiembre de 2024].

[1] Twitter. [En línea]. Disponible en: https://x.com

[2] Service Hub de HubSpot. [En línea]. Disponible en: https://www.hubspot.com/

[3] Social mention. [En línea]. Disponible en: https://www.socialmention.com/

[4] Brand24. [En línea]. Disponible en: https://brand24.com/

[5] Repustar. [En línea]. Disponible en: https://www.repustate.com/

[6] Lexalíticos. [En línea]. Disponible en: https://www.lexalytics.com/

[7] Kaggle. [En línea]. Disponible en: https://www.kaggle.com/

[8] Github. [En línea]. Disponible en: https://github.com/

[9] TASS. [En línea]. Disponible en: http://tass.sepln.org/

[10] Regresión Logística. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Regresi%C3%B3n_log%C3%ADstica

[11] CountVectorizer - Scikit-learn feature extraction. [En línea]. Disponible en: https://scikit-learn.org/

[12] Red neuronal recurrente. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/recurrent-neural-network/#:~:text=Una%20red%20neuronal%20recurrente%20bidireccional,para%20predecir%20la%20salida%20posterior.

[13] Web Scraping. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Web_scraping#:~:text=Web%20scraping%20o%20raspado%20web,un%20navegador%20en%20una%20aplicaci%C3%B3n.

[14] Beautiful Soup. [En línea]. Disponible en: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[15] Dash. [En línea]. Disponible en: https://dash.plotly.com/

[16] datos.gob.es. [En línea]. Disponible en: https://datos.gob.es/en/catalogo

[17] Python. [En línea]. Disponible en: https://www.python.org/

[18] CSV. [En línea]. Disponible en: https://www.geeknetic.es/Archivo-CSV/que-es-y-para-que-sirve

[19] Regular expression operations (re). [En línea]. Disponible en: https://docs.python.org/es/3/library/re.html

[20] Emoji. [En línea]. Disponible en: https://pypi.org/project/emoji/

[21] Spacy. [En línea]. Disponible en: https://spacy.io/

[22] Word Embedding. [En línea]. Disponible en: https://www.athos-cap.com/post-11-word-embeddings-el-corazon-de-la-revolucion-del-nlp/

[23] Word2Vec. [En línea]. Disponible en: https://datascientest.com/es/nlp-word-embedding-word2vec-es

[24] Gensim. [En línea]. Disponible en: https://radimrehurek.com/gensim/

[25] Keras. [En línea]. Disponible en: https://keras.io/

[26] Machine Learning. [En línea]. Disponible en: https://www.ibm.com/es-es/topics/machine-learning

[27] Deep Learning. [En línea]. Disponible en: https://www.ibm.com/es-es/topics/deep-learning

[28] scikit-learn. [En línea]. Disponible en: https://scikit-learn.org/stable/

[29] GridSearchCV. [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[30] TensorFlow. [En línea]. Disponible en: https://www.tensorflow.org/?hl=es

[31] HTML. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTML

[32] Selenium. [En línea]. Disponible en: https://www.selenium.dev/

[33] JavaScript. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript

[34] Google Chrome. [En línea]. Disponible en: https://www.google.es/chrome/index.html

[35] ChromeDriver. [En línea]. Disponible en: https://developer.chrome.com/docs/chromedriver/downloads?hl=es-419

[36] Pandas. [En línea]. Disponible en: https://pandas.pydata.org/

[37] NumPy. [En línea]. Disponible en: https://numpy.org/

[38] Matplotlib. [En línea]. Disponible en: https://matplotlib.org/

[39] Seaborn. [En línea]. Disponible en: https://seaborn.pydata.org/

[40] NLTK. [En línea]. Disponible en: https://www.nltk.org/

[41] CSS3. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/CSS

[42] Visual Studio Code. [En línea]. Disponible en: https://code.visualstudio.com/

[43] Jupyter Notebook. [En línea]. Disponible en: https://jupyter.org/

[44] XML.etree.ElementTree. [En línea]. Disponible en: https://docs.python.org/3/library/xml.etree.elementtree.html

[45] Tweets Sentiment Worldcup. [En línea]. Disponible en: https://github.com/charlesmalafosse/open-dataset-for-sentiment-analysis/tree/master

[46] Tweets Sentiment Teams. [En línea]. Disponible en: https://github.com/charlesmalafosse/open-dataset-for-sentiment-analysis/tree/master

[47] IMDB Dataset Spanish. [En línea]. Disponible en: https://www.kaggle.com/datasets/luisdiegofv97/imdb-dataset-of-50k-movie-reviews-spanish

[48] Segura Dataset. [En línea]. Disponible en: https://datos.gob.es/en/catalogo

[49] TASS2019 Country ES Train. [En línea]. Disponible en: http://tass.sepln.org/

[50] Twitter API. [En línea]. Disponible en: https://developer.x.com/en/docs/x-api

Capítulo 9. ANEXOS

9.1 Anexo 1: Estructura de directorios

La estructura de directorios en que se organizan los ficheros de datos y programas del proyecto es la que se muestra en la Figura 9-1. El directorio tweetSentiment contiene todo lo referido al trabajo realizado.

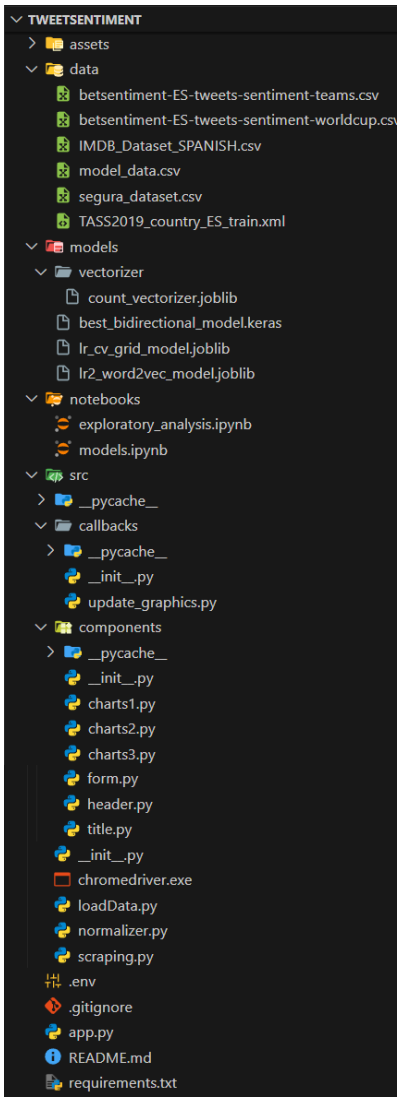


Figura 9-1. Estructura de directorios

9.2 Anexo 2: Guía de uso

En este anexo se explica cómo utilizar la herramienta para que el usuario la pueda usar.

Requisitos previos

- Tener instalado Google Chrome [34].
- Contar con una cuenta de Twitter [1] para acceder a los datos.
- Es necesario contar con Python [17] versión 3 instalada y si se está trabajando en una máquina Windows, será necesario tener como variable del sistema para poder ejecutar Python [17].
- Instalar las librerías necesarias para el funcionamiento del proyecto. Estas se encuentran en el archivo requirements.txt. Para instalar las librerías basta con ejecutar el siguiente comando

pip install -r requirements.txt

Pasos para utilizar la aplicación

Primero se debe ubicar el archivo principal app.py dentro de la carpeta del proyecto. A continuación, se abre una terminal o línea de comandos en la carpeta que contiene dicho archivo y se ejecuta el siguiente comando:

python app.py

Este comando iniciará el servidor local de la aplicación. Una vez que el servidor esté en funcionamiento, se abre un navegador web y se ingresa la dirección: http://localhost:8050/. Esto cargará la interfaz principal de la aplicación.

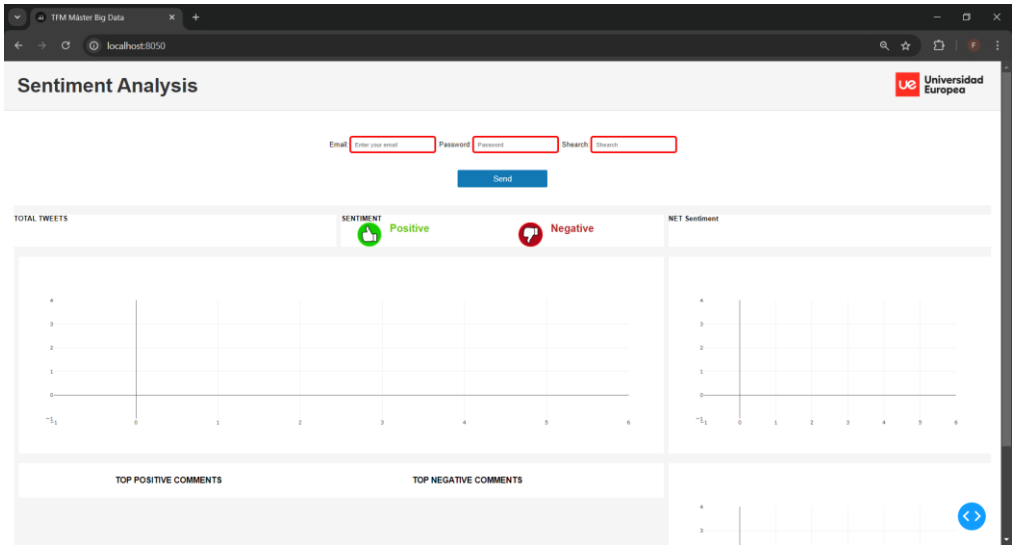


Figura 9-2. Interfaz principal de la herramienta

En la parte superior de la interfaz, se encuentra un formulario donde es necesario ingresar las credenciales de Twitter [1], como el correo electrónico y la contraseña, además del parámetro de búsqueda deseado. Este parámetro será utilizado por la herramienta para realizar el scraping en Twitter [1].

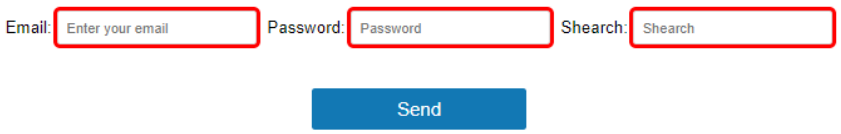


Figura 9-3. Formulario de la herramienta

Tras introducir los datos y presionar "send", la aplicación realizará automáticamente el scraping en Twitter [1], recopilando tweets relacionados con el término de búsqueda proporcionado. Posteriormente, se aplicará el modelo de análisis de sentimientos a los datos extraídos.

Finalmente, los resultados se presentarán en la misma ventana, mediante gráficos que muestran el porcentaje de tweets positivos y negativos, el net sentiment (sentimiento neto), entre otros análisis visuales. Estos gráficos permiten al usuario obtener una visión clara de la percepción general en Twitter [1] sobre el tema o la entidad buscada, facilitando la toma de decisiones estratégicas basadas en la opinión pública extraída en tiempo real.

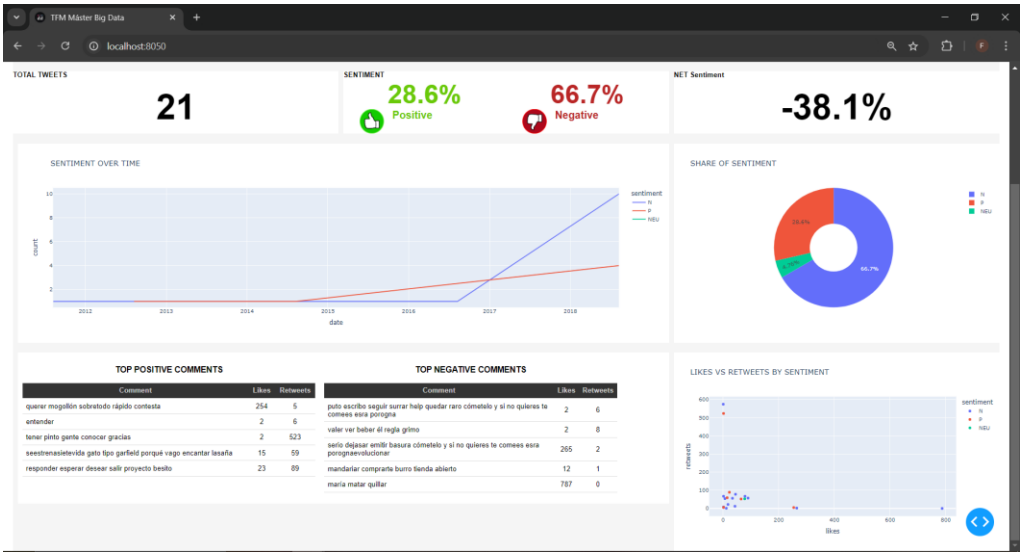


Figura 9-4. Interfaz principal con los resultados de análisis

9.3 Anexo 3: Extracción de los datos de Twitter

Para extraer nuevos datos se ejecuta el archivo scraping.py. Este archivo contiene el código necesario para realizar el scraping de tweets en Twitter [1].

```
tweet_text_element = tweet_element.find_element(By.CSS_SELECTOR, '[data-testid="tweetText"]')
tweet_text = tweet_text_element.text

language = tweet_text_element.get_attribute('lang')

if language == 'es':
    try:
        likes = tweet_element.find_element(By.CSS_SELECTOR,  '[data-testid="like"]').text
    except NoSuchElementException:
        likes = "0"

    try:
        retweets = tweet_element.find_element(By.CSS_SELECTOR,  '[data-testid="retweet"]').text
    except NoSuchElementException:
        retweets = "0"

    try:
        views = tweet_element.find_element(By.CSS_SELECTOR, '[data-testid="app-text-transition-container"]').text
    except NoSuchElementException:
        views = "0"

    tweets.append({
        'tweet_text': tweet_text,
        'likes': likes,
        'retweets': retweets,
        'views': views
    })
```

Figura 9-5. Fragmento de código de Web Scraping

9.4 Anexo 4: Limpieza de datos

Para limpiar los nuevos datos se ejecuta el archivo normalizer.py. Este archivo contiene el código necesario para realizar la limpieza de los nuevos tweets.

```python
def clean(text):
    # Eliminar URLs
    text = re.sub(r'https?://\S+|www\.\S+', '', text)
    # Eliminar palabras con @ y #
    text = re.sub(r'[@#]\w+', '', text)
    # Tratar los emojis
    text = emoji.demojize(text, language='es')
    # Eliminar puntuaciones
    text = re.sub(r'[^\w\sáéíóúüñ]', ' ', text)
    # Sustituir el '_' por un espacio en blanco al tratar los emojis
    text = text.replace('_', ' ')
    # Sustituir los '\n' por espacios en blanco
    text = text.replace('\n', '')
    # Convertir a minúscula
    text = text.lower()
    # Procesar el texto con spacy
    doc = nlp(text)
    # Lematización y filtrado de stopwords
    lemmatized_tokens = [
        token.lemma_ for token in doc
        if token.lemma_ not in stop_words and token.lemma_ != '' and token.lemma_ not in english_words and token.pos_ in
        ['NOUN', 'ADJ', 'VERB', 'ADV']]
    # Unir tokens en un string
    cleaned_text = ' '.join(lemmatized_tokens)

    return cleaned_text
```

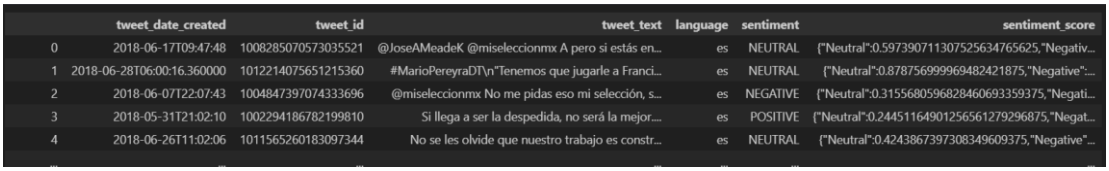
Figura 9-6. Fragmento de código de la limpieza de datos

9.5 Anexo 5: Descripción de los datos

Se utilizaron varios conjuntos de datos de diversas fuentes para el análisis de sentimientos. A continuación, se describe cada conjunto, detallando sus características, número de registros y distribuciones de clases

- **Tweets Sentiment Worldcup (betsentiment-ES-tweets-sentiment-worldcup.csv) [45]**

  Conjunto de datos que recopila tweets relacionados con la Copa Mundial de la FIFA, centrados en opiniones sobre partidos, equipos y eventos del torneo.



	tweet_date_created	tweet_id	tweet_text	language	sentiment	sentiment_score
0	2018-06-17T09:47:48	1008285070573035521	@JoseAMeadeK @miseleccionmx A pero si estás en...	es	NEUTRAL	{"Neutral":0.5973907113075256347656 25,"Negativ...
1	2018-06-28T06:00:16.360000	1012214075651215360	#MarioPereyraDT\n"Tenemos que jugarle a Franci...	es	NEUTRAL	{"Neutral":0.8787569999694842187 5,"Negative":...
2	2018-06-07T22:07:43	1004847397074333696	@miseleccionmx No me pidas eso mi selección, s...	es	NEGATIVE	{"Neutral":0.3155680596828460693359375,"Negati...
3	2018-05-31T21:02:10	1002294186782199810	Si llega a ser la despedida, no será la mejor....	es	POSITIVE	{"Neutral":0.2445116490125656127929687 5,"Negat...
4	2018-06-26T11:02:06	1011565260183097344	No se les olvide que nuestro trabajo es constr...	es	NEUTRAL	{"Neutral":0.4243867397308349609375,"Negative"...

Figura 9-7. Muestra de Tweets Sentiment Worldcup

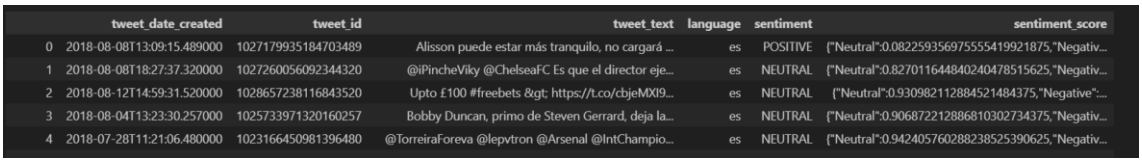
- **Número de registros**. Aproximadamente 1072461 tweets.
- **Distribución de clases**. NEUTRAL (791427 tweets), POSITIVE (156474 tweets), NEGATIVE (116954 tweets) y MIXED (7606 tweets)**.**
- **Características**.

```
 #   Column             Non-Null Count    Dtype
---  ------             --------------    -----
 0   tweet_date_created  1072461 non-null  object
 1   tweet_id            1072461 non-null  int64
 2   tweet_text          1072460 non-null  object
 3   language            1072461 non-null  object
 4   sentiment           1072461 non-null  object
 5   sentiment_score     1072461 non-null  object
dtypes: int64(1), object(5)
```

Figura 9-8. Características de Tweets Sentiment Worldcup

- **Tweets Sentiment Teams (betsentiment-ES-tweets-sentiment-teams.csv) [46]**

Conjunto de datos enfocado en opiniones sobre equipos de fútbol.



	tweet_date_created	tweet_id	tweet_text	language	sentiment	sentiment_score
0	2018-08-08T13:09:15.489000	1027179935184703489	Alisson puede estar más tranquilo, no cargará ...	es	POSITIVE	{"Neutral":0.0822593569755554199218 75,"Negativ...
1	2018-08-08T18:27:37.320000	1027260056092344320	@iPincheViky @ChelseaFC Es que el director eje...	es	NEUTRAL	{"Neutral":0.8270116448402404785156 25,"Negativ...
2	2018-08-12T14:59:31.520000	1028657238116843520	Upto £100 #freebets > https://t.co/cbjeMXI9...	es	NEUTRAL	{"Neutral":0.9309821128845214843 75,"Negative":...
3	2018-08-04T13:23:30.257000	1025733971320160257	Bobby Duncan, primo de Steven Gerrard, deja la...	es	NEUTRAL	{"Neutral":0.9068722128868103027343 75,"Negativ...
4	2018-07-28T11:21:06.480000	1023166450981396480	@TorreiraForeva @lepvtron @Arsenal @IntChampio...	es	NEUTRAL	{"Neutral":0.9424057602882385253906 25,"Negativ...

Figura 9-9. Muestra de Tweets Sentiment Teams

- **Número de registros**. Aproximadamente 132707 tweets.
- **Distribución de clases**. NEUTRAL (111334 tweets), POSITIVE (11004 tweets), NEGATIVE (9489 tweets) y MIXED (880 tweets).
- **Características**.

```
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   tweet_date_created  132707 non-null  object
 1   tweet_id            132707 non-null  int64
 2   tweet_text          132707 non-null  object
 3   language            132707 non-null  object
 4   sentiment           132707 non-null  object
 5   sentiment_score     132707 non-null  object
dtypes: int64(1), object(5)
```

Figura 9-10. Características de Tweets Sentiment Teams

- **IMDB Dataset Spanish (IMDB_Dataset_SPANISH.csv) [47]**

  Conjunto de datos que incluye reseñas de películas en español del sitio IMDB, orientado al análisis de sentimientos.

review_en	review_es	sentiment	sentimiento
One of the other reviewers has mentioned that ...	Uno de los otros críticos ha mencionado que de...	positive	positivo
A wonderful little production. The filming tec...	Una pequeña pequeña producción.La técnica de f...	positive	positivo
I thought this was a wonderful way to spend ti...	Pensé que esta era una manera maravillosa de p...	positive	positivo
Basically there's a family where a little boy ...	Básicamente, hay una familia donde un niño peq...	negative	negativo
Petter Mattei's "Love in the Time of Money" is...	El "amor en el tiempo" de Petter Mattei es una...	positive	positivo

Figura 9-11. Muestra de IMDB Dataset Spanish

- **Número de registros**. Aproximadamente 50000 tweets.
- **Distribución de clases**. Positive (25000 tweets) y negative (25000 tweets).
- **Características**.

```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Unnamed: 0   50000 non-null  int64
 1   review_en    50000 non-null  object
 2   review_es    50000 non-null  object
 3   sentiment    50000 non-null  object
 4   sentimiento  50000 non-null  object
dtypes: int64(1), object(4)
```

Figura 9-12. Características de IMDB Dataset Spanish

- **Segura Dataset (segura_dataset.csv) [48]**

  Conjunto de datos en español, utilizado para el análisis de sentimientos en redes sociales.

	Texto	Sentimiento
0	@jmartineze_ Haber como apoyaron a mayra goñi....	POSITIVO
1	@diarioojo CANTANTE MAYRA GOÑI\nYO TUVE UN PR...	NEUTRAL
2	No solo de ahí salieron los q bajaron la cuent...	NEUTRAL
3	Sé que dije que iba a estár inactiva en tw per...	NEUTRAL
4	@Motta01Alonso @munchii_tx Alonso mejor hablem...	NEGATIVO

Figura 9-13. Muestra de Segura Dataset

  - **Número de registros**. Aproximadamente 1000 tweets.

  - **Distribución de clases**. POSITIVO (431 tweets), NEUTRAL (358 tweets) Y NEGATIVO (210 tweets).

  - **Características**.

```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Texto        1000 non-null   object
 1   Sentimiento  999 non-null    object
dtypes: object(2)
```

Figura 9-14. Características de Segura Dataset

- **TASS2019 Country ES Train (TASS2019_country_ES_train.xml) [49]**

  Conjunto de datos del Taller de Análisis de Sentimientos en la Web (TASS) [9], centrado en el análisis de sentimientos en español.

	tweet_text	sentiment
0	@myendlesshazza a. que puto mal escribo b. me ...	N
1	Quiero mogollón a @AlbaBenito99 pero sobretodo...	P
2	Vale he visto la tia bebiendose su regla y me ...	N
3	@Yulian_Poe @guillermoterry1 Ah. mucho más por...	P
4	@toNi_end seria mejor que dejasen de emitir es...	N

Figura 9-15. Muestra de TASS2019 Country ES Train

- **Número de registros**. Aproximadamente 1125 tweets.
- **Distribución de clases**. P (354 tweets), N (474 tweets), NONE (157 tweets) y NEU (140 tweets).
- **Características**.

```
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   tweet_text  1125 non-null   object
 1   sentiment   1125 non-null   object
dtypes: object(2)
```

Figura 9-16. Características de TASS2019 Country ES Train

[PÁGINA INTENCIONADAMENTE EN BLANCO]